Università
Ca'Foscari
Venezia

Corso di Dottorato di ricerca

in Informatica

ciclo 38

Tesi di Ricerca

Explainable Classification and the Limits of Language Understanding in Language Models

SSD: INF/01

Coordinatore del Dottorato
Prof. Andrea Torsello

Supervisore
Prof. Andrea Albarelli

Dottorando
Alessandro Zangari
Matricola 956721

Università Ca' Foscari Venezia

Dottorato di Ricerca in Informatica, 38° Ciclo

(A.A. 2022/2023– 2024/2025)

Explainable Classification and the Limits of Language Understanding in Language Models

Settore scientifico disciplinare di afferenza: INF/01

Tesi di dottorato di Alessandro Zangari
Matr. 956721

Tutore del dottorando

Prof. Andrea Albarelli

Coordinatore del dottorato

Prof. Andrea Torsello

December, 2025

Author's e-mail: alessandro.zangari@unive.it

Author's address:

Dipartimento di Scienze Ambientali, Informatica e Statistica
Università Ca' Foscari Venezia
Via Torino, 155
30172 Venezia Mestre – Italia
web: https://www.dais.unive.it

I primarily dedicate this work to my family for their continuous support, both emotional and financial, and to my friends for sharing a good laugh and occasionally reminding me of the importance of what I was doing. I believe that without them, this journey would have been much harder to complete.

I am grateful to my supervisor, Andrea Albarelli, for his advice, his positive attitude toward my research, and for helping me navigate the administrative hurdles we encountered from the start. A big thank you to my fellow PhD students and colleagues at Digital Strategy Innovation for the lunches, social moments, and experiences we shared, including the occasional deadlines and anxieties that are inevitably part of the PhD package.

I also deeply appreciate the NLP group at Cardiff University for hosting me during an excellent six months of research (and a lot of cooking). To all the friends I met there: I sincerely hope our friendships will be long-lasting.

Finally, a special thanks to Matteo Marcuzzo and Matteo Rizzo, two colleagues and friends with whom I shared quite a few collaborations, summer schools, and holidays (I promise, they were separate events). Despite some highs and lows, learning from one another has made completing this PhD even more meaningful.

December 2025 – Alessandro Zangari

Abstract

The widespread adoption of powerful yet opaque deep-learning-based Language Models (LMs) has transformed the field of Natural Language Processing (NLP), but it has also increased the demand for solutions that explain their decision-making processes.

My work contributes to this area on two fronts. First, we develop text classification systems and evaluate them in challenging multilingual and hierarchical contexts, introducing novel datasets and a Transformer architecture specifically designed to manage labels organized hierarchically. This architecture demonstrates promising performance on complex, noisy datasets, surpassing several baselines.

Second, we address the challenge of explainability through two main contributions. We first demonstrate the effectiveness of a transparent system in a real-world image classification case study. Building on this, we propose a more general, modular framework that integrates opaque components within a transparent structure. This approach demonstrates a viable compromise between performance and the ability to faithfully explain predictions, and was validated on two text classification tasks.

The final part of my work examines the limits of generative Large LMs (LLMs) in understanding nuanced language, specifically on the task of classifying and explaining sentences containing humorous wordplay, such as puns. Our findings reveal that LLMs struggle to accurately identify valid puns within their context. Furthermore, their explanations often misinterpret contextual cues and phonetic elements, suggesting that their understanding of such language devices may be more superficial than what previous studies have claimed. Consequently, we highlight weaknesses in existing benchmarks for measuring these abilities.

My research work thus contributes both practical solutions for text classification and a critical analysis of Explainable AI practices and LLM capabilities, emphasizing the necessity of developing AI that is not only powerful but also transparent and reliable, while cautioning against unrealistic expectations regarding their current abilities.

Contents

List of Figures

List of Tables

Preface

This dissertation summarizes the work conducted during my postgraduate studies in the fields of Natural Language Processing (NLP) and Explainable AI (XAI). My research journey, especially in its early phases, was marked by an exploration of multiple topics, often motivated by collaborations and cross-disciplinary interests. This varied research path proved to be beneficial, offering me the chance to see the field from different perspectives and to learn from collaborations with fellow researchers.

My work began with a focus on text classification in multilingual settings, which led to comprehensive reviews of the field [2, 1] and the creation of various benchmark datasets [361]. Building on this foundation, this initial research drew my attention to the frequent use of taxonomies for organizing labels. This led me to contribute to a project developing an industrial classification system for support tickets, where we designed a solution to handle a two-level label hierarchy. This work resulted in my first publications on the topic [4, 6], and later, a comprehensive review of hierarchical text classification and the curation of several public datasets [7, 374].

Subsequently, I collaborated with the Ca' Foscari Department of Management on a pilot project to classify fruit quality in wholesale markets using automated methods while ensuring transparency for end users [8]. This work marked the beginning of my exploration into the field of XAI.

Given my primary interest in NLP, I continued to pursue research at its intersection with XAI, exploring the challenges of both explaining black-box models and evaluating the explanations themselves. Motivated by the gap between simple, transparent models and high-performance deep learning, I co-authored a pragmatic, explainability-by-design framework that embeds opaque components within a more transparent structure [9]. My final research project was a collaboration with researchers from Cardiff University in which we evaluated text comprehension in modern generative AI. Using a classification and explanation paradigm based on self-rationalization, we probed the reasoning of LLMs on a nuanced task, discovering several limitations of these models [11].

Through these experiences, I have gained valuable insights into the complexities of AI systems, their strengths, and limitations. This journey has reinforced my belief in the importance of critical evaluation and the pursuit of AI that is not just powerful, but also transparent and genuinely understood.

Published Papers

[1] Andrea Gasparetto, Matteo Marcuzzo, Alessandro Zangari and Andrea Albarelli. 'A Survey on Text Classification Algorithms: From Text to Predictions'. In: *Information* 13.2 (Feb. 2022). Publisher: Multidisciplinary Digital Publishing Institute, p. 83. ISSN: 2078-2489. DOI: 10.3390/info13020083.

[2] Andrea Gasparetto, Alessandro Zangari, Matteo Marcuzzo and Andrea Albarelli. 'A survey on text classification: Practical perspectives on the Italian language'. In: *PLOS ONE* 17.7 (July 2022). Publisher: Public Library of Science, pp. 1–46. DOI: 10.1371/journal.pone.0270904.

[3] Matteo Marcuzzo, Alessandro Zangari, Andrea Albarelli and Andrea Gasparetto. 'Recommendation Systems: An Insight Into Current Development and Future Research Challenges'. In: *IEEE Access* 10 (July 2022), pp. 86578–86623. DOI: 10.1109/ACCESS.2022.3194536.

[4] Matteo Marcuzzo, Alessandro Zangari, Michele Schiavinato, Lorenzo Giudice, Andrea Gasparetto and Andrea Albarelli. 'A multi-level approach for hierarchical Ticket Classification'. In: *Proceedings of the Eighth Workshop on Noisy User-generated Text (W-NUT 2022)*. Gyeongju, Republic of Korea: Association for Computational Linguistics, Oct. 2022, pp. 201–214. URL: https://aclanthology.org/2022.wnut-1.22.

[5] Matteo Rizzo, Matteo Marcuzzo, Alessandro Zangari, Andrea Gasparetto and Andrea Albarelli. 'Fruit ripeness classification: A survey'. In: *Artificial Intelligence in Agriculture* 7 (Mar. 2023), pp. 44–57. ISSN: 2589-7217. DOI: 10.1016/j.aiia.2023.02.004.

[6] Alessandro Zangari, Matteo Marcuzzo, Michele Schiavinato, Andrea Gasparetto and Andrea Albarelli. 'Ticket automation: An insight into current research with applications to multi-level classification scenarios'. In: *Expert Systems with Applications* 225 (Sept. 2023), p. 119984. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2023.119984.

[7] Alessandro Zangari, Matteo Marcuzzo, Matteo Rizzo, Lorenzo Giudice, Andrea Albarelli and Andrea Gasparetto. 'Hierarchical Text Classification and Its Foundations: A Review of Current Research'. In: *Electronics* 13.7 (Jan. 2024). Publisher: Multidisciplinary Digital Publishing Institute, p. 1199. ISSN: 2079-9292. DOI: 10.3390/electronics13071199.

[8] Matteo Rizzo, Matteo Marcuzzo, Alessandro Zangari, Michele Schiavinato, Andrea Albarelli and Andrea Gasparetto. 'Stop Overkilling Simple Tasks with Black-Box Models, Use More Transparent Models Instead'. In: *Pattern Recognition and Artificial Intelligence*. Ed. by Christian Wallraven, Cheng-Lin Liu and Arun Ross. Singapore: Springer Nature, Feb. 2025, pp. 279–293. ISBN: 978-981-97-8702-9. DOI: 10.1007/978-981-97-8702-9_19.

[9] Alessandro Zangari, Matteo Marcuzzo, Matteo Rizzo, Andrea Albarelli and Andrea Gasparetto. 'Crossing the Divide: Designing Layers of Explainability'. In: *Artificial Intelligence and Soft Computing*. Ed. by Leszek Rutkowski, Rafał Scherer, Marcin Korytkowski, Witold Pedrycz, Ryszard Tadeusiewicz and Jacek M. Zurada. Cham: Springer Nature Switzerland, Feb. 2025, pp. 253–265. ISBN: 978-3-031-84353-2. DOI: 10.1007/978-3-031-84353-2_22.

[10] Matteo Marcuzzo, Alessandro Zangari, Andrea Albarelli, Jose Camacho-Collados and Mohammad Taher Pilehvar. 'Morables: A Benchmark for Assessing Abstract Moral Reasoning in LLMs with Fables'. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2025. Ed. by Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose and Violet Peng. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 27715–27739. ISBN: 979-8-89176-332-6. DOI: 10.18653/v1/2025.emnlp-main.1411.

[11] Alessandro Zangari, Matteo Marcuzzo, Andrea Albarelli, Mohammad Taher Pilehvar and Jose Camacho-Collados. 'Pun Unintended: LLMs and the Illusion of Humor Understanding'. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2025. Ed. by Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose and Violet Peng. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 27924–27959. ISBN: 979-8-89176-332-6. DOI: 10.18653/v1/2025.emnlp-main.1419.

1

Introduction

My doctoral studies have coincided with one of the most transformative moments in the recent history of Artificial Intelligence (AI). The advancement of Language Models (LMs and foundational LLMs) has marked a paradigm shift, evolving these systems from tools fine-tuned on specific tasks into general-purpose reasoning machines. By scaling neural architectures and the underlying training data to unprecedented levels, these models have introduced a new approach to language understanding, arguably as significant as the leap from shallow to deep learning. When I began my PhD in late 2022, LLMs were an active area of research; however, few could have anticipated how quickly these models would become a major focus of research in both academia and industry. This change occurred alongside their rapid adoption in industrial applications and a massive growth in public awareness, the latter largely driven by their visibility as chatbots. These recent foundational LMs are a category of models built on the seminal Transformer architecture [128]. This architecture revolutionized NLP by enabling models to learn sequential and contextual dependencies without requiring sequential data processing, a significant bottleneck in previous methods. Instead, it introduced a self-attention mechanism that learns these dependencies in parallel. This design enabled more scalable training, paving the way for the massive models we see today.

This rapid evolution, however, has amplified a central tension in the contemporary AI landscape: the pursuit of performance through increasingly complex, opaque models, set against the need for transparency and accountability in many critical domains. While research pushes the boundaries of NLP through transfer learning, improved pre-training, and more efficient architectures, Explainable AI (XAI) pursues the ambitious goal of making these black boxes understandable to human observers. However, the very nature of deep learning, with its distributed and sub-symbolic representations, makes it impossible to reverse-engineer the inference process to find a single, causal path for a system's decision [367]. Consequently, XAI must operate with multiple practical definitions of "explanation", grounded in approximated interpretations of a model's internal mechanisms or in pragmatic analyses designed to answer specific questions.

Organization of the Dissertation

This dissertation presents my path through this landscape, roughly following the chronological order of my research, starting with the benchmarking and development of novel text classifiers and then turning to the pursuit of XAI paradigms that allow for a clearer understanding of machine learning models' behavior.

Part I lays the groundwork by reviewing the relevant literature on these topics and providing the necessary background for the contributions that follow. Chapter 2 specifically delves into the foundational NLP themes relevant to my research, most of which were explored during my first year of PhD study [1]. Following this, Chapter 3 presents an up-to-date overview of XAI, its main challenges, and the future directions that motivated my work.

The first phase of my research, detailed in Part II, was dedicated to building effective classification systems with industrial applications in mind. Text classification is an essential task with a wide range of specialized applications, from sentiment analysis to text summarization. A key focus of my initial work was the use of LMs in multilingual settings, particularly their performance in categorizing Italian text. At that time, the NLP community was actively developing and exploring the applications of pre-trained Transformer-based models such as BERT [192] and T5 [292], which were replacing older recurrent architectures. Hence, my work in Chapter 4 presents a performance benchmark of LMs on text classification in three languages [2], which also resulted in the publication of new multilabel datasets extracted from Wikipedia [361]. This study highlighted the challenges of adapting NLP pipelines to multilingual contexts and provided the practical knowledge needed to address a more complex classification setting with hierarchically labeled content.

As humans, we instinctively organize the world around us by moving from broad categories to more specific ones. This natural way of structuring knowledge is why hierarchical categorization schemes (i.e., taxonomies) are commonly used in many classification tasks. Hierarchical labels carry relational information that can be leveraged to improve performance, for example, by ensuring consistency across different levels of the hierarchy. Furthermore, specialized metrics can be used to quantify the severity of a misclassification based on the distance between the true and predicted labels in the hierarchy tree, offering a more nuanced evaluation than standard "flat" classification metrics. Motivated by these considerations, Chapter 5 describes our development of a practical classification system for technical support tickets — a setting further complicated by noisy, jargon-filled text and a lack of labeled data. This effort led to a novel dataset and a new Transformer-based architecture tailored for tasks with a hierarchical label structure, which stimulated further co-authored publications on the topic [6, 4, 7], including a public collection of hierarchical datasets [374].

During the initial phase of my doctoral work, I was increasingly exposed to the scientific and public debate advocating for more transparent systems. This debate is driven both by new regulations in the European Union and by the accelerating trend

of building larger, yet more opaque, models. This context motivated the research presented in Part III, which delves into the challenge of explaining machine learning methods.

One of the primary difficulties in the field of XAI is the lack of standardized terminology, as many concepts are used inconsistently. The term "explanation" itself is semantically overloaded, since its precise meaning is context-dependent and changes according to the specific question it aims to answer. This ambiguity, combined with the irreversible nature of computation in deep learning models, means the field lacks a strong consensus on what constitutes a proper explanation. Furthermore, evaluation objectives like faithfulness (whether the explanation reflects the model's true reasoning) and plausibility (whether the explanation makes sense to humans) are also challenging to define and measure rigorously. The literature review in Chapter 3 details these terminologies and establishes the formal framework used throughout my research.

In this landscape, my first contribution [8], presented in Chapter 6, demonstrates how a simple algorithm (a decision tree classifier), paired with an intuitive data representation and expert-guided preprocessing, can provide a transparent and effective solution with minimal performance loss compared to deep learning alternatives. The goal of this project was to practically demonstrate how explainability can be enhanced by pairing domain knowledge with a machine learning algorithm selected to be no more complex than the task requires, an approach advocated by Rudin [191]. This work was also motivated by a collaboration with a local fruit market as a case study in grounding research in tangible problems. Following this, Chapter 7 proposes a more general *explainability-by-design* framework that allows the use of opaque components (e.g., deep learning methods) while still demonstrating a good trade-off between performance and faithfulness of explanation on two text classification tasks [9].

Finally, Chapter 8 completes my work with a project at the intersection of machine comprehension, prompting for LLMs, and XAI [11]. In this chapter, we study the limits of LLM text understanding by assessing both classification performance and the quality of model-generated explanations. We conduct this study on three novel collections of short sentences containing humorous puns, tasking various LLMs to detect and explain them using specific semi-structured rationales. Our results show that while LLMs can identify common forms of wordplay, they struggle to distinguish authentic pun contexts from sentences that are merely similar to known puns, often failing to provide coherent explanations. We argue that this stems from a lack of genuine understanding of the linguistic mechanisms that create humorous wordplay. Moreover, this work argues that challenging benchmarks for language sensitivity (as opposed to just reasoning and knowledge) are urgently needed as these systems become increasingly prevalent.

Beyond the primary research thread of my doctoral studies, I also took part in several other collaborations that resulted in co-authored publications, some of which are currently in review. These include a work on AI-based fruit ripeness

classification [5] that stimulated the research in Chapter 6, and a minor contribution to the theoretical XAI framework partly introduced in Chapter 3. My frequent collaboration with my colleague Matteo Marcuzzo also led to several publications, and the portions of that joint work most relevant to my primary line of research are discussed within the respective chapters. Furthermore, I explored multidisciplinary collaborations, some of which involved applying computer vision to industrial problems, including pattern recognition in textiles and the use of XAI methods for defect detection in industrial inspection. In a separate project, I provided NLP expertise to an interdisciplinary team, using clustering methods and LLMs to support their analysis of scholarly literature. Finally, my early research also included work on recommendation systems, which led to a publication on the topic [3]. Related to this research, I also contributed to the development of a proof-of-concept for a destination recommendation platform as part of a larger team effort within a regional project on digital tourism.

Taken together, the works presented here reflect my two primary research interests: the practical role of NLP and the need to assess the transparency and linguistic understanding of LMs. Thus, these two themes provide the structure for the main parts of this dissertation. The first, detailed in Part II, addresses the challenge of *building* better classification approaches in multilingual and hierarchical settings. The second, in Part III, confronts the challenge of *understanding* these systems by developing new frameworks for explainability and probing their limitations.

I

Background and State of the Art

2

Foundations of NLP for Text Classification

Natural Language Processing (NLP) has undergone a paradigm shift, catalyzed by the developments of deep-learning-based Language Models (LMs) and, more recently, the rising interest in generative Large LMs (LLMs), which scale LMs to massive sizes. This chapter introduces foundational concepts and modern developments in NLP, with a particular focus on Text Classification (TC) tasks.

TC encompasses a variety of procedures that roughly share the common objective of assigning predefined labels to a given text. Over the years, TC procedures have evolved from simple, rule-based systems to highly specialized deep learning (DL) architectures. The latter have demonstrated an unprecedented ability to capture a text's underlying semantics, using this understanding in the classification process. Classic examples of TC applications include information retrieval, topic labeling, sentiment analysis, and news classification. However, the practical applications of TC also extend to related tasks, such as multiple-choice question answering and extractive summarization systems. In these cases, the intuitive notion of a "label" is replaced with a choice among candidates (e.g., an answer or a sentence to include in a summary). Different classification settings are possible: a document may be assigned a single label (multiclass classification), multiple independent labels (multilabel), or labels organized in a hierarchy. Given the prevalence of such structures in real-world data, hierarchical classifiers have been widely researched.

This chapter will introduce approaches to TC by covering the essential components of a typical TC pipeline, many of which are fundamental to a wide range of NLP tasks. It then discusses the specific principles behind TC in hierarchical settings.

2.1 Text Classification Pipeline

Both modern LMs and older approaches to TC rely on a process that can be broken down into three essential steps: (i) *preprocessing*, which programmatically cleans and prepares input data for subsequent steps; (ii) *representation* of the text into a vectorial form that machines can process; and (iii) *classification*, where this repres-

entation is used to produce an outcome. This section will describe each of these steps, outlining their respective goals and challenges, particularly in handling different languages, and providing a historical perspective on their development.

2.1.1 Preprocessing

Raw text is *unstructured* and lacks a straightforward numerical representation (unlike, for example, the RGB triples that encode visual data). From a linguistic perspective, language is governed by a complex structure that is intuitive to a native speaker but not to a machine.

It is therefore necessary to project text into an appropriate feature space so that a learning algorithm can process it. In this section, we discuss the procedures that prepare textual data for this projection, whether this is done through manual feature extraction (common in traditional methods) or automatically (as with recent DL approaches).

2.1.1.1 Normalization

In traditional NLP pipelines, especially those reliant on statistical models, like Bag-of-Words, aggressive *text normalization* is a prerequisite for effective feature engineering [454]. Procedures such as lowercasing, stopwords removal, stemming, and lemmatization are essential for reducing the vocabulary size and consolidating related terms into a single feature. For instance, reducing "child" and "children" to a common stem helps some simpler methods, which lack semantic understanding, to recognize their relationship [51, 44]. Similarly, removing high-frequency, low-information words like articles and pronouns (stopwords) is intended to reduce the number of different elements to be projected in the feature space and improve its discriminative power [87, 236]. However, applying these "destructive" normalization techniques can create a significant mismatch between the training and inference data distributions, potentially degrading model performance. Therefore, these preprocessing techniques should be considered carefully on a case-by-case basis, as modern NLP models are trained to extract context from grammatically and morphologically sound sentences. LLMs, in particular, are pre-trained on vast, web-scale corpora (i.e., collections of digitized textual data) of natural, grammatically rich text. They learn to distinguish which features are relevant for a given task, including capitalization (which can denote proper nouns) and punctuation (which structures sentences), elements often treated as "noise" by older, less capable models. LLMs can also perform complex data cleaning and repair tasks; for example, they can be prompted to correct grammatical errors, standardize formats, or refine noisy outputs from other Artificial Intelligence (AI) systems. This shift to bigger, more capable systems motivates the diminishing role of classic normalization and places an even greater emphasis on the subsequent step, described in the next section.

2.1.1.2 Segmentation and tokenization

An essential operation in an NLP pipeline is *text segmentation*, which involves breaking a stream of text into smaller chunks. This process decomposes a text to create a "vocabulary" of all unique terms. At a practical level, this vocabulary is used to generate an index-based mapping between the text segments and a numerical representation suitable for a given feature extraction technique. While early techniques often segmented text into whole words, modern methods typically split text into sub-word units, known as *tokens*. Tokens produced by *tokenizers* are mapped to *embeddings* — rich vectorial representations, which will be covered in § 2.1.2. As each token in the vocabulary is encoded in a potentially large vector, and models cannot handle vocabularies of arbitrary size due to computational and memory constraints, most modern LMs require a fixed-size vocabulary, typically created by truncating it based on some criteria. In the following sections, we equate *tokenization* and *segmentation*, as they refer to the same procedure in practice.

Rule-based segmentation or pre-tokenization Historically, segmentation has been approached with *rule-based* methods. In writing systems that use whitespace, a straightforward segmentation can be achieved by splitting text around spaces, punctuation, and contractions. This intuitive method has seen numerous refinements, often incorporating language-specific grammatical rules [321]. While not flawless, such techniques were widely used in traditional language modeling, providing an acceptable approximation of morphemes that balanced linguistic relevance with purely typographic splits [321]. However, a key limitation of this static approach is its inability to adapt to the specific word distribution of a given corpus. This rigidity often results in excessively large vocabularies and a significant loss of information when encountering out-of-vocabulary (OOV) terms.

In contemporary literature, these methods are often referred to as "pre-tokenizers", reflecting their role as a preliminary step for the more advanced, data-driven algorithms discussed next. Modern tokenizers typically require this initial segmentation because they cannot operate on raw text. This pre-tokenization step has a profound effect on the final output, primarily because it establishes hard boundaries that subsequent tokenization algorithms cannot cross (the pre-token boundary constraint) [356, 457]. For instance, using a standard whitespace pre-tokenizer makes it impossible for many tokenization algorithms to form a single token for common multi-word expressions (e.g., "in spite of"), as the spaces ensure they are always split into at least three separate tokens [461].

Data-driven tokenization To overcome the inflexibility of rule-based methods, modern NLP has shifted towards data-driven tokenization. The guiding principle of this paradigm is that frequent words should be kept as single tokens, while rare words are decomposed into smaller, reusable sub-word fragments. This approach resolves the fundamental trade-off between vocabulary expressiveness and size. While

Table 2.1: Most widely adopted recent tokenization approaches.

Tokenizer	Training Procedure	Inference Procedure
BPE [101]	Iteratively merges the most frequent pair of adjacent tokens	Merge incrementally, keeping the merged term if in the vocabulary
BBPE [237]	Same as BPE, but operates on individual bytes instead of characters	Same as BPE
WordPiece [64]	Iteratively merges pairs that maximize the LM likelihood	Greedy segmentation that finds the longest possible tokens in the vocabulary
UnigramLM [154]	Starts with a large vocabulary and iteratively prunes tokens that least contribute to the LM likelihood	Finds the most probable segmentation using the Viterbi algorithm
SaGe [397]	Starts with a large vocabulary and iteratively prunes tokens based on a contextual signal (skip-gram objective)	Same as UnigramLM
SentencePiece [163]	Depends on the algorithm (package of optimized routines)	Depends on the algorithm

using whole words as tokens is linguistically intuitive, it leads to unmanageable vocabularies and OOV issues, especially in languages with rich morphology where derivations and inflections can cause the vocabulary to explode [321]. Conversely, using individual characters eliminates the OOV problem but produces much longer sequences, which increase processing time and make reconstructing semantics more difficult. Sub-word tokenization provides a robust, language-agnostic solution that effectively creates an open-vocabulary system from a fixed-size inventory of subwords. It has been argued that tokenization is the most critical language-dependent operation in any NLP pipeline, as the choice of tokenizer can significantly impact downstream performance, particularly in multilingual contexts [22, 430]. Table 2.1 provides a concise view of the main modern tokenizers.

2.1.1.3 Preprocessing across languages

Preprocessing is not a one-size-fits-all procedure; it must be adapted to the linguistic features of the language in question. For example, standard English-centric operations like lowercasing and stemming are irrelevant for logographic languages like Chinese, which lack capitalization and use minimal inflection. The task of segmentation also varies dramatically. Languages such as Chinese, Japanese, and Vietnamese do not use spaces to delimit words, requiring sophisticated segmentation algorithms, whereas English can often rely on simple whitespace splitting.

This linguistic diversity presents a central challenge in tokenization, centered on the choice of the basic textual unit. One approach, rooted in linguistics, aims to segment text into morphemes — the smallest units of a language that carry semantic meaning. This is appealing for morphologically rich languages (e.g., Turkish, Finnish) [34]. For example, the word *evlerden* ("from the houses") consists of three distinct morphemes: *ev* (house), *-ler* (plural), and *-den* (from), a structure a simple whitespace tokenizer would miss. However, this linguistically-motivated segmentation is notoriously difficult and costly due to language-specific irregularities [37, 70, 34].

In contrast, modern sub-word tokenizers (e.g., BPE, WordPiece) prioritize statistical efficiency over linguistic significance [321]. These data-driven methods learn to segment text by identifying frequently co-occurring character sequences within a corpus. The resulting tokens do not necessarily correspond to morphemes but serve as efficient units that effectively mitigate the OOV problem, especially for complex languages. To further improve the efficiency of this strategy, new research is exploring alternative approaches capable of splitting pre-tokens across their boundaries [461].

Building on data-driven principles, research has explored tokenization strategies for multilingual and language-agnostic settings. For multilingual applications, a common approach is to train a single tokenizer with a shared vocabulary for multiple languages, capturing cross-lingual similarities. However, this method is prone to under-segmenting morphologically simple languages while over-segmenting complex ones. Furthermore, the shared vocabulary is frequently biased towards high-resource languages, a problem exacerbated by the disproportionate representation of languages within multilingual corpora [315, 186]. Despite these issues, the approach is widely used, especially with very large vocabularies [251]. A more radical solution is to create a truly language-agnostic system by decomposing text into its most fundamental representation: bytes. Byte-level tokenizers, for instance, apply algorithms like BPE directly to raw byte sequences [237]. This method is universally applicable with a small vocabulary (at most 256 unique bytes), but it produces significantly longer sequences, increasing computational costs. Furthermore, byte-level modeling is not entirely unbiased, as the underlying character encodings (e.g., Unicode) were not designed with linguistic principles in mind, leading to different byte representations for characters across languages, some of which require more than one byte [321]. Due to these challenges, the development of robust and fair tokenization for diverse linguistic contexts remains an active area of research.

2.1.2 Text Representation

Following preprocessing, a body of text is transformed into a list of standardized tokens. Before a computer can process this list, it must be expressed in numeric form. Feature representation techniques are a fundamental part of any NLP application, and their impact on performance often surpasses that of the choice of the final

classification model. In this section, we give an overview of both traditional methods and more recent approaches based on data-driven language modeling.

2.1.2.1 Language modeling

An important concept in text representation is that of *language models* (LMs). These are statistical models of natural language, and while the idea has existed for decades, it has been revitalized by the application of deep neural networks. Intuitively, a LM aims to capture the statistical properties of a language, often by learning to predict the probability of a token given its surrounding context [454].

Formally, a statistical LM can be described as a probability distribution over sequences of words. A simple example is the n-gram model, which uses the Markov assumption to approximate the probability of a sequence, as shown in Eq (2.1):

$$P(w_1 w_2 \dots w_m) \approx \prod_{i=1}^{m} P(w_i | w_{i-(n-1)} \dots w_{i-1}) \tag{2.1}$$

In other words, the probability that the ith word (w_i) is observed is only dependent on the preceding $n-1$ words. While simple, the core idea of using statistical distributions to model language is the foundation upon which even the most complex modern text representation techniques are built.

2.1.2.2 From sparse to dense representations

The practical application of language modeling to feature extraction has evolved from creating long, sparse vectors based on word counts to learning short, dense vectors that capture semantic meaning.

Bag-of-Words The *Bag-of-Words* (BoW) model is a traditional representation of text that disregards grammar and word order, reducing a document to an unordered collection of its words [240]. This representation is often implemented using the Term Frequency-Inverse Document Frequency (TF-IDF) weighting scheme [13]. TF-IDF assesses the relevance of each term t by weighting its frequency within a document d ($\mathrm{tf}_{t,d}$) while penalizing common words across the corpus, which typically have low discriminatory power. This results in a large, sparse vector for each document, with most entries being zero.

The definitions are shown in Eq (2.2), where $\mathrm{df}_{t,D}$ is the number of documents containing a term t, and D is the set of documents.

$$\mathrm{df}_{t,D} = |\{d \in D \mid t \in d\}| \qquad \mathrm{idf}_{t,D} = \log\left(\frac{|D|}{\mathrm{df}_{t,D}}\right) \tag{2.2}$$

The final TF-IDF weight for a term t found in document d is computed as: $w_{t,d} = \mathrm{tf}_{t,d} * \mathrm{idf}_{t,D}$. Often, terms with low frequency across all documents can be

dropped from the vocabulary. Ultimately, this text encoding generates large, sparse matrices of size proportional to the vocabulary length and the number of documents.

Word embeddings A major limitation of sparse representations such as BoW is their inability to capture semantic relationships; for instance, "car" and "automobile" are treated as separate features. To overcome this, the field has shifted to dense vector representations known as *word embeddings*, learned through self-supervision. This technique maps words to low-dimensional vectors of real numbers, resulting in shorter, denser representations that are independent of vocabulary size.

Word embeddings are typically learned using shallow neural networks based on the *distributional hypothesis*, which posits that a word's meaning can be inferred from its context. By training on large text corpora, models like Word2Vec [73], GloVe [79], and FastText [114] create vector spaces where semantically similar words are positioned closely together.

For instance, the Word2Vec *skip-gram model* frames this learning process as a self-supervised binary classification task using an objective known as *skip-gram negative sampling*. For each target word w in the text, a pair consisting of w and a word from its actual context c_+ forms a "positive example" (w, c_+). The algorithm samples several negative examples by pairing the same word w with random words c_- from the vocabulary that do not appear in its context window. This results in negative pairs (w, c_-^i) for $i = 1...k$, where k is the number of negative samples per positive example. The model's goal is to iteratively adjust the word vectors to maximize the similarity (e.g., dot product) for positive pairs and minimize it for negative pairs. This objective is optimized using a logistic regression loss (σ is a hyperparameter):

$$\mathscr{L}_{w2v} = -\left[\log\sigma(c_+ \cdot w) + \sum_{i=1}^{k} \log\sigma(-c_-^i \cdot w)\right] \tag{2.3}$$

The learning algorithm starts with two randomly initialized embedding matrices — one for target words and one for context words — and then iterates through the training corpus, using gradient descent to update the matrices to minimize the loss in Eq (2.3). This simple local objective, when applied over a massive corpus, effectively pulls the vectors for co-occurring words together while pushing unrelated ones apart. A popular alternative architecture, Continuous Bag-of-Words, performs a related task by instead predicting the center word from an aggregate of its surrounding context vectors.

Because these embeddings are static (meaning each word is mapped to a single vector representation), they can be pre-trained on vast, general-purpose corpora and released for public use. They can be used directly for tasks like semantic similarity searches or serve as powerful initial feature representations for downstream models. While effective off-the-shelf, these representations can also be enhanced with more discriminative features through tuning on domain-specific or task-specific data [252].

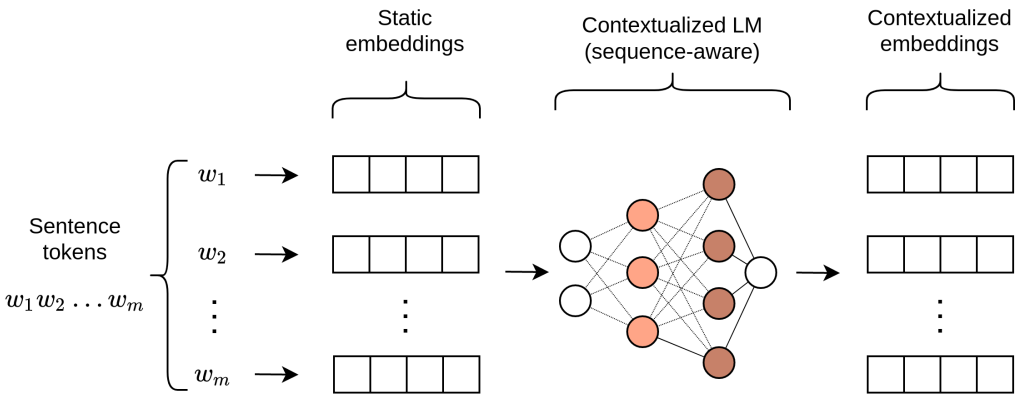


Figure 2.1: Sample generation process for contextualized embeddings.

Contextualized embeddings A key limitation of encoding text using embeddings like Word2Vec is that they are *static*: a word with multiple meanings (polysemy) is mapped to a single, fixed vector, regardless of its context. The paradigm shifted with the development of *contextualized embeddings*, where a deep neural network generates a unique vector for each token based on its surrounding sentence. This paradigm shift began with recurrent architectures, such as ELMo [146] and ULMFiT [153], but was fully realized with the introduction of the Transformer architecture [128], which enabled the creation of much deeper networks like BERT [192] and GPT [174].

These models are first *pre-trained* on a general language modeling task using massive text corpora. They are then adapted to specific downstream tasks through a process called *fine-tuning* [311, 454]. This approach, a form of transfer learning, has become the standard in modern NLP.

Several pre-training objectives have been proposed, but the most influential for bidirectional models like BERT is *Masked Language Modeling* (MLM). During MLM pre-training, the model is presented with a sentence where a fraction of the input tokens (e.g., 15%) have been randomly replaced with a special "[MASK]" token. The model's objective is to predict the original identity of these masked tokens by leveraging the surrounding unmasked context, which is optimized using a standard cross-entropy loss over the vocabulary. This "fill-in-the-blank" task forces the model to fuse information from both the left and right context simultaneously, enabling it to learn a bidirectional representation of language. The original BERT model was co-trained on a secondary task, *Next Sentence Prediction* (NSP), which required the model to determine if two sentences were sequential or not. However, subsequent models like RoBERTa [194] demonstrated improved performance by dropping the NSP task and training exclusively on MLM. These approaches stand in contrast to autoregressive models like GPT [174], which use a *Causal Language Modeling* (CLM) objective to predict the next token given only the preceding ones (and not using the right context like BERT). Other objectives seek to combine these paradigms; for

instance, XLNet [211] uses *Permutation Language Modeling* (PLM). This method learns bidirectional representations by predicting tokens in a permuted order, aiming to better capture long-range dependencies while retaining an autoregressive formulation. Regardless of its pre-training objective, the contextualizer LM acts as a sophisticated feature extractor. In contrast to static embeddings that map words to fixed vectors, a contextualizer LM generates rich, context-aware representations for each token, as illustrated in Fig 2.1.

2.1.2.3 Text representation in a multilingual context

The development of these dense, semantic embeddings also transformed multilingual NLP. As earlier approaches were not capable of expressing the semantic meaning of words, they can be largely seen as "detached" from language (with most linguistic aspects falling on tokenization approaches, as discussed). Therefore, the performance of methods such as BoW or TF-IDF relies largely on preprocessing and principled usage of statistical methods.

In contrast, modern representation techniques, like word embeddings, are designed to capture semantics, making them inherently language-specific and highly dependent on the scale and quality of their training corpora. For instance, the original BERT model required a massive dataset of 3.3 billion words to achieve its strong generalization capabilities [192, 90]. This dependency creates significant challenges. While the ideal for any given language is a high-quality *monolingual model* pre-trained exclusively on a massive native corpus, developing such models from scratch is often infeasible. The two major barriers are the scarcity of large, clean corpora and the immense computational cost of pre-training, which are prohibitive for all but the highest-resource languages. Consequently, researchers often must rely on pre-existing open-source models contributed by larger institutions.

To mitigate this resource gap, *multilingual models* like mBERT [192] and XLM-RoBERTa [251] have been proposed. The latter model is pre-trained on a concatenated corpus of over 100 languages. By learning from such diverse data, they develop cross-lingual representations that can enable powerful *zero-shot cross-lingual transfer*, where a model fine-tuned on a task in one language can be directly applied to another [201, 317]. However, multilingual models are not a universal solution and introduce a trade-off between monolingual specialization and multilingual generalization. Well-documented challenges include:

- **Performance Trade-off**: When a high-quality monolingual model is available, it almost always outperforms a multilingual model on downstream tasks in its specific language, particularly those requiring deep semantic understanding [201, 334, 317, 307].
- **Curse of Multilinguality**: A multilingual model's fixed capacity is spread thinly across many languages, which can dilute its effectiveness for any single one compared to a dedicated monolingual counterpart [277, 443].

- **Data Imbalance**: Multilingual corpora are often dominated by high-resource languages. This can lead to under-representation and weaker performance for low-resource languages, a bias that originates in data collection and propagates through the modeling pipeline [186].

Despite these challenges, current generative LLMs like those in the GPT [417] and Gemini [475] families are pre-trained on corpora that are orders of magnitude larger than those used for previous multilingual models, often incorporating not just a vast range of languages but also multimodal data, including source code and images. The sheer scale of these models appears to mitigate some classic challenges, such as the "curse of multilinguality" by providing enough capacity to handle numerous tasks and languages effectively. However, these foundational models still reflect the data imbalances of their training sources, and inconsistent performance on low-resource languages remains a persistent issue [470].

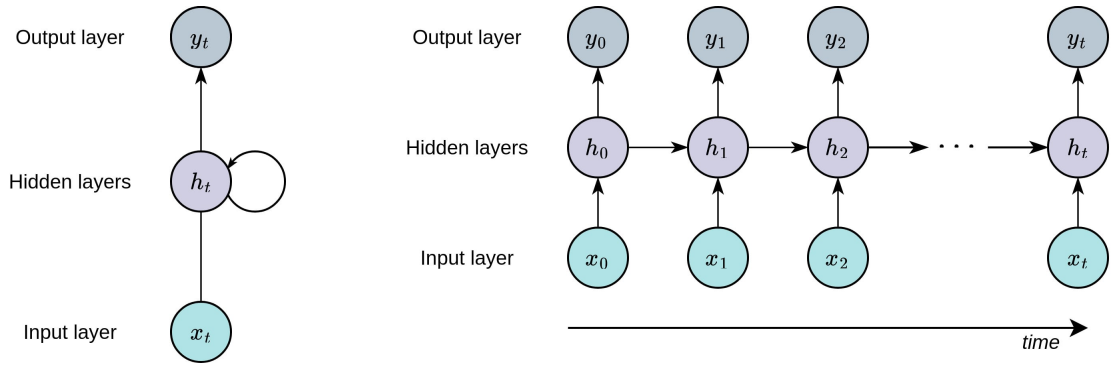
2.1.3 Underlying Neural Architectures

The remarkable progress from static embeddings to deep, contextualized representations is almost entirely attributable to the development of sophisticated neural network architectures. While early methods like Word2Vec used shallow networks [73], subsequent breakthroughs were achieved by employing deeper and more complex models. These architectures are the backbone of modern end-to-end TC systems, responsible for both learning rich representations and performing classification. We will now briefly review the most influential among them.

2.1.3.1 Recurrent neural networks (RNNs)

RNNs [17] are networks particularly well suited to environments that utilize sequential data, such as text or time series. This particular architecture allows these networks to retain a certain amount of "memory", making them able to extract latent relationships between elements within sequences. In practice, this is done by utilizing information from prior inputs to influence the current input and output of the network.

Simple RNNs for text processing are fed a sequence of word embeddings, which are processed sequentially. At each time step, the network receives both the next word vector as well as the hidden state of the previous time step (Fig 2.2). Unfortunately, because of their structure, standard RNN architectures are vulnerable to gradient-related issues, such as vanishing and exploding gradients [69]. In response to this issue, these architectures are frequently enhanced with *gating* mechanisms, the most popular being Long Short-Term Memory (LSTM) [25] and Gated Recurrent Unit (GRU) [80]. Briefly, these gates allow the network to control which and how much information to retain, such as to enable the modeling of long-term dependencies. RNNs that utilize these gates are often referred to with the acronym of the gate itself, i.e., LSTM networks and GRU networks.



(a) Compressed basic RNN. (b) Basic RNN shown unfolded at different time steps.

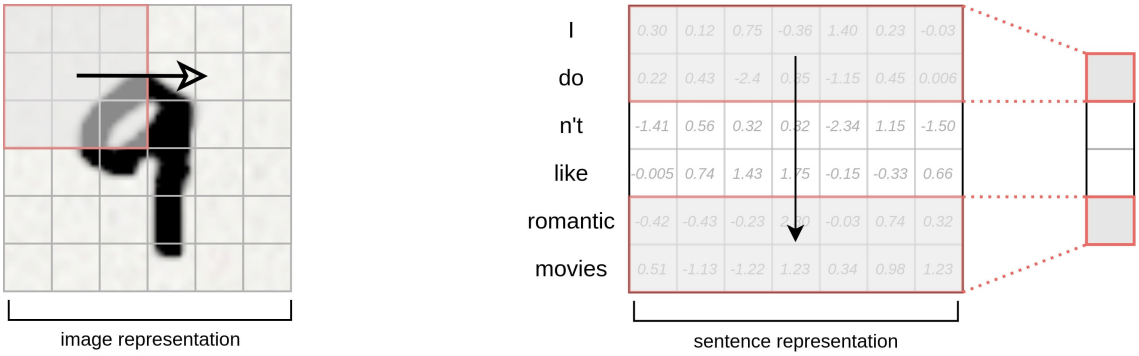
Figure 2.2: Exemplification of a simple RNN structure.

In the context of text representation, RNNs based on encoder-decoder architectures have been widely utilized to extract meaningful textual representations [77]. An encoder-decoder structure can be understood as an architecture by which inputs are mapped to a compressed representation (contained in the hidden states between the encoder and decoder). This representation should capture the most relevant features of the input, and can then be decoded for a variety of different tasks (e.g., translation). Autoencoders [20] are a particular class of encoder-decoder networks that attempt to regenerate the input exactly; they are particularly useful in creating efficient representations of unlabeled data.

In this context, the hidden states between the encoder and the decoder make for a semantically meaningful and compact representation of input words. The introduction of bidirectionality (influence in both left-to-right and right-to-left directions) in RNNs has also been proven to be beneficial and has been used to achieve notable results, such as the aforementioned ELMo [146], a bidirectional-LSTM-based LM which marked one of the first milestones towards the development of contextualized word embeddings. Nonetheless, RNNs have inherent limitations because of how they process data sequentially, making them ill-suited for parallelization and scaling. Furthermore, despite the improvements introduced by LSTM and GRUs, RNNs still struggle with long sequences because of memory constraints and their tendency to forget earlier parts of the sequence [69].

2.1.3.2 Convolutional neural networks (CNNs)

CNNs [19] are well-known neural architectures, originally devised for Computer Vision applications. However, these networks have since been extended to other fields, achieving excellent results in NLP tasks as well [78, 126]. The core structural element of CNNs is the *convolutional layer*, which applies a feature detector (*kernel* or *filter*) across subsets of the input (i.e., the convolution operation) to extract features. While this has a more intuitive interpretation in Computer Vision (where the filter



(a) A convolutional filter (top left) sliding across a digit from the MNIST dataset.

(b) A convolutional filter sliding across the vectorial representation of a sentence. Here, the filter is sliding in the conventional reading direction.

Figure 2.3: Application of convolutional filters in images and text.

moves across the image to search for features), the same reasoning can be applied to text. Intuitively, convolution as applied to images can be thought of as a weighted average of each pixel based on its neighborhood; the general idea of the process is outlined in Fig 2.3a. If we consider a vectorial representation of text (i.e., word embeddings), applying a filter as wide as the embedding size (a common approach) allows us to search for features within the sentence, as shown in Fig 2.3b. CNNs often use pooling operators (such as max or average) to reduce the size of the learned feature maps, as well as to summarize information.

Various works have tested the efficacy of CNNs, especially as feature extractors on word embeddings [78]. Recent works have also revitalized the interest in CNNs in NLP by introducing Temporal CNNs. In short, these aim to extend CNNs by allowing them to capture high-level temporal information [132, 248].

2.1.3.3 Transformers and the attention mechanism

As previously mentioned, one of the most influential neural architectures introduced in recent years, especially for text processing, is the Transformer architecture [128]. The foundational framework is that of an encoder-decoder with a variable number of encoder and decoder blocks. The main innovation of the Transformer is its departure from the recurrent connections that defined earlier sequence-to-sequence models. By eliminating recurrence, which had represented a significant bottleneck for parallelization and scaling, the Transformer models dependencies between words entirely through an *attention mechanism* [92, 89].

Attention is a weighting strategy, devised to learn how different components contribute to a result. It was initially proposed in the machine translation domain [92] as an alignment mechanism that matched each word of the output (translated sentence) to the respective words in the input sequence (original sentence). The

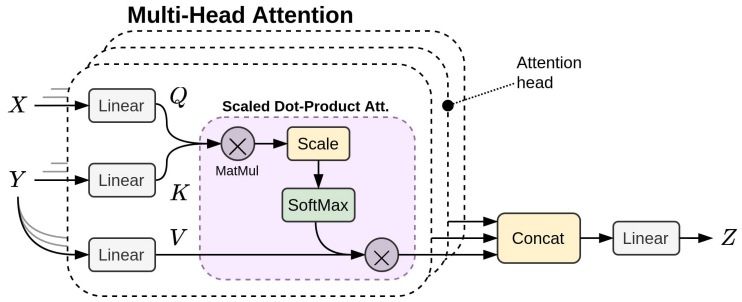


Figure 2.4: The multi-head attention layer used in the Transformer architecture.

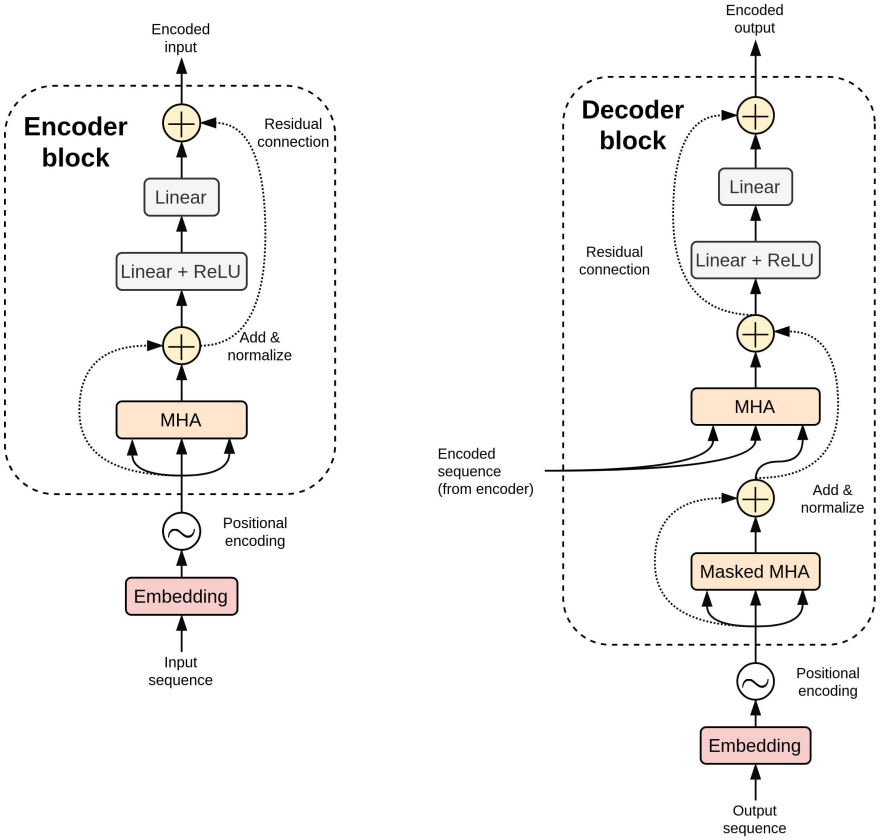
rationale behind this is that, when translating a sentence, a good translation can only be obtained by looking at the context of words and paying attention to specific words.

Vaswani et al. [128] used this mechanism in the Transformer architecture to allow the model to process all input tokens simultaneously, rather than sequentially, as was the case in previous recurrent networks. Input sequences are fed to the Transformer encoder at the same time, and a *positional encoding* scheme is used in the first layers of the encoder and decoder to inject some ordering information in the word embeddings. This ensures word ordering properties are not lost, e.g., that two words appearing in different positions in a sentence will have a different representation. Then, in all the remaining layers, Transformers use self-attention layers to learn dependencies between tokens. The layers are *self-attentive* as each token pays attention to every other token in a sentence, which is the main learning mechanism that allows for the disposal of recurrence. Indeed, since tokens in the input sequence are being processed simultaneously, the encoders can look at surrounding tokens in the same sentence and produce context-dependent token representations [2].

Stacking several self-attention layers produces a Multi-Head Attention (MHA) layer, whose structure is shown in Fig 2.4. Vaswani et al. [128] argue that having multiple attention heads in the layer enables the model to pay attention to different information in distinct feature spaces, and at different positions. Input sequences in the attention heads are transformed using linear transformations to generate three different representations, which the authors name Q (queries), K (keys), and V (values), following the naming convention used in information retrieval (as in Eq (2.4)):

$$\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}_{\boldsymbol{Q}},\ \boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_{\boldsymbol{K}},\ \boldsymbol{V} = \boldsymbol{W}_{\boldsymbol{V}} \tag{2.4}$$

Where $\boldsymbol{W}_{\boldsymbol{Q}}, \boldsymbol{W}_{\boldsymbol{K}} \in \mathbb{R}^{dim \times d_k}, \boldsymbol{W}_{\boldsymbol{V}} \in \mathbb{R}^{dim \times d_v}$ are the learned weight matrices. In the Transformer architecture, $\boldsymbol{K}$ and $\boldsymbol{V}$ are always generated from the same sequence (as we will discuss, $\boldsymbol{Q}$ is used differently in the encoder and decoder parts). The transformed sequences are then used to compute the scaled dot-product attention,



(a) Encoder block. (b) Decoder block.

Figure 2.5: Transformer encoder-decoder architecture.

as follows:

$$\boldsymbol{Z_k} = \text{ScaledDotAttention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{2.5}$$

While many definitions of attention exist in the literature, the authors decided to use the scaled dot-product mainly for efficiency reasons. The product of the key and query matrices is scaled by $\sqrt{d_k}$ to improve the stability of the gradient computation. At an intuitive level, we may envision a word $q \in \boldsymbol{Q}$ as being queried for similarity/relatedness against all keys $k \in K$, finally obtaining the relevant word representation by multiplying by V.

The final result of the MHA layer is the concatenation of all heads multiplied by matrix $\boldsymbol{W_O} \in \mathbb{R}^{hd_v \times dim}$ that reduces the output to the desired dimension $\mathbb{R}^{N \times dim}$:

$$\boldsymbol{Z} = \text{Concat}(\boldsymbol{Z_k})\boldsymbol{W_O} \tag{2.6}$$

In encoder blocks, $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ are all generated by the input sequence and have the same size; the general structure of an encoder block can be seen in Fig 2.5a.

In decoder blocks, $\boldsymbol{Q}$ comes from the previous decoder layer, while $\boldsymbol{K}$ and $\boldsymbol{V}$ come from the output of the associated encoder. These blocks structurally differ from encoders in the presence of another MHA layer, which precedes the standard one and is introduced to mask future tokens in a sentence (Fig 2.5b). This allows the decoder to be "autoregressive" during training, as, otherwise, it would trivially look forward in a sentence to obtain the result.

With the widespread usage of Transformer blocks with MHA, researchers have studied the features extracted at each layer. Interestingly, some of these works found that each encoding block may focus on the extraction of linguistic features at different levels: syntactic features are mostly extracted in the first blocks, while deeper layers progressively focus on semantic features [275, 196]. This "layer specialization" phenomenon suggests that stacking attention layers creates expressive representations that blend morphological and grammatical features.

The attention mechanism served as the basis of various derived attention schemes. For example, in the TC domain, Hierarchical Attention Networks (HANs), proposed by Yang et al. [98], have been widely used [309, 218, 415]. Their idea is to apply attention hierarchically at different levels of granularity. In many applications, the mechanism is initially applied to words and used to produce more informative sentence representations, and then it is also applied to sentences to produce better document representations. This method adapts particularly well to the processing of long texts since it leverages the hierarchical nature of human-produced written documents, which are often structured hierarchically.

2.1.3.4 Graph neural networks (GNNs)

The ubiquity of graph structures in most domains has sparked much interest in the application of neural networks directly to graph representations. In the NLP domain, a body of text can be represented as a graph of words, where connections represent relations that are potentially semantic or grammatical. As a simple example, a sentence could be represented by word nodes, and adjacent words in the sentence would be linked by an edge. Entire documents can also be linked together in a graph, for instance, in citation networks, or simply considering their relatedness [296, 2]. In hierarchical TC, graphs of labels have often been used to propagate hierarchy information between connected labels, with connections usually representing parent-child relations.

Message passing The principle behind graph processing models is generalized by the Message Passing Neural Network [117]. In this model, a *message passing phase*, lasting T time steps, is used to update nodes and edges representations by propagating information along edges. First, for a graph $G = (V, E)$, the message at time $t+1$ is computed for each node $u \in V$, depending on the previous values of the

nodes and edges $e \in E$. Therefore, for a node u:

$$m_u^{t+1} = \sum_{v \in \mathcal{N}(u)} \phi(x_u^t, x_v^t, e_{uv}^t) \tag{2.7}$$

where ϕ is a function learned by a neural network, function $\mathcal{N}(u)$ gives the neighbor nodes of u, and $x \in X$ are node embeddings (representations). Eq (2.7) uses the sum operation to aggregate the multiple messages coming from the neighbors of u, although it could be replaced with other permutation-invariant operations, such as the average or minimum. Once the messages have been computed, the node embeddings are updated using an update function σ, which is also learned by a neural network:

$$x_u^{t+1} = \sigma(x_u^t, m_u^{t+1}) \tag{2.8}$$

Analogously, Equations (2.7) and (2.8) could be adapted to compute the message starting from neighbor edges instead of nodes and to update the edge representation accordingly. Finally, in the *readout phase*, values from all nodes are aggregated by summing them together, and the output is used for a graph-level classification task.

Graph convolution The concept of message passing lends itself to the definition of a *convolution operation* for graphs that can be used within Graph Convolutional Neural Networks (GCNs) [187]. In the literature, two main categories of GCNs are typically distinguished: spatial-based and spectral-based. Similar to the conventional convolution operation over an image, spatial-based GCNs define graph convolutions based on the graph topology, while spectral-based methods are based on the graph's spectral representation [296, 75, 342]. We will only discuss the former type, which is more closely related to the message-passing concept.

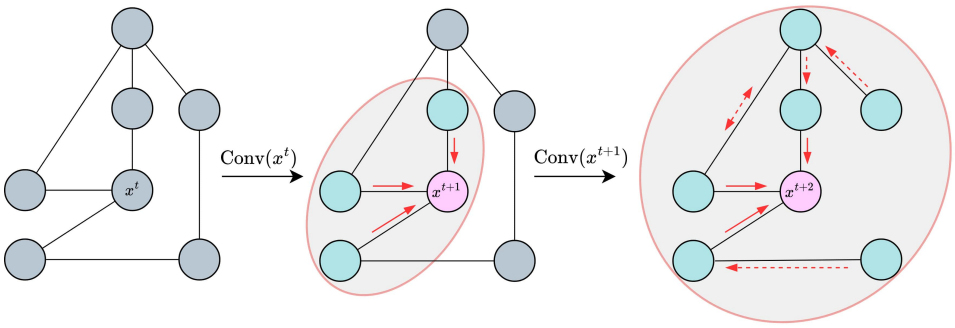


Figure 2.6: The propagation effect operated by two sequential graph convolutions on node x. The red arrows show the information flow toward the target node.

While many different definitions of convolution have been proposed, a simple spatial convolution operator on a graph can be defined as:

$$\text{Conv}(x_u^t) = \sigma\left(\bigodot_{v \in \mathcal{N}(u)} \phi(x_u^t, x_v^t)\right) \tag{2.9}$$

in which $\odot$ is a permutation invariant operation. The result of this operation can be used to update each node representation, as exemplified in Fig 2.6. When $\odot = \sum$ and $\phi(x_u^t, x_v^t) = x_v^t \mathbf{W}^T + b$, which is a simple linear transformation of the representation of neighbor nodes, this operation can be defined in matrix form as:

$$\text{Conv}(\boldsymbol{X}) = \sigma(\boldsymbol{A}(\boldsymbol{X}\boldsymbol{W}^T + \boldsymbol{b})) \tag{2.10}$$

In the equation above, $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is the adjacency matrix, $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ contain the nodes' representation of dimensionality d, and $\boldsymbol{W} \in \mathbb{R}^{d_{out} \times d}$ and $b \in \mathbb{R}^{d_{out}}$ are the weight matrix and bias term for one layer, respectively. The multiplication by the adjacency matrix $\boldsymbol{A}$ guarantees that only neighbor nodes contribute to the updated representation $\boldsymbol{X}$. Consequently, if multiple convolutions are stacked, the message from each node can propagate further in the graph. However, if too many layers are used, large portions of the graph could end up having similar nodes' representations, an issue often referred to as *oversmoothing* [141].

As a natural extension of the operation defined in Eq (2.10), $\boldsymbol{A}$ can be weighted to reflect the importance of neighbors, for instance by multiplying each entry with edge weights. The attention mechanism can also be used to autonomously learn how much each node should contribute to its neighbors' representations. An example of this is Graph Attention Networks (GATs) [177], which use the Transformer's MHA to compute the hidden states of each node.

2.1.4 Text Classification

Having detailed the foundational stages of preprocessing, text representation, and modern neural architectures, we now arrive at the final step of the pipeline: the classification algorithm. The approach to classification itself has evolved in a progression of paradigms, each changing how models learn from data and generalize to new tasks. Drawing from the comprehensive survey by Zhao et al. [460], we can characterize this progression in three major stages: (i) an early stage of statistical and neural modeling, which separated the feature representation step from classification; (ii) a transfer learning stage, which introduced the pre-training and fine-tuning paradigm to create specialized task solvers; and (iii) the current stage of generalized solvers, where massive LLMs address a wide array of tasks through prompting and in-context learning. The following sections will detail the classification approach within each of these paradigms.

2.1.4.1 Statistical and neural models

The first paradigm for TC treated feature representation and classification as two distinct, sequential steps. This approach encompasses both early statistical LMs and the first generation of neural language models built on top of static word embeddings.

Statistical LMs rose to prominence in the 1990s and relied on statistical learning methods to build classifiers. After creating a task-specific feature representation, such as BoW or TF-IDF vectors, from the preprocessed text, these sparse, high-dimensional vectors were fed into a separate machine learning (ML) classifier, such as Naïve Bayes, Support Vector Machines, or Logistic Regression [189]. While still useful in limited computational resources scenarios, these methods often suffer from the "curse of dimensionality" especially in relation to the data sparsity issue that makes it difficult to accurately estimate model parameters.

As discussed, the progress in neural language modeling around 2013 marked a significant leap forward by introducing dense, low-dimensional word embeddings learned by shallow neural networks like Word2Vec using language modeling training objectives. The approach to feature engineering (or text representation) shifted from a manual one toward learning task-agnostic representations that could capture semantic relationships between words. However, the core classification pipeline remained largely the same: these pre-trained, static embeddings were first used to generate a document-level representation (e.g., by averaging the word vectors), which was then used as input for a separate classification model. Although this approach reduced the burden of feature engineering and improved performance by incorporating limited semantic information, the representation and classification steps were still decoupled [189].

2.1.4.2 Transfer learning for specialized task solvers

The second major paradigm shift was driven by the emergence of deep, contextualized LMs, also known as pre-trained LMs. The pre-training and fine-tuning paradigm transformed TC from a two-step process into an integrated, end-to-end trainable system. This leap was enabled by the Transformer architecture, the foundational technology for most modern LMs, that we described in § 2.1.3.3.

Models like BERT (Bidirectional Encoder Representations from Transformers) [192] and the early GPT (Generative Pre-trained Transformer) series [174] are first pre-trained on massive, unlabeled text corpora using self-supervised objectives, such as MLM (see § 2.1.2). The pre-training phase allows the model to develop a deep, context-aware, and general understanding of language, far surpassing the capabilities of static word embeddings.

For a downstream task like TC, the pre-trained LM is then adapted through fine-tuning. This involves adding a simple, task-specific classification layer (or "head") on top of the LM's architecture. The entire system (both the pre-trained body and the new head) is then trained on a smaller, labeled dataset specific to the classification

task. During this process, the model's general linguistic knowledge is specialized to the nuances of the target task. Other strategies include fine-tuning only the task-specific head or partially fine-tuning only specific layers of the pre-trained component. Both these strategies are less computationally expensive and produce excellent performance on many tasks [2, 189]. This transfer learning approach proved to be exceptionally powerful, establishing new state-of-the-art results across a wide range of NLP benchmarks and creating highly effective, specialized task solvers. This approach is widely used across various NLP tasks with different architectures, like BERT [192] and generative ones [292, 260]. Adapting them to various tasks is done by just introducing a task-specific head and fine-tuning it.

2.1.4.3 Generalized solvers

The most recent paradigm is characterized by the scaling of pre-trained LMs to enormous sizes, producing what are now known as LLMs. This scaling did not just lead to quantitative improvements; it unlocked qualitatively different behaviors and *emergent abilities* that were not present in smaller LMs. One of the most significant of these abilities is *in-context learning* (ICL), which has fundamentally altered the approach to many tasks, including TC. It is important to note that this last stage is not a new technical concept, but is the consequence of progressive advances in the field that shifted the focus from the modeling of language to complex task solving.

Emergent abilities Emergent abilities are formally defined as capabilities that are not present in smaller models but appear once a model's scale reaches a certain threshold, often marked by a significant jump in performance. Three such abilities are central to the function of LLMs as generalized solvers [460]:

- **In-Context Learning (ICL)**: This is the ability to perform a task using a natural language instruction and/or a few task demonstrations provided within a prompt. The model generates the expected output without needing additional training or gradient updates. This ability was strongly present in the 175B-parameter GPT-3 model but not in its smaller predecessors, GPT-1 and GPT-2.
- **Instruction Following**: Through a process called instruction tuning, LLMs learn to perform well on unseen tasks described via natural language instructions, giving them powerful generalization without needing explicit examples for every new task. According to the experiments in [373, 347], this ability began to appear in LaMDA-PT and PALM [381] models with at least around 68 billion parameters, but was absent in smaller versions.
- **Step-by-Step Reasoning**: Unlike smaller LMs, which struggle with complex, multi-step problems, LLMs can solve such tasks by generating intermediate reasoning steps before providing a final answer. This is often elicited using

a technique called Chain-of-Thought prompting. This ability brings significant performance gains when applied to models with more than 100 billion parameters.

The mechanisms that enable these emergent abilities are still an active area of research. Current theories suggest that LLMs may perform a form of implicit learning during forward computation, akin to meta-optimization or Bayesian inference, where demonstrations guide the model to select and apply knowledge it has already learned during its pre-training phase [460].

To put this into perspective with the previous task-specific fine-tuning paradigm, the scale that enables these abilities is, at a minimum, 200 times larger than that of the largest original BERT model. While these LLMs can also be fine-tuned, it is often advisable to first benchmark their performance using prompting strategies before committing to this computationally expensive step.

Classification via prompting With LLMs, the goal is no longer to create numerous specialized models via fine-tuning. Instead, a single, massive, "general-purpose" task solver can be guided to perform a wide variety of tasks through carefully crafted prompts, without updating its parameters. This interaction method, known as *prompting*, leverages the emergent abilities of ICL and instruction following. For TC, a prompt is formatted to include a task description and, optionally, a few examples (demonstrations). The LLM then uses this context to classify a new, unseen piece of text. For example, a sentiment analysis task can be formulated with the following prompt:

```
Instructions: Given a movie review, classify the sentiment as either
Positive, Negative, or Neutral.

Example Review: "I loved this film, the acting was superb!"
Sentiment: Positive

Example Review: "The plot was slow and completely uninteresting."
Sentiment: Negative

Review: "The movie was okay, but not very memorable."
Sentiment:
```

The model is expected to complete the final line by generating the correct label, "Neutral", based on the pattern established by the demonstrations. This approach eliminates the need for large, task-specific datasets and the computational cost of fine-tuning, while providing unprecedented flexibility.

However, prompting is a delicate process. LLMs are highly sensitive to how a task is framed and which examples are provided. Poorly designed prompts can bias

a model towards incorrect answers, and seemingly minor changes can sometimes degrade performance significantly [440]. For instance, models can exhibit a recency bias, tending to repeat answers that appear near the end of the demonstrations [460]. The science of designing and refining these inputs is widely known as *prompt engineering*. Common prompting strategies include:

- **Few-shot prompting**, as shown in the example above, provides a small number of demonstrations to help the model learn the semantic mapping between input and output for a specific task.
- **Zero-shot prompting**, in contrast, provides only a task description without any examples, relying entirely on the model's pre-existing ability to follow instructions.
- **Chain-of-Thought (CoT) prompting** is an advanced technique used to elicit the model's step-by-step reasoning ability. By including intermediate reasoning steps in the demonstrations, especially for mathematical tasks, the model is guided to produce a more grounded and often more accurate answer. A *zero-shot* version of CoT simply adds a phrase like "Let's think step by step" to the prompt, which can trigger the model's reasoning process without requiring manual examples [385, 370].

2.1.4.4 Challenges and recent developments

LLMs have substantially influenced recent approaches in computational NLP. Pre-trained on diverse, multilingual, and often multimodal datasets (i.e., containing various types of data, such as code and images), these models have altered the approach to language processing. Their scale has led to remarkable improvements in cross-lingual transfer, with models demonstrating strong capabilities across many languages, even those underrepresented in the training data [405, 453].

A recent major development addresses the prohibitive cost of adapting these massive models. Fully fine-tuning a foundational model is computationally intensive, making frequent updates or domain specializations impractical. This challenge is now being addressed by Parameter-Efficient Fine-Tuning (PEFT) techniques, such as LoRA, QLoRA, and various adapter modules [463]. PEFT methods enable the specialization of pre-trained models by updating only a small fraction of their parameters, significantly reducing computational demands and mitigating the risk of catastrophic forgetting or overfitting, especially on smaller datasets. While much of the focus has been on generative models, efficiency improvements are not exclusive to them. Encoder-only architectures continue to be refined for specific tasks, with newer proposals like ModernBERT driving further progress [471].

Despite these advancements, achieving equitable performance across all languages remains a primary challenge, particularly for low-resource languages. Key issues persist, including severe data scarcity, the risk of propagating biases from

imbalanced training data, and inefficient tokenization schemes for non-English languages that amplify computational costs [452, 453]. Furthermore, general-purpose LLMs can underperform specialized models on domain-specific tasks in low-resource languages and are more prone to generating harmful content due to inadequate safety alignment in those languages [437, 435]. Addressing these multifaceted challenges to build a more language-fair AI will require extensive research focused on sustainable language representation and culturally sensitive evaluation benchmarks to mitigate undesirable biases.

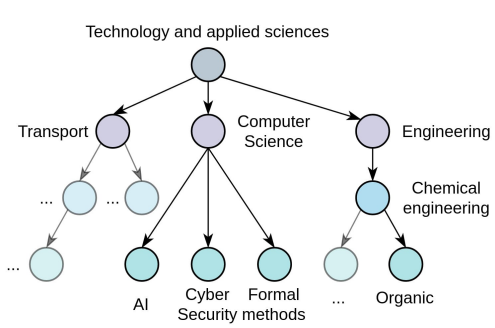
2.2 Hierarchical Text Classification

As a sub-task of the broader TC area, HTC methods have much in common with standard text classifiers. This section will outline what differentiates HTC from standard approaches, along with the main considerations for designing and evaluating these methods in the presence of a taxonomy of labels.

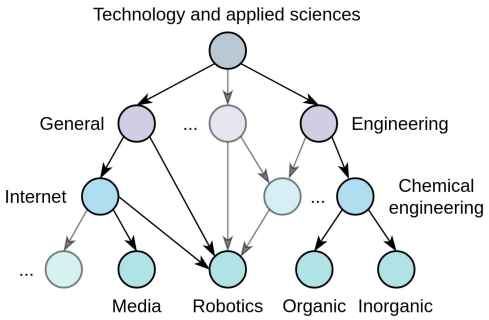
2.2.1 Types of Hierarchical Classification

Standard classifiers focus on what HTC literature defines as *flat classification*. In it, categories are treated in isolation (i.e., as having no relationship between one another) [29, 31]. In contrast, HTC deals with documents whose labels are organized in structures that resemble a tree or a directed acyclic graph (DAG) [358, 173]. In these structures, each node contains a label to be assigned, such as in Fig 2.7. Methods able to work with both trees and DAGs can be devised, though a simpler technique is to simply "unroll" or "flatten" sub-nodes with multiple parents for DAGs, thus obtaining a tree-like representation. For this reason, this section (and much of HTC literature) focuses on hierarchies with a tree structure.

HTC approaches can be divided into two groups: *local* and *global* approaches [390]. Local approaches (sometimes called "top-down") are defined as such because they "dissect" the hierarchy, constructing multiple local classifiers that work with a subset of the node labels. While more specialized than flat classifiers — which ignore the hierarchy — there is an inevitable loss of hierarchical information, as the aggregation of these classifiers tends to ignore the holistic structural information of the taxonomic hierarchy. Depending on the chosen approach, the amount of information regarding the hierarchy can be partial or absent [392]. Local approaches (Figures 2.8b to 2.8d) have been criticized because of their structural issues, the most notable one being that they may easily propagate misclassifications [48, 62]. Furthermore, these models are often large in terms of trainable parameters, and may easily develop exposure bias because of the lack of holistic structural information [297]. This arises from the fact that, at test time, lower-level classifiers use the prediction of previous classifiers, thus leading to a discrepancy between the training (which uses ground truth) and the inference phase.



(a) Label hierarchy based on a tree structure.



(b) Label hierarchy based on a DAG structure.

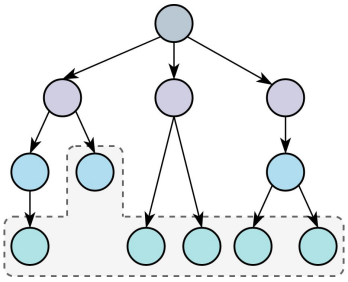
Figure 2.7: Hierarchically structured labels. Both forms are frequent in practice, though labels organized in a DAG require adaptation of either the method or the structure.

Global methods aim to solve these shortcomings, as well as being frequently less computationally expensive (as there is a single classifier) [57]. While the definition of global classifiers is deliberately generic, one might imagine as global any individual classification algorithm that directly takes into account the hierarchy. It is also worth noting that global and local approaches can also be combined [152, 225]. Fig 2.8 showcases the main approaches to HTC, which we now briefly summarize.

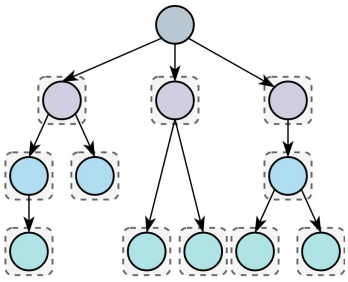
Flattened classifiers The flat classification approach (Fig 2.8a) reduces the task to a multiclass (or multilabel) classification problem, therefore discarding hierarchical information entirely. Typically, only leaf nodes are considered, and the classification of higher levels of the hierarchy is inherited from parent nodes [392, 214].

Local classifiers Local classification approaches are generally divided into three categories, which differ in how they dissect the hierarchy. First off, the class of *local per-node (binary)* approaches (Fig 2.8b) considers each label as a separate class, disregarding the hierarchy completely. This approach is among the simplest and resembles a one-versus-rest approach [26]. *Local per-level* methods (Fig 2.8c) assign a classifier to each relevant category level, which is tasked to decide for that level alone. Finally, *local per-parent* methods (Fig 2.8d) assign a classifier to all parent nodes, tasking them to assign a sample to one of its children, therefore capturing part of the sample's current path through the hierarchy.

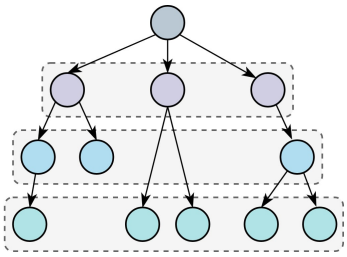
Global classifiers A global (or *big-bang*) [217] classification approach (Fig 2.8e) makes use of the entire hierarchy to make the final classification decision. It is common (though not strictly necessary) for global classifiers to make this decision on the flattened set of labels; hierarchy information is therefore achieved through



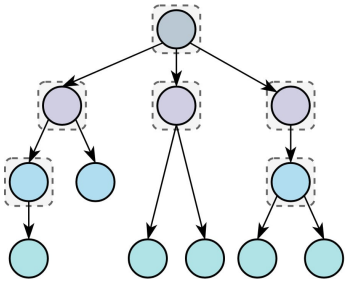
(a) Flattened.



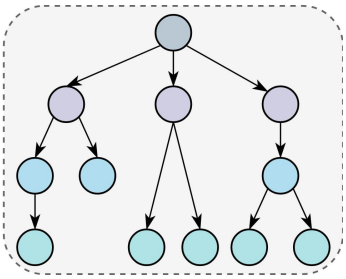
(b) Local binary.



(c) Local per-level.



(d) Local per-parent.



(e) Global.

Figure 2.8: The most common approaches to HTC, exemplified on a tree-like hierarchy. Flattened classifiers (Fig 2.8a) disregard the hierarchy, while local classifiers (Figures 2.8b to 2.8d) incorporate some of its structure. Global classifiers (Fig 2.8e) are designed to exploit the label structure within a single model. Dashed boxes indicate the presence of a classifier.

structural bias (i.e., the architecture of the model) [57].

2.2.2 Non-Mandatory Leaf Node Prediction and Blocking

In hierarchical classification datasets, it may not always be the case that all prediction targets correspond to leaves. Many authors distinguish between *mandatory* and *non-mandatory leaf node prediction* (MLNP, NMLNP) [29, 42]. Quite simply, in NMLNP scenarios, the classification method should be able to consider stopping the classification at any level of the hierarchy, regardless of whether it is a leaf node or not. The term applies to both Tree- and DAG-structured hierarchies. In this section, we briefly outline how the different types of HTC approaches can deal with the latter, more complex case.

Flattened classifiers The flat classification approach simplifies the problem to a standard, non-hierarchical classification problem. In that sense, if the target is restricted to the leaf nodes of the structure, methods that follow this paradigm are unable to deal with NMLNP by design [57]. It is possible to naively extend the classification targets to include all possible labels, though this would make the task

much harder since flattened classifiers do not have any inherent information about the hierarchy.

Local classifiers To deal with NMLNP, local approaches must implement a *blocking mechanism*, so that inference may be stopped at any level of the hierarchy. A simple way to do this is by utilizing a threshold on the confidence of each classifier; if, during top-down prediction, the confidence doesn't meet the requirement, the inference process is halted [43].

An issue with such thresholds is that they may lead to incorrect early stopping of classification. Sun et al. [36] define this as the *blocking problem*, which refers to any case in which a low confidence rating of a parent classifier mistakenly hints that an example does not belong to its actual macro-class. As a consequence, the example will never reach the classifier for its appropriate subclass. The authors propose some blocking reduction methods, which generally act by reducing the thresholds of inner classifiers or by allowing lower-level classifiers to have a second look at rejected examples.

Global classifiers Global classifiers utilize a single, usually complex algorithm that integrates the hierarchy into its internal reasoning. As mentioned, it is common to base global classifiers on an existing flat classification approach and modify it to take into account the class hierarchy. Global classifiers can also be used in NMLNP scenarios, though specific strategies might be needed in this case to enforce class-membership consistency in predictions. Likewise, it is also possible to integrate a top-down prediction approach (which would be internal to the algorithm) to avoid this issue.

2.2.3 Evaluation Measures for Classification

Because HTC tasks are inherently multiclass or multilabel, many researchers use standard classification metrics. However, several authors argue these measures are inappropriate [29, 40, 84, 229, 57]. Their main concern is that ignoring the hierarchy leads to imprecise evaluations, as standard metrics fail to account for the *mistake severity*: models that make "less severe" mistakes should be preferred [483]. This follows from two considerations. First, predicting a label that is structurally close to the ground truth should be penalized less than predicting a distant one. Second, errors at higher hierarchy levels are more harmful (e.g., misclassifying "football" as "rugby" is preferable to misclassifying the category "sport" with "food"). These considerations have been discussed in research on real-world HTC tasks such as ICD coding [401, 310] and legal document labeling [216]. A "better" mistake typically preserves most ancestor nodes in the prediction path, so the sample's macro categories remain correct; this is usually preferable to an error that alters the macro category, which can have severe consequences. Moreover, characterizing

error severity and type helps guide model development and suggests strategies to mitigate such errors.

This section outlines the most common evaluation metrics used in HTC, beginning with standard "flat" TC metrics and then discussing hierarchical-specific measures. For a more in-depth analysis of these metrics, their challenges, and potential solutions, see Kosmopoulos et al. [84].

2.2.3.1 Standard metrics

The most common performance measures utilized in "flat" classification are derived from the classic information retrieval notions of *Accuracy* (Acc), *Precision* (Pr), and *Recall* (Re). As in any other supervised learning task, we consider the truthfulness of a model's predictions against the *ground truth* derived from the dataset.

These metrics are calculated by considering each class individually. For any given class C, we can identify four outcomes for each sample in the dataset:

- **True Positive (TP)**: The sample's true label is C, and the model correctly predicts C.
- **False Positive (FP)**: The sample's true label is *not* C, but the model incorrectly predicts C.
- **True Negative (TN)**: The sample's true label is *not* C, and the model correctly predicts a label other than C.
- **False Negative (FN)**: The sample's true label is C, but the model incorrectly predicts a label other than C.

Accuracy measures the ratio of correct predictions over the total number of predictions. *Precision* is instead a measure of *correctness*, quantifying the proportion of correct predictions among the ones predicted as positive only, while *Recall* is a measure of *completeness*, quantifying the proportion of overall positives captured by the model. Accuracy, precision, and recall are defined as in Eq (2.11):

$$Acc = \frac{|TP| + |TN|}{|TP| + |TN| + |FP| + |FN|} \qquad Pr = \frac{|TP|}{|TP| + |FP|} \qquad Re = \frac{|TP|}{|TP| + |FN|} \tag{2.11}$$

Since precision and recall often present a trade-off, they are commonly combined into a single score. The most popular is the F-measure (F_β), which is the weighted harmonic mean of the two. Typically, β is set to 1, giving equal weight to both metrics, as shown in Eq (2.12):

$$F_\beta = \frac{(\beta^2 + 1) \cdot Pr \cdot Re}{\beta^2 \cdot Pr + Re} \qquad F_1 = 2 \cdot \frac{Pr \cdot Re}{Pr + Re} \tag{2.12}$$

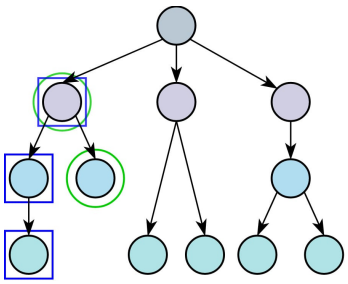


Figure 2.9: Tree hierarchy with predicted (squares) and ground truth (circles) labels, with the ancestors of each highlighted. As the two sets have a node in common, the misprediction should be considered less severe.

In multiclass settings, precision and recall can be averaged across all classes. *Macro-averaging* (Eq (2.13)) gives equal weight to each class, while *micro-averaging* (Eq (2.14)) gives equal weight to each instance, making it more suitable for datasets with class imbalance, as in Eq (2.14):

$$Pr_{macro} = \frac{\sum_{i=1}^{m} Pr_i}{m} \qquad Re_{macro} = \frac{\sum_{i=1}^{m} Re_i}{m} \tag{2.13}$$

$$Pr_{micro} = \frac{\sum_{i=1}^{m} |TP_i|}{\sum_{i=1}^{m} |TP_i| + |FP_i|} \qquad Re_{micro} = \frac{\sum_{i=1}^{m} |TP_i|}{\sum_{i=1}^{m} |TP_i| + |FN_i|} \tag{2.14}$$

where m is the number of categories. F-measures may be micro- or macro-averaged as well, utilizing the corresponding averaged versions of precision and recall.

2.2.3.2 Hierarchical metrics

While these standard metrics are widely used in TC, they cannot account for the relationships between classes. Sun and Lim [29] propose to solve this issue by introducing hierarchical metrics based on category *similarity* and *distance*. Category similarity evaluates the cosine distance of the feature vectors representing predicted and true categories, while category distance considers the number of links between the two in the hierarchy structure. While interesting, these measures have practical issues, which Kiritchenko et al. [40] outline (such as inapplicability to DAGs and multilabel tasks). Instead, they propose to extend traditional precision and recall. To do this, they augment the set of predicted and true labels to include all their ancestors (Fig 2.9) and calculate the metrics on the augmented sets.

Formally, let $\hat{Y}_{aug}$ be the augmented set of predictions, containing the most specific predicted classes and all of their ancestors (set denoted as L), and Y_{aug} the augmented set of the most specific ground truth class(es) and all of their ancestors encoded as binary vectors $\{0,1\}^{|L|}$. Then, hierarchical metrics (h-metrics) can be defined as follows (Eq (2.15)):

$$hPr = \frac{\sum_i |\hat{Y}_{aug} \cap Y_{aug}|}{\sum_i |\hat{Y}_{aug}|} \qquad hRe = \frac{\sum_i |\hat{Y}_{aug} \cap Y_{aug}|}{\sum_i |Y_{aug}|} \qquad hF_\beta = \frac{(\beta^2+1) \cdot hPr \cdot hRe}{\beta^2 \cdot hPr + hRe} \tag{2.15}$$

However, Kosmopoulos et al. [84] observe that this approach over-penalizes errors in which nodes have many ancestors. Specifically, false positives predicted at lower levels of the hierarchy strongly decrease the precision metric, while recall tends to be boosted since adding ancestors is likely to increase the number of true positives (false positives are ignored). Drawing from the Lowest Common Ancestor (LCA) concept defined in graph theory [14], they propose LCA-based measures as a solution. Briefly, and in the context of tree structures, the LCA of two nodes can be defined as the lowest (furthest from the root) node that is an ancestor of both. Therefore, these new measures are defined similarly to the one in Eq (2.15), but with the expanded sets defined in terms of LCA rather than on full node ancestries [229]. Despite these issues, the hierarchical metrics of Eq (2.15) are still considered effective measures across a broad range of scenarios [57].

Some authors report metrics that explicitly evaluate the difference, in terms of path distances through the hierarchy, between class predictions and ground truth labels. In particular, Sainte Fare Garnot and Landrieu [337] define the Average Hierarchical Cost (AHC) of a set of predictions. Given two labels $a, b \in L$, define dist(a,b) as the number of edges that separate two labels in the hierarchy tree. In turn, for a set of labels S, let dist(a,S) be the minimum distance between the node and the set, i.e., $min(dist(a,s)\ \forall s \in S)$. If predictions and ground truth labels are expressed as vectors $\in \{0,1\}^{|L|}$ (as before), the AHC can be defined as Eq (2.16):

$$AHC(Y, \hat{Y}) = \frac{1}{|Y|} \sum_j \hat{\mathbf{y}}_j \ \text{dist}(j, Y) \tag{2.16}$$

In practice, this measure evaluates how "far off" the predictions were from the closest common ancestor (and, in that, are not too dissimilar for LCA-based measures). As a consequence, a low AHC indicates that mistakes were, on average, not too distant from the ground truth.

2.2.3.3 Other metrics

Depending on the particular aspect of HMC being tackled, other types of metrics, often borrowed from neighboring disciplines, may be warranted [83]. In this section, we briefly introduce them.

Some works report rank-based evaluation metrics, which are widely used in the more general context of multilabel classification [238, 285]. Among many, the two most commonly utilized are Precision and Normalized Discounted Cumulative Gain at k ($Pr@k$, $NDCG@k$), which are often utilized in retrieval and recommendation systems [3]. Briefly, given a model's prediction (i.e., a probability vector across labels),

we can sort the labels in descending order based on the output probabilities. Then, $Pr@k$ indicates the fraction of correct predictions in the top-k labels of this sorted list, whereas $NDCG@k$ measures ranking quality (i.e., how high correct labels are ranked).

In methods able to categorize labels at different levels of the hierarchy separately, some authors chose to showcase the accuracy score at each separate level, as well as a single overall score [290, 341]. The overall score is the one obtained by classifying the last level of the hierarchy given the (possibly incorrect) predictions of the parent classes. In some cases, authors utilize *subset* accuracy, where all labels must match the ground truth exactly.

Lastly, some authors utilize the *Hamming loss* [344, 83] metric for HTC. This measure evaluates the fraction of misclassified instances, i.e., true labels that were not predicted or predictions that were not in the ground truth.

2.3 Future Directions

To conclude this chapter, we outline some of the latest advancements and promising directions in TC and HTC.

A growing body of literature is applying LLMs to classification tasks, exploiting their discriminative capabilities as generalized solvers, as detailed in § 2.1.4.3. Specifically, the few-shot prompting paradigm allows for effective problem modeling with significantly fewer annotations than required by traditional fine-tuning. This offers benefits for HTC as well; these models can embed flexible, class-specific information and, through adaptive prompting, accommodate diverse hierarchical structures with minimal engineering effort. Recent contributions have focused on enhancing knowledge representation and optimizing loss functions [459, 483, 345], while others explicitly target the few-shot capabilities of generative models in hierarchical settings [462, 400, 396].

However, the deployment of massive generative LMs incurs substantial computational costs. While efficient training paradigms offer one solution (as discussed in § 2.1.4.3), a complementary and promising path for future development is the investigation of generative capabilities within more compact architectures, often referred to as "BabyLMs". Scaling down generation to these smaller models offers the potential to improve data efficiency and develop techniques that can benefit larger models as well [473].

Finally, we highlight the importance of designing more comprehensive benchmarks for model evaluation. Recently, most models are benchmarked against a few popular ones (such as GPQA [450] and MMLU [298]), but these results should be interpreted with caution. The metrics often assess narrow capabilities, and incremental improvements on general evaluations do not necessarily imply an increase in reasoning ability. As we observed in Chapter 8, even models that perform well on public leaderboards can struggle when confronted with specific domain tasks. One

potential cause for this discrepancy is the extensive post-training process generative LMs undergo, which fine-tunes them for user alignment and moderation. While this process has merits for general safety, it has been theorized to dull certain linguistic abilities required in specific contexts. For instance, Quan et al. [469] and Mirowski et al. [428] note how standard safety alignment impacts the ability to generate humorous content. Humor often requires playing with words and meanings in ways that generic filters might flag as unsafe or offensive, yet such nuances remain valid for specific audiences (e.g., comedians or linguists). To mitigate this, the authors argue for shifting from general post-training procedures toward audience-specific and language-specific alignment [435]. This approach would ensure models are better tuned to recognize distinct contextual constraints, particularly regarding fairness, safety, and linguistic requirements.

3

Principles of Explainable AI

The rapid proliferation of AI and ML has led to the development of models with unprecedented predictive power. Deep neural networks, in particular, have achieved state-of-the-art performance across a vast range of domains, from computer vision to NLP. However, this performance often comes at the cost of transparency. The most powerful models operate as "black boxes", where the internal logic and decision-making processes are opaque to human users [299]. This trade-off between performance and transparency presents significant challenges, especially in high-stakes applications such as medical diagnosis, financial credit scoring, and autonomous systems, where understanding the rationale behind a decision is not merely desirable but often a legal, ethical, and practical necessity [299, 249]. The European Union's AI Act, for instance, places stringent constraints on "high-risk" systems, defined as those that "may produce significant, adverse effects to an individual's health, safety, or fundamental rights". For these applications, the Act mandates a *right to explanation*, compelling developers to provide "clear and meaningful explanations of the role of the AI system in the decision-making procedure and the main elements of the decision taken" [429].

In response to this challenge, the field of Explainable AI (XAI) is dedicated to developing methods that can render the decisions of complex ML models understandable to humans, helping to build systems that are accurate, trustworthy, and robust [249, 339].

3.1 Motivations of XAI

Aside from legal compliance, the need for XAI stems from a fundamental issue of *trust*. Statistical machines trained only on experience offer no inherent guarantee that they will operate reasonably well in all real-world scenarios, especially in the presence of unforeseen conditions that were not extensively modeled in the training data. In such cases, XAI can provide a set of tools that enable expert operators to validate and supervise the AI's work, as well as to create criteria for accountability. Moreover, when used as support tools for human decision-makers, AI can influence their decisions, and research has proven the benefits of pairing ML models with XAI for more impartial AI-assisted decision-making [362, 421].

For these reasons, we argue that explainability should be considered a non-functional requirement in the software design process, not an afterthought [278]. Even the most accurate black-box model can be unreliable if we cannot be sure whether it has learned to discriminate based on meaningful features or on spurious correlations. A classic example, shown by Ribeiro et al. [100], is an image classifier that correctly distinguishes wolves from huskies only because it has learned to associate them with snowy backgrounds, rather than the features of the animals themselves. Their work demonstrates the role of XAI as a diagnostic tool, showing how such methods can uncover and diagnose problems rooted in the training data. This diagnostic capability is important because datasets are not neutral artifacts. They are constructed and often reflect historical inequalities and societal biases, which can lead ML models to perpetuate or amplify discrimination when used uncritically. Analogous to the previous example, it is not difficult to imagine systems that unfairly discriminate against people simply because data misrepresents an ethnicity, gender, or other social attribute, especially since these attributes can be encoded implicitly (i.e., unintentionally) in the dataset [408]. Although often left out of traditional interpretability discourse, which focuses more on the computational aspects of ML, data curation is a key part of building robust models, as model bias likely stems from biased data. XAI can help diagnose data issues, but correcting these also requires critically examining how data are gathered and the assumptions underlying their collection [408].

A useful summary of these motivations is provided by Barredo Arrieta et al. [249]. They identify three key reasons for pursuing interpretability. First, it helps ensure impartiality by detecting and enabling the correction of biases in the training data. Second, it facilitates robustness by highlighting potential adversarial perturbations that could produce unexpected outputs. Third, it helps guarantee causality by confirming that the model uses meaningful variables to make its decisions.

To these points, we add another motivation: knowledge discovery and extraction. In scientific domains, especially the life and physical sciences, the explanation process can be used to extract novel, human-understandable insights and hypotheses that a model has learned from complex data. While XAI methods are far from solving the "black box" problem entirely, their ability to even partially explain the function learned by a model can serve as a starting point for future investigations into the causal relationships behind certain phenomena [348]. For example, the use of DL in drug discovery has led to applications of XAI to support the "analysis and interpretation of increasingly more complex chemical data, as well as in the formulation of new pharmacological hypotheses, while avoiding human bias" [263].

Ultimately, the motivations for XAI are both practical and ethical, and can be understood within the broader objective of building *Responsible AI*, a concept that blends model explainability, fairness, and privacy by design [249]. This points to the need for a holistic approach that considers not just a model's predictions, but also its internal reasoning, the data underlying it, and the type of interaction with the end-user. While this overview helps contextualize the field within its broader goals,

the remainder of this chapter will focus specifically on the foundational concepts and terminology of XAI.

3.2 Core Concepts

The most central concept in XAI is *explainability* itself. It can be informally defined as a property related to the ability to "extract knowledge from a ML model about its reasoning and present it in a way that is both useful and accurate for a given objective". Although this concept has no single, universally accepted definition, it provides a useful starting point [205].

The following subsections will introduce key terminology, establish a formal framework for explanation, and provide a taxonomy of XAI methods.

3.2.1 Interpretability and Explainability

Within the XAI literature, the terms *interpretability* and *explainability* are often used interchangeably and can be a source of confusion. While a universal consensus has yet to emerge, there is growing agreement on certain elements that distinguish the two concepts [159, 205].

Interpretability is often described as the degree to which a human can understand the cause of a model's decision [184]. Some researchers frame this in terms of the input-output relationship: a system is considered interpretable if a user can intuitively grasp or predict the model's output for a given input, even if the internal mechanisms remain opaque [185, 299, 127]. For example, a user might find a loan approval model interpretable if they can predict that applicants with low credit scores will be denied, without needing to know precisely how the credit score is weighted against other variables. Other authors frame interpretability as a passive property of a model itself, related to its intrinsic simplicity [249, 156, 167]. A logistic regression model, for instance, has high intrinsic interpretability because its parameters can be directly understood (mathematically) by a human observer [159].

In contrast, *explainability* is best understood as the product of an active process designed to clarify a model's internal mechanics or rationale to end-users [299, 402]. In this view, an explanation is seen as a *user-centric* concept, specifically an interface between the end-user and the machine [184, 156, 185, 127]; it focuses on characterizing the process that produces an outcome, allowing end-users to understand not only *what* the model does, but also *how* it works. This distinction, however, is not universally adopted. Due to the semantic similarity of the words themselves, some researchers equate interpretability with explainability, arguing they serve the same goal [184, 205]. Others propose explainability as a stronger property that requires interpretability as a prerequisite, plus additional context related to the explanation goal, user expectations, and domain knowledge [294, 479].

Given this ambiguity, both terms can be seen as loosely referring to the "extraction of relevant knowledge from a ML model" [205], which aligns with the initial definition used in this chapter. What is generally agreed upon, however, is that explainability is an *active* property, achieved through a system or interface that conveys information to its users [299, 156, 127]. This is the perspective adopted in my research, where an *interpretation* is treated as a specific step within the broader, active process of generating an *explanation*. Accordingly, we favor the term *explainable* to describe a system that can explain (part of) its decisions. This framework, which refines the meaning of these terms for my work, is described in the next section.

A related concept is *transparency*, which can be defined as an intrinsic property of a model that enables explainability. A model is considered transparent if it possesses either algorithmic transparency (its decision process is mathematically traceable), simulatability (it is simple enough for a human to simulate mentally), or decomposability (each part of a model can be intuitively understood) [249, 159, 294]. A transparent model's internal mechanics are thus open to inspection. Conversely, while we can observe what a black box algorithm does (making it potentially interpretable), we cannot understand how it works, placing it in opposition to a transparent system. A classic example of a model satisfying these criteria is a shallow Decision Tree: each of its decision nodes is understandable (decomposable), the path to a prediction is algorithmically transparent, and the entire model can be simulated by a human.

These concepts are better understood as a spectrum rather than in terms of binary conditions. Unlike purely symbolic models that operate on human-readable rules, most ML models learn from experience, which gives them a degree of inherent opacity since no one hard-codes their behavior. This is especially true for deep neural networks, where the distributed, high-dimensional internal representations make complete transparency at a granular level an unattainable goal [367, 418, 414]. Consequently, the goal of XAI becomes providing explanations appropriate for a given *level of abstraction* [414]. For instance, a recommendation system that suggests items based on embedding similarity with a user representation is explainable at the *conceptual algorithmic level*, as the high-level logic of comparing semantic vectors is formalized and straightforward, and can be simulated by a human. At the *parameter level*, however, the system remains opaque, as the specific numerical values of the high-dimensional embeddings are devoid of direct semantic meaning (lack of decomposability). This shows how a model can be transparent in its high-level design while its low-level operations remain unintelligible. This principle can be extended to systems that combine opaque components with more verifiable mechanisms, an approach we explore further in Chapter 7. Ultimately, for XAI to be meaningful in real-world applications, explanations must be intentionally curated for specific levels of detail and audiences [414, 249].

3.2.2 Formal Framework for Explanations

To move beyond intuitive notions and establish a rigorous foundation, it is useful to formalize the components involved in generating an explanation. An explanation is not a static object but an interactive process [185]. It can be seen as "an interface between humans and a decision-maker that is at the same time both an accurate proxy of the decision-maker and comprehensible to humans" [156]. This section describes a formal framework that decomposes an XAI system into its core constituents, adapted from Rizzo et al. [402].

Given a ML model M that takes an input x and produces an output $\hat{y}$ (of any kind, including scalar values and vectors), the process of generating an explanation involves the following components:

- **Evidence** (e): Any objective piece of information derived from the model that offers insight into its internal workings or decision rationale. Examples include learned model parameters, feature activation values, or gradients of the loss function with respect to the input.

- **Evidence Extractor** (ξ): A computational method used to retrieve the evidence from the model. Formally, it is a function that takes the model, input, and prediction to produce the evidence: $e = \xi(x, \hat{y}, M)$. For inherently transparent models like linear regression, the evidence extraction is direct, as the model's parameters (the evidence) are already in a human-understandable format. For DL models, in contrast, the evidence must be extracted indirectly. Techniques such as gradient back-propagation or the use of proxy models can be used to generate this evidence, which then requires further processing to become human-understandable.

- **Interpretation** (g): A function that transforms extracted evidence into a human-understandable explanation by assigning semantic meaning to the raw evidence. More specifically, the function g describes the semantic relation between the evidence and the output of the model. This definition differs from some uses in the literature, as it emphasizes interpretation as one component of the explanation process. Depending on the type of evidence (which determines its *explanatory potential* and expressiveness), the interpretation functions can explain single predictions or the overall behavior of the model.

- **Explanation** (E): The final output of the interpretation function applied to some evidence, which provides an answer to a question posed by a user. Explanations can be local (for a specific prediction) or global (for the entire model). They are defined as:

$$E_{\text{local}} = g(e, x, \hat{y}, M) \tag{3.1}$$

$$E_{\text{global}} = g(e, M) \tag{3.2}$$

- **Explanation User Interface (XUI)**: The modality through which the explanation is presented to the user. This can range from visualizations like plots and heatmaps to natural language text or mathematical equations.

This framework and definitions highlight the nature of an explanation as a communicative and interactive act, tailored to the context, audience, and goals of the user [185]. This observation challenges the notion of "intrinsically" interpretable models; it suggests that any explanation is the result of a process of giving an interpretation to some evidence grounded in both ML and domain-specific knowledge. The quality of the final explanation is determined by the expressiveness of this evidence and the degree to which the interpretation adheres to the model's actual inference process (referred to as *faithfulness*). Faithfulness is often discussed alongside *plausibility*, a user-centric metric that quantifies how convincing an explanation is to a human, regardless of its technical fidelity to the model [258]. While we will occasionally reference these metrics, a more detailed perspective on XAI evaluation is provided in § 3.4.

Finally, the explanation must be conveyed through an XUI, which must be designed for the specific explanation type and end-user characteristics. The end-user, in turn, interprets the information presented on the XUI, and their understanding is shaped by their unique expectations and mental models. For example, the coefficients of a linear regressor might be immediately understood by a data scientist, while a non-expert may require correlation plots and examples to grasp the same concept. The overall efficacy of the explanation, therefore, depends heavily on a well-designed XUI tailored to guide the user's comprehension so that it aligns with the model's actual behavior. This reinforces the argument that placing the human at the center of the design loop is essential for deploying genuinely explainable systems.

Another observation is that, under this framework, an *interpretation* is treated as a testable *hypothesis* about how a model's evidence influences its output. As such, a good hypothesis should be verifiable and clearly define the dependent and independent variables. For example, in a linear regressor — formally defined as $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$ — the interpretation of a coefficient β_j as the expected change in y (dependent variable) for a one-unit change in x_j (feature j), while holding all other features constant, is mathematically guaranteed and can be directly tested. Of course, more than one interpretation is possible; an equally valid, but simplified interpretation would be that a larger coefficient for a feature indicates a greater impact on the final prediction. For complex models like neural networks, such direct testing is not possible. Instead, their behavior must be probed, for example, by using altered input examples. Many XAI methods use this strategy to learn a local approximation of the model's behavior, thereby providing a testable hypothesis that is faithful within a local neighborhood of the input.

3.2.3 Taxonomy of XAI Methods

The landscape of XAI is populated by a diverse array of methods, each with different assumptions, capabilities, and limitations. XAI techniques are typically categorized along several key axes, which define their relationship to the model, the scope of their output, and their general applicability [249, 375]. These axes are not independent but represent a set of correlated design choices and trade-offs that researchers and practitioners must navigate.

3.2.3.1 Ante-hoc vs. post-hoc

A fundamental distinction in XAI is between developing ML models that are easily interpretable by design (*ante-hoc*) and applying methods to explain opaque ML models after they have been trained (*post-hoc*) [249, 316].

Ante-hoc explainability, also known as *in-model* or *model-based* explainability, involves creating simple models that are transparent by design. These 'white-box' models have architectures that are simple or constrained enough for humans to easily understand their decision-making process. Classic examples include linear regression, where feature importance is given by the learned coefficients, and Decision Trees, where the decision path can be explicitly traced (and simulated).

More recent research has focused on developing more interpretable neural networks, such as Neural Additive Models (NAMs) [320], Concept Bottleneck Models (CBMs) [254], and Prototypical Part Networks (ProtoPNet) [208], which structure their reasoning in a way that is somewhat understandable to humans [439]. These ML models are designed to expose part of their reasoning process, allowing for the unambiguous collection of evidence to produce easy-to-grasp explanations. In practice, this requires the models to satisfy the transparency properties discussed in § 3.2.1, which directly support the interpretation of the model's evidence at a certain level of abstraction. For the reasons discussed in the previous sections, we refrain from describing these models as *intrinsically interpretable* or completely *transparent*, although this terminology is frequently used.

Post-hoc explainability refers to the application of explanation methods to a model *after* it has been trained [249]. This is the dominant paradigm in XAI research and practice, primarily because it allows practitioners to use high-performance, but complex "black-box" models and retroactively attempt to explain their behavior. These methods generally probe the model's behavior to construct an explanation, often by creating a second, simpler model to explain the first [300, 375, 339, 151]. A practical issue with post-hoc methods relates to the faithfulness of explanations to the actual model behavior (see § 3.4 for definitions of XAI metrics). Since the proxy model does not know the original model's internal logic, its accuracy depends on the number of examples used to build it. For computational reasons, this number is often small, so the proxy model may not always reliably approximate the function learned by the original black-box model [402].

3.2.3.2 Scope of the explanation: local vs. global

Explanations can be further classified by their scope: whether they explain a single decision or the model's behavior as a whole [375].

Local explanations provide insights into an individual prediction for a specific input instance. They answer the question, "Why did the model make this decision for this input?" For example, a local explanation for a loan application denial might highlight that the applicant's income was the primary reason for the decision. Feature-attribution methods paired with local approximations of the main model are often used to demonstrate which features in the input have a strong influence on the model's prediction.

Global explanations aim to describe the overall behavior of the model across the entire dataset or a large subset of it. They answer the question, "How does the model make decisions for a given output class?" For example, a global explanation might reveal that a model consistently gives high importance to the "age" feature across all predictions in a specific class. Global insights can be derived from aggregating many local explanations (e.g., SHAP's summary plots) or using methods like partial dependence plots. Model visualization [108] and decision rules [102, 8, 191] are examples of explanations falling in this category.

3.2.3.3 Applicability: agnostic vs. specific

The applicability of an XAI method to different types of models is another element of differentiation.

Model-agnostic methods are designed to be applied to any black-box model, regardless of its internal architecture. This flexibility is a significant advantage, as it decouples the choice of explanation technique from the choice of predictive model. Agnostic methods typically work by observing the model's input-output behavior, for instance, by perturbing the input and observing the change in the prediction. LIME and SHAP are canonical examples of model-agnostic techniques (see § 3.3). A method that is model-agnostic is, by necessity, a post-hoc method, as it cannot be tied to the model's design phase.

Model-specific methods, in contrast, are tailored to a particular class of models. They leverage the specific architecture or properties of the model family to generate explanations. For example, gradient-based methods like Grad-CAM [125] are designed specifically for differentiable models like neural networks, as they rely on accessing gradients and internal feature maps. While less flexible than model-agnostic approaches, model-specific methods can, in theory, provide more faithful explanations due to their direct access to the model's inner workings [339]. This category also includes various *ante-hoc* models [332, 208], where the explanation mechanism is embedded into the model's architecture itself, making them inherently model-specific by design.

3.2.3.4 Explanation modality

Finally, methods can be categorized by the form or modality of the explanation they produce.

Feature Attribution and Saliency Maps assign an importance, relevance, or attribution score to each input feature, indicating its contribution to the model's prediction. For tabular data, this often results in a bar chart of feature importance, while for image data, it produces a saliency map (or heatmap) that highlights the pixels or regions that most influenced a specific output class.

Contrastive Explanations (also known as "why-not" explanations) help users understand what would need to change in the input to achieve a different, expected outcome. They answer questions of the type, "Why did X happen rather than Y?".

Counterfactual Explanations (or "what-if" explanations) allow users to model hypothetical scenarios and observe how the model would react if one or more features had different values. Such methods can interactively engage with the user's curiosity and partial understanding of the system, allowing them to create a mental model through iterative testing. Both contrastive and counterfactual explanations are highly intuitive for human users [184].

Concept-Based Explanations represent a newer class of methods that move beyond low-level features to explain model decisions in terms of high-level, human-understandable concepts [339]. Instead of identifying important pixels, for example, such a method might report that a model's classification of an image as a "zebra" is sensitive to the presence of the concept "stripes", where this concept is defined by a set of representative example images.

Example-Based Explanations: These methods justify a prediction by retrieving influential or archetypal examples from the training data. For instance, a local explanation based on reasoning might state: "This input is classified as 'Class A' because it is most similar to these three examples of the same class from the training set." This approach leverages analogical reasoning, a natural mode of human cognition [339].

As a final note, depending on the specific field of application and context, further categorization is possible. For example, it is common to classify XAI methods based on the *training artifacts* used to derive explanations. Gradient-based methods utilize gradients (derivatives of the loss function) as a measure of input relevance, attention-based methods leverage attention weights in Transformers or recurrent models, while perturbation-based methods alter different elements of the input (features) to quantify which input changes correlate with variations in the output.

3.3 Notable Explanation Methods

This section provides an overview of influential and widely adopted explanation methods and paradigms in the literature. While not intended to be an exhaustive

review, the goal is to outline the motivations and principles behind the specific paradigms and techniques relevant to my research.

3.3.1 Model-Agnostic Post-Hoc Methods

Model-agnostic methods are highly valued for their flexibility, as they can be applied to any pre-trained black-box model. Their power lies in decoupling the explanation from the model's architecture, allowing practitioners to select the best-performing model for a task and provide an explanation afterward.

3.3.1.1 Local Interpretable Model-agnostic Explanations (LIME)

Proposed by Ribeiro et al. [100], Local Interpretable Model-Agnostic Explanations (LIME) was a pioneering method that popularized the concept of local surrogate models. The core intuition behind LIME is that while a complex model may be globally nonlinear, its decision boundary can be reasonably approximated by a simple, easily interpretable model (e.g., a sparse linear model or a small Decision Tree) in the local vicinity of a single prediction.

To generate an explanation for an instance x, LIME creates a new dataset by generating perturbations of x (e.g., by zeroing out words, tokens, or superpixels). It then utilizes the black-box model that needs to be explained (f) to get predictions for these perturbed samples and store them as a new dataset. Finally, it trains an interpretable model h (surrogate model) on this new dataset, weighting the samples by their proximity to the original instance x. The learned model h can only well approximate the original model f locally, guaranteeing *local fidelity* [479]. The resulting interpretable model h is used to produce the local explanation.

Formally, the explanation model h minimizes the following objective function:

$$h(x) = \underset{g \in G}{\arg\min}\ \mathcal{L}(f, g, \pi_x) + \Omega(g) \tag{3.3}$$

where G is the class of interpretable models (e.g., linear models), f is the original black-box model, and g is a surrogate model. The term $\mathcal{L}(f, g, \pi_x)$ is a measure of local fidelity, quantifying how unfaithful g is in approximating f in the locality defined by the proximity measure π_x. The term $\Omega(g)$ is a complexity penalty (e.g., encouraging sparsity in a linear model) to ensure the explanation is simple enough for a human to understand.

For a linear surrogate model, a locally weighted squared loss is used as the fidelity loss:

$$\mathcal{L}(f, h, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z)\ (f(z) - h(z'))^2 \tag{3.4}$$

where $\mathcal{Z}$ is the set of perturbed samples, and π_x is a kernel that assigns higher weight to samples closer to x, defined in terms of a task-specific distance function (e.g., cosine distance for text, L2 distance for image).

In the context of Rizzo et al. [402]'s framework, the *evidence* used by LIME consists of the predictions from the original model on the set of perturbed samples $\mathcal{Z}$. The underlying *interpretation* is that the learned coefficients of the locally accurate surrogate model h reflect the importance of each input feature for the prediction in the locality of the examples in $\mathcal{Z}$. Finally, this local explanation must be presented to the user through an XUI, which may take the form of plots, charts, or heatmaps, depending on the data type.

3.3.1.2 SHapley Additive exPlanations (SHAP)

While LIME provided an intuitive approach to local explanations, its results lacked strong theoretical guarantees. In response, Lundberg and Lee [133] introduced SHapley Additive exPlanations (SHAP), a unified framework for interpreting predictions grounded in cooperative game theory. SHAP assigns each feature an importance value based on Shapley values, which provide a unique and fair distribution of a model's prediction ("payout") among the input features ("players").

SHAP connects LIME with Shapley values by identifying a new class of additive feature attribution methods. An explanation model in this class is a linear function of binary variables representing whether a feature is present or absent in a coalition (i.e., a subset of features, like super-pixels, sets of tabular features, etc.). SHAP values are the only solution in this class that simultaneously satisfies three desirable properties: *local fidelity* (the explanation model h matches the original model's output for a specific input), *missingness* (missing features in a coalition have zero importance), and *consistency* (a feature's attribution should not decrease if its marginal contribution to the model increases) [133].

Formally, the explanation model h is defined as a linear function of binary variables:

$$h(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j \tag{3.5}$$

where $z' \in \{0,1\}^M$ is a simplified binary vector representing which features are present or absent in the coalition, M is the total number of features, ϕ_0 is the base value (the expected output of the model over a background dataset) and ϕ_j is the attribution for feature j, known as the SHAP value. To satisfy the desirable properties of the framework, SHAP values are defined by the theoretical Shapley values from game theory. The Shapley value is a feature's average marginal contribution across all possible coalitions:

$$\phi_j(f,x) = \sum_{z' \subseteq x'} \frac{(|z'|-1)!(M-|z'|)!}{M!} \left[f_x(z') - f_x(z' \setminus j) \right] \tag{3.6}$$

where M is the set of all features, $|z'|$ is the number of non-zero entries in coalition vector z', x' is the coalition vector for input x (all 1s), and $f_x(z' \setminus j)$ is the prediction of the original model given only the feature present in coalition z' when setting $z'_j = 0$.

As the exact computation of this formula is intractable for more than a few features, SHAP introduces practical methods to estimate these values. For model-agnostic explanations, KernelSHAP approximates the Shapley values by treating Eq (3.5) as a weighted linear regression problem. It operates in four steps: (1) sample some feature coalitions, (2) get the original model's prediction for each coalition, (3) use the SHAP kernel to assign a weight to each sampled coalition that prioritizes very small and very large coalitions, and (4) solve the weighted linear regression to find the optimal ϕ_j values (attribution). These estimated coefficients are the SHAP values, which serve as an approximation of the true Shapley values.

SHAP differs from LIME primarily in its weighting mechanism. While LIME weights perturbed samples based on their proximity to the original input, which is a heuristic choice, KernelSHAP gives the greatest importance to the smallest and largest coalitions. Intuitively, small coalitions are most informative for understanding a feature's individual impact, while large coalitions are most informative for understanding the impact of missing features. While KernelSHAP provides a model-agnostic approximation, more efficient, model-specific algorithms also exist. One of these is TreeSHAP, which can compute the exact Shapley values for tree-based models in polynomial time [188].

Similar to LIME, SHAP operates on the assumption that the behavior of a complex model f can be locally approximated by a simple, additive model h. To build this model, the *evidence* used by SHAP consists of the prediction of the original model on the altered examples described by the coalition vectors z'. After optimizing the surrogate model in Eq (3.5), the resulting SHAP values (ϕ_j) are interpreted as importance scores proportional to each feature's contribution to the prediction $f(x)$.

3.3.2 Gradient- and Activation-Based Methods

This family of model-specific techniques leverages the internal architecture of neural networks, particularly their differentiability, to generate explanations. These methods are typically more computationally efficient than their model-agnostic counterparts [479].

The core idea behind gradient-based methods is to compute the derivative of a target class score with respect to the input features. This process generates a saliency map, which is the same size as the input and contains both positive and negative values. These raw gradients (the *evidence*) are then interpreted as feature importance for the prediction: each partial derivative indicates how a small change in an input feature (e.g., a pixel intensity) would affect the target score. Large positive values imply a positive contribution to the class score; large negative values imply a negative contribution. While the underlying principle is similar, each method differs in how it computes or processes these gradients to create the final explanation.

3.3.2.1 Gradient-weighted Class Activation Mapping (Grad-CAM)

Grad-CAM, proposed by Selvaraju et al. [125], is a widely used technique for producing visual explanations for CNNs' decisions. It generates a coarse localization map, or heatmap, that highlights the regions of an input image that were important for a specific classification. Grad-CAM achieves this without requiring any changes to the CNN architecture, making it broadly applicable.

The method works by using the gradients of the score for target class c (y^c, which can be any differentiable activation produced by the model) with respect to the feature maps of the final convolutional layer (indicated as A^k). These gradients, i.e. $\frac{\partial y^c}{\partial A^k}$, indicate the extent to which each neuron in that layer contributes to the class score. The neuron importance weights, α_k^c, for a given class c and feature map A^k are computed by global average pooling [479]:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k} \tag{3.7}$$

where A_{ij}^k is the activation at spatial location (i, j) of feature map k in the last convolutional layer, and Z is the number of pixels in the feature map. Weight α_k^c is interpreted as the "importance" of feature map k for a target class c.

The final Grad-CAM heatmap, $L_{\text{Grad-CAM}}^c$, is a weighted combination of the feature maps (weighted on the gradients), followed by a Rectified Linear Unit (ReLU) to isolate features with a positive influence on the class of interest:

$$L_{\text{Grad-CAM}}^c = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right) \tag{3.8}$$

The success of Grad-CAM has led to a family of derivative methods (such as Grad-CAM++ [142]) and complementary techniques (like SmoothGrad [115]) that aim to produce more detailed and faithful visualizations.

3.3.2.2 Integrated Gradients (IG)

Standard gradient-based attribution methods, which simply use the gradient of the output with respect to the input, can suffer from issues like gradient saturation, where the gradient becomes zero even for highly influential features. Integrated Gradients (IG), introduced by Sundararajan et al. [118], is an axiomatic attribution method designed to overcome this limitation.

IG attributes a model's prediction to its input features by accumulating the gradients along a straight-line path from a baseline input x' (i.e., an input where the model's prediction is close to a neutral value, for example, 0 or the average prediction) to the actual input x that we want to explain [434]. Formally, the Integrated Gradients attribution for the i-th feature is defined as the path integral

of the gradients:

$$\mathrm{IG}_i(x) ::= (x_i - x_i') \times \int_{\alpha=0}^{1} \frac{\partial f(x' + \alpha(x - x'))}{\partial x_i} d\alpha \tag{3.9}$$

where f is the model's output function and α interpolates along the path from the baseline x' to the input x. In practice, computing this integral is not always feasible and must be approximated. Therefore, IG approximates the integral using a Riemann sum, which involves summing the gradients at a finite number of discrete steps along the path from the baseline x' to the input x.

A key advantage of IG is that it satisfies the *completeness* axiom, which states that the sum of the attributions across all features is equal to $f(x) - f(x')$. This ensures that the full change in prediction is distributed among the features [118].

3.3.2.3 Layer-wise Relevance Propagation (LRP)

Layer-Wise Relevance Propagation (LRP) is another technique for explaining the predictions of neural networks [85]. It operates by propagating the model's output score backward through the network, layer by layer, until it reaches the input features. At each layer, the relevance is redistributed from higher-level neurons to lower-level neurons based on their contribution to the activation of the higher-level neurons.

The core of LRP is a set of local propagation rules that adhere to a relevance conservation principle. This principle ensures that the total amount of relevance is preserved at each step of the backward pass, such that the sum of the relevance scores at the input layer equals the model's original output score [140]. Let j and k be indices for neurons of two successive layers. Let R_k be the relevance of neuron k for the prediction $f(x)$. We define $R_{j \leftarrow k}$ as the share of R_k that is redistributed to neuron j in the lower layer. The conservation property can be expressed as:

$$\sum_j R_{j \leftarrow k} = R_k \tag{3.10}$$

Under certain conditions, LRP can be interpreted as a deep Taylor decomposition, which gives the following basic propagation rule:

$$R_j = \sum_k \frac{z_{jk}}{\sum_j z_{jk}} R_k \tag{3.11}$$

where R_j is the relevance of a lower-layer neuron j, R_k is the relevance of a higher-layer neuron k, and z_{jk} quantifies the contribution of neuron j to the activation of neuron k (e.g., $z_{jk} = a_j w_{jk}$ in a linear layer). Different propagation rules have been developed to improve stability and handle different types of layers (e.g., normalization or pooling layers), making LRP a versatile explanation method [140, 112].

3.3.3 Semantic and Case-Based Explanations

Feature attribution methods, such as those described in the previous sections, typically produce pixel-wise or token-wise heatmaps. However, these methods have inherent limitations. The importance of a single pixel, for instance, reveals little about what that pixel represents or how it is combined with others in deeper layers to form higher-level concepts. Furthermore, the expressiveness of such explanations is limited to the vocabulary of the raw input features, which is inadequate for unambiguously conveying such higher-level concepts [479, 339].

Recognizing the "semantic gap" between the low-level features used to construct explanations and the high-level reasoning employed by humans, concept- and prototype-based methods have been proposed. These techniques aim to provide explanations in terms of human-understandable concepts or illustrative examples, thereby aligning more closely with how humans explain their own decisions [184, 479].

3.3.3.1 Testing with Concept Activation Vectors (TCAV)

The Testing with Concept Activation Vectors (TCAV) method, developed by Kim et al. [151], provides a way to quantify the importance of a high-level, user-defined concept for a model's prediction. For example, instead of asking which pixels are important for classifying the subject of an image as a "zebra", TCAV answers the question: "How important is the concept of 'stripes' for the 'zebra' classification?" [479].

TCAV works by first defining a concept with a set of user-provided examples (e.g., images containing stripes as representative of the "stripes" concept). The second step is learning a Concept Activation Vector (CAV), $\mathbf{v}_l^C$, which is a vector oriented in the direction of the concept C's examples in a target layer l's activation space. This is achieved by training a linear classifier to separate the target layer activations of the concept examples from the activations of random examples; the vector normal to the separating hyperplane is the CAV.

The importance of the concept C for a class k is then measured by its *conceptual sensitivity*, which is the directional derivative of the class logit in the direction of the CAV. Intuitively, this measures how sensitive the model's prediction is to the presence of the concept C in layer l's activation space [151]. For an input x, this is:

$$S_{C,k,l}(x) = \nabla h_{l,k}(f_l(x)) \cdot \mathbf{v}_l^C \tag{3.12}$$

where $f_l(x)$ is the activation vector at layer l and $h_{l,k}$ is the function mapping the activation of layer l to the logit for class k.

TCAV treats the conceptual sensitivity scores from many examples as *global evidence* that is interpreted into a global explanation. The intuition is that since the gradient $\nabla h_{l,k}(f_l(x))$ points in the direction that most rapidly increases the logit for class k, a positive sensitivity score $S_{C,k,l}(x) > 0$ means the concept's direction aligns with the direction that favors the class (the angle is less than 90°). Based on this

premise, a positive sensitivity score is interpreted as evidence that the concept C encourages the model to classify an input as class k. Finally, to produce a global explanation, TCAV simply calculates the fraction of examples in a class for which the concept has this positive influence:

$$TCAV_{Q_{C,k,l}} = \frac{|\{x \in X_k : S_{C,k,l}(x) > 0\}|}{|X_k|} \tag{3.13}$$

As TCAV requires user-selected examples, validation of the learned concepts is necessary to ensure the model has learned significant concepts that are aligned with human expectations. The authors propose various strategies for this. One is statistical significance testing, where multiple CAVs are trained using different sets of random examples while the concept examples remain fixed. A meaningful concept should yield consistent TCAV scores across these runs, which can be validated with a two-sided t-test against scores from a random concept. Other proposed validation strategies include manually inspecting images sorted by their affinity with the concept or producing artificial inputs that maximize CAV activation (an approach called "empirical deepdream"). Both of these approaches allow for visual inspection and can be used to identify spurious or misaligned CAVs.

TCAV represents a significant step toward more human-centric global explanations by allowing domain experts to test hypotheses about their models using concepts they understand and feel relevant. Despite its advantages, TCAV has several practical limitations [479]. First, it requires significant manual effort to curate a clean dataset of examples for each concept being tested. Second, its effectiveness is often limited in shallower networks, as the concepts learned in early layers tend to be less separable in the activation space. Finally, and most critically, verifying abstract concepts remains a challenge. For instance, it is difficult to ensure a deep neural network has truly learned abstract concepts like "sadness", as it may have simply identified a correlated one like "crying" or "funeral". This raises the difficult question of whether the model has generalized to the intended concept or is merely exploiting a spurious correlation.

3.3.3.2 Prototype-based reasoning

Another approach that mimics human reasoning is example-based or prototype-based explanation [52]. These methods justify a model's prediction by identifying the most similar or archetypal examples from the training data. One key difference from the previous set of concept-based models is that prototypes are typically learned automatically, making them suitable for applications where users do not have prior knowledge [300, 143]. For instance, a medical diagnosis system might explain its prediction of a malignant tumor by displaying learned archetypal patches of malignant tumors from its training set. This approach is intuitive because it relies on analogical reasoning (comparing a new case to representative "anchors"). While

many of these methods are intrinsic (ante-hoc), with the explanation emerging directly from the model's architecture, post-hoc versions also exist [439].

An example of an intrinsic, or ante-hoc, prototype-based model is the Prototypical Part Network (ProtoPNet) [208]. ProtoPNets are designed to make predictions by first identifying prototypical parts of an object in an image and then combining evidence from these prototypes to make a final classification. The architecture consists of three main components:

- **Convolutional Backbone**: a standard CNN (e.g., VGG or ResNet), with its final classification layer removed, acts as a feature extractor, mapping the input image into a lower-dimensional latent space.
- **Prototype Layer**: models prototypes, with each prototype represented as a vector in the same latent space as the features extracted by the backbone. During training, the layer learns a fixed number of prototypes for each class. To make a prediction, it computes a similarity score between each prototype and patches of the convolved image, generating a similarity map for each prototype. This feature map is then max-pooled to produce a single similarity score for each prototype, which is interpreted as a measure of how strongly a prototypical part is present in any patch of the input image.
- **Fully Connected Layer**: computes a weighted sum of the similarity scores produced by the prototype layer to generate the final class logits. After training, the weights in this layer can be interpreted as the extent to which a specific prototype contributes to a given class (global explanation).

The end-to-end training process allows the network to learn prototypes that are assumed to represent meaningful visual concepts. The resulting similarity scores for these prototypes are then used as input features for the final fully-connected layer. The model is also regularized to encourage the prototypes to be distinct from one another. The explanation for a prediction consists of the visually-grounded prototypes that were most activated, along with their similarity scores and locations in the input image. In the context of the XAI framework previously described, the *evidence* for an explanation is the set of similarity scores between the input image and the learned prototypes. The *interpretation* is embedded in the model's architecture: a high score means a specific prototypical part is present, and the weights of the final layer determine how the presence of each part contributes to the final classification. This is what makes the model's reasoning process transparent and more aligned with human analogical reasoning.

While ProtoPNet is an interpretable model that must be trained from scratch, other methods have been developed to bring prototype-based reasoning to existing black-box models in a post-hoc fashion. One such method is the Explanation Neural Network (XNN) [300]. XNN is a post-hoc explanation module that can be attached to an intermediate layer of a pre-trained deep network. It works by learning a

non-linear embedding of the high-dimensional activation vectors from the original network into a much lower-dimensional, more interpretable "explanation space". The reduced dimensionality of this space enables the extraction of a few high-level prototypes that can be visualized and understood by humans. Finally, the visualization of the learned explanation space can be achieved with various post-hoc techniques like Grad-CAM.

However, a limitation of prototype-based methods is the potential ambiguity of the similarity itself. When a model explains its decision by showing that an input is "similar" to a prototype, it doesn't specify on what basis this similarity was calculated. The high-dimensional vector distance may not align with human-perceived similarity (e.g., pattern, color, shape). This can lead to explanations that are convincing yet incomplete, as the user is left to infer the reason for the similarity, which may not faithfully reflect the full complexity of the model's logic.

3.3.4 Generative Explanations

A newer trend in XAI seeks to have models learn to generate natural language explanations, or rationales, that justify their decisions. This approach aims to bridge the semantic gap between low-level explanatory evidence (like feature attributions) and a high-level explanation, moving from heatmaps and concepts to fully formed, human-readable narratives. Natural language explanations are often more intuitive and immediately useful, as they can mirror the high-level concepts and causal language that humans use in their own reasoning [432].

Rationale generation methods are typically either *extractive*, selecting verbatim subsets of the input as evidence [104], or *abstractive*, generating free-form natural language to explain a decision. While abstractive rationales offer greater expressiveness, this flexibility makes them more susceptible to producing convincing explanations that are not directly grounded in the input and the model's true reasoning process.

3.3.4.1 Pipeline models: explain-then-predict

Pipeline methods decouple the process of explanation from prediction. These approaches, often called *explain-then-predict* systems, first train a component to produce a rationale and then train a second, separate component to make a prediction based solely on that rationale. This architectural constraint is designed to ensure that the prediction is grounded in the provided explanation.

One work in this category is the extractive rationalization model by Lei et al. [104], which introduced a two-component architecture: a generator and an encoder. The generator acts as a selection mechanism, identifying a concise subset of the input text (the rationale). The encoder then makes a prediction using only this selected text. The components are trained jointly with objectives that encourage the generator to select short, coherent text spans that are sufficient for the encoder

to make an accurate prediction. By highlighting the exact text used as evidence, this method provides a grounded explanation. The architectural constraint that the predictor only receives the rationale as input is designed to improve faithfulness and demonstrates sufficiency (i.e., that the rationale alone is sufficient for the prediction).

While powerful, the extractive approach is limited to selecting existing text. Other pipeline models focus on generating more flexible, abstractive rationales. For example, recent work in the context of LLMs has explored hybrid pipelines to improve faithfulness. Turpin et al. [419] propose a multi-stage approach where a LLM uses CoT prompting to generate a symbolic, step-by-step reasoning plan. This plan is then passed to an external, deterministic solver to compute the final answer. This method represents a promising direction for LLM-augmented problem solving. It enhances transparency by making the reasoning process explicit and verifiable by an external tool, even though the initial generation of that reasoning by the LLM remains an opaque step.

3.3.4.2 Self-rationalization: jointly predicting and explaining

In contrast to pipeline models, the self-rationalizing paradigm uses a single, unified model trained to output both a task prediction and its rationale simultaneously. The explanation and prediction are generated together in an end-to-end fashion, often using a multi-task learning setup.

A typical architecture for this paradigm features a shared model body with two distinct output "heads": one that performs the primary task (e.g., classification) and another that generates the supporting text [364, 318, 148]. For instance, Rajagopal et al. [318] proposed a multi-task architecture built on a Transformer encoder, jointly trained to both predict and explain. The resulting explanation consists of outputs from a local module that estimates feature importance and a global explainer that identifies relevant concepts. In the domain of visual question answering, Park et al. [148] employed a multi-task architecture that learns an attention map to identify relevant image regions; these features, in turn, guide an LSTM decoder in generating a textual rationale conditioned on the image, question, and answer. The recent development of LLMs has also popularized a form of self-rationalization through ICL. Prompting techniques like CoT encourage LLMs to generate a step-by-step reasoning trace before giving a final answer, effectively producing a rationale for their prediction [389, 425].

The premise of the self-rationalizing paradigm is that by constraining a model to generate a prediction and an explanation simultaneously, the rationale will be faithful to its actual reasoning. However, the validity of this assumption has been challenged. Research has shown that these models can still produce plausible-sounding rationales that are unfaithful to the features that actually drove the prediction. At the very least, it has proven difficult to establish a clear correlation between the generated rationales and the model's decision-making process [259, 419, 403].

3.4 Evaluating Explainability

Evaluating the quality of an explanation is a complex, multidimensional problem with no single solution. While the research community has developed a range of metrics to assess the intrinsic properties of explanations (such as faithfulness and stability), it has made far less progress on systematically evaluating their utility from an end-user perspective [467, 257, 479].

While intrinsic properties are essential, the ultimate efficacy of an explanation is tied to its reception by a human user. We support the opinion that explanations should not be optimized solely for subjective criteria, as this can encourage models to generate convincing yet unfaithful justifications [414, 467]. Nevertheless, since the goal of an explanation is to be understood, the end-user perspective must be considered in the evaluation process.

The quality of an explanation is typically judged by several orthogonal criteria, each with its own evaluation metrics. Doshi-Velez and Kim [167] distinguish levels at which explanation methods can be tested: *application-level* testing (experiments in specific settings with real end-users), *human-level* testing (experiments with human participants outside their applied contexts), and *function-level* testing, where users are not required (for example, computational "intrinsic" properties of explanations are measured). While functional criteria do not depend on human opinions, human- and application-level testing focus on human-centered evaluation, assessing the practical usefulness of explanations for end-users. This section outlines the most common criteria discussed in the literature.

3.4.1 Functional Criteria

Faithfulness *Faithfulness*, also referred to as *fidelity*, measures how accurately an explanation reflects the model's internal reasoning process [479, 258]. It is often considered the most critical technical property of an explanation, as it concerns the truthfulness of the explanation with respect to the model being explained. Since direct measurement is impossible for black-box models, faithfulness is typically assessed using quantitative proxy metrics.

The most common approach involves *perturbation tests*, which measure the impact of altering important features on the model's output [199, 112, 119]. Specifically, *deletion* tests systematically remove the features an explanation identifies as most important and observe the effect on the model's prediction. A sharp drop in prediction confidence indicates that the explanation correctly identified influential features. Conversely, *insertion* tests progressively add the most important features to a baseline input (e.g., a blurred image or zero vector). A sharp increase in prediction confidence suggests the explanation has identified features sufficient to support the prediction.

To formalize these tests, benchmarks such as ERASER [256] define metrics like *comprehensiveness* and *sufficiency*. Given a model's confidence $p_{c(x)}$ for the predicted

class of an input x, and $x_{:k\%}$ as the top-$k\%$ most important features, comprehensiveness measures the drop in confidence when these features are removed:

$$\text{COMP} = \frac{1}{|B|} \sum_{k \in B} (p_{c(x)}(x) - p_{c(x)}(x \setminus x_{:k\%})) \tag{3.14}$$

Sufficiency measures how much of the original confidence is retained when only the top-$k\%$ features are kept:

$$\text{SUFF} = \frac{1}{|B|} \sum_{k \in B} (p_{c(x)}(x) - p_{c(x)}(x_{:k\%})) \tag{3.15}$$

A more faithful explanation method is expected to achieve higher comprehensiveness and lower sufficiency scores. These metrics have become a standard approach for assessing faithfulness in recent literature [353, 410].

Stability *Stability*, often discussed in terms of its inverse, *sensitivity* [175], measures whether an explanation method produces similar explanations for similar inputs on a fixed model. An ideal explanation should be stable (i.e., have low sensitivity), meaning it does not change significantly in response to minor input alterations that do not change the model's prediction.

However, many explanation methods have been shown to fail this criterion. Kindermans et al. [222] demonstrated that some feature importance methods, including IG and LRP, are sensitive to simple, constant shifts in the input, even though such shifts do not affect the output of many models. Furthermore, a significant body of work has shown that small, adversarially optimized perturbations can be crafted to arbitrarily alter an explanation while leaving the model's prediction unchanged [212, 195, 209, 255, 166]. A lack of stability can result from high variance in the explanation method itself, or from non-deterministic components such as the data sampling step used by local surrogate methods like LIME. Low stability undermines user trust, as the resulting explanations can appear feeble, noisy, or arbitrary.

Consistency *Consistency* evaluates whether two models with similar predictive behavior also produce similar explanations for the same input. While *stability* assesses one model's explanations for similar inputs, *consistency* assesses different models' explanations for the same input. The underlying assumption is that if two models have learned a similar function, their reasoning process and their explanations should also be similar. In practice, however, two models with identical performance can yield divergent explanations, partly due to the variance of certain explanation methods [175, 398].

Simulatability *Simulatability* was originally introduced as a human-centric criterion to measure an explanation's utility by testing if a user could "simulate" a model's behavior in forward (given the input and explanation) or counterfactual

scenarios (predicting the model's output on a perturbed input) [257]. However, this concept has since been adapted into an automated proxy task to measure the quality of explanations, especially free-text rationales, without human involvement [319]. In this setup, the usefulness of a rationale is quantified by how much it improves the performance of a second "simulator" model that uses both the original input and the rationale, compared to a baseline model that does not see the rationale. A higher performance gain indicates a more useful explanation, assuming the rationale does not inadvertently leak the target label [257].

3.4.2 Human-Centric Criteria

Unlike function-level criteria that measure intrinsic properties, human-level evaluation assesses how effective, useful, and satisfying an explanation is for an end-user.

Plausibility Distinct from faithfulness, which is model-centric, *plausibility* is a human-centric metric that measures how convincing or reasonable an explanation appears to a human user [258, 458]. It assesses the alignment between a machine-generated explanation and a human's prior knowledge or intuition about the task. For example, an explanation for a bird classifier that highlights the bird's beak would likely be considered more plausible than one that highlights a random part of the image background, regardless of its faithfulness.

Plausibility is a qualitative property and is almost exclusively evaluated through human-grounded studies. Common evaluation strategies include [438]:

- **User studies**: Participants are shown explanations and asked to rate them on Likert scales according to criteria like understandability and perceived accuracy. Alternatively, users are presented with explanations from multiple XAI methods for the same prediction and asked to select the "most reasonable" one [304].
- **Comparison to Human Rationales**: In tasks where human-annotated ground-truth explanations exist (e.g., salient text spans or image regions), plausibility can be quantified by measuring the overlap or agreement between the machine explanation and the human rationale [467].

However, a growing consensus in the literature urges that plausibility be treated with caution [416, 165, 234], or even avoided as a primary objective [467]. Optimizing for plausibility can lead to unfaithful yet convincing explanations that exploit human biases rather than revealing the model's true logic, thereby creating a false sense of trust [402, 258].

Understandability *Understandability*, also called *comprehensibility*, measures the degree to which a user can comprehend the information presented by an explanation. Unlike plausibility, which relates to a user's expectations, understandability

is concerned solely with the cognitive ease of processing the explanation itself [402]. Although difficult to measure directly, it is often assessed using proxies. For instance, an explanation's complexity, such as the number of features it highlights, can indicate its cognitive load. Similarly, explanations based on a model's original inputs can be more understandable than those using heavily transformed features. A user's ability to predict a model's behavior after seeing its explanation (i.e., user simulatability) can also serve as a proxy for their comprehension [479, 175].

Usability *Usability* is a broad, task-dependent criterion that evaluates how effectively an explanation helps a user achieve a specific goal, measuring its practical value in a given context. This criterion is not consistently defined, but typically encompasses several desirable properties, such as *actionability* (i.e., the degree to which it provides insights a user can act upon, for instance, to debug or correct the model) and the ability to convey the model's *certainty* in its predictions [175, 479]. Furthermore, usability is strongly linked to *interactivity*. Users tend to place more trust in systems that provide information to help them accomplish tasks with greater confidence, and an interactive explanation is often more usable than a static one [156]. The ability to probe the XUI, test hypotheses, or explore counterfactuals is therefore another aspect of a usable system.

Finally, it is important to note that much of the terminology for the user-centered criteria is not yet standardized in the XAI community, and terms are often used with different meanings in different contexts [438].

3.5 Challenges and Future Directions

Despite significant advancements, the field of XAI faces several fundamental challenges that question the reliability and adequacy of current paradigms. This section outlines the most significant of these issues.

3.5.1 Unreliability of Explanations

As partly discussed in earlier sections, a primary challenge is ensuring that an explanation is *faithful* to the model's internal reasoning process. A significant body of evidence suggests that popular explanation methods can be plausible yet unfaithful. Feature attribution methods, for example, have been shown to be surprisingly fragile; studies have demonstrated that explanations from LIME and SHAP can be manipulated by adversarial attacks without changing the model's prediction [235]. Similarly, gradient-based methods have been shown to lack reliability. They have failed "sanity checks" by producing seemingly reasonable explanations even for models with randomized parameters [165], and their high sensitivity to reference points and minor input transformations can lead to unstable explanations [222]. Furthermore, the non-linearity of deep models means that significant shifts in

important features can sometimes result in a zero gradient, rendering them invisible to these methods [387]. The use of attention weights as explanations has also been criticized, with research finding that multiple attention distributions yield suspiciously identical results and discovering a weak correlation between attention scores and other feature importance measures [199, 193].

This faithfulness problem extends to self-rationalizing models. Recent work has shown that the CoT rationales produced by LLMs can be misleading; they may offer plausible but incorrect justifications for biased or stereotyped answers, and can even be generated for deliberately flipped predictions [419, 364]. This creates a significant risk of engendering false trust in systems whose true reasoning remains opaque.

3.5.2 The Complexity of Rigorous Evaluation

Another major challenge is the difficulty of rigorously evaluating and comparing explanations, a problem exacerbated by the community's tendency to prioritize developing new explanation methods over procedures to validate existing ones [479, 449]. The root of this issue is the lack of a formal definition of what constitutes an "explanation". Different expectations and formulations lead to divergent evaluation approaches, as exemplified by the "Attention is not Explanation" debate [193, 207]. This ambiguity has led to a proliferation of metrics, which in turn results in the problem of disagreement across these metrics: different automated metrics for properties like faithfulness often conflict, yielding inconsistent conclusions [407, 423]. This suggests that the search for a universal "gold standard" metric is futile. Since "explanation" is a user-centric concept without a formal definition and a univocal ground truth, a one-size-fits-all evaluation is not feasible. Progress will instead depend on developing robust, task-specific evaluations based on specific desiderata, which need to include the end-user in the equation [414].

3.5.3 Explaining Generative LLMs

The transition from discriminative LMs (e.g., BERT [192]) to generative LLMs has made explainability an even more ambitious goal. While traditional approaches often focused on feature attribution in smaller, fine-tuned architectures, the emergent properties of LLMs, such as ICL and reasoning, introduce new complexities.

On one side, the scale of generative AI models renders traditional techniques, like gradient-based methods and SHAP [133], computationally intractable. Exact Shapley value calculation, for instance, faces computational complexity exponential in the number of features, which is prohibitive for the long context windows typical of LLMs [444, 372]. Additionally, the high dimensionality of these models makes it difficult to produce locally accurate surrogate models that are sufficiently simple to be interpretable yet able to approximate the model's behavior [456].

In light of these challenges, new explanation techniques tailored to the prompting paradigm are emerging. Rationalizations and CoT explanations, for example, appear

to offer transparency by verbalizing the model's reasoning process. However, some recent studies suggest that these explanations can be systematically unfaithful [419, 364]. For instance, when Turpin et al. [419] introduced biased text into the prompt to steer the model's prediction, models often yielded to the bias while generating a CoT explanation that fabricates a plausible semantic justification, completely ignoring the artificially introduced bias. Paradoxically, this unfaithfulness is aggravated by model scale. Research indicates that larger, more capable models are more likely to produce unfaithful CoT reasoning than smaller models, effectively generating convincing rhetoric that misaligns with their internal decision-making processes [399]. These explanations may be uninformative or contain hallucinated and factually incorrect information, obscuring the true causal drivers of the prediction [474].

The explanations are further complicated by the models' ICL capabilities, which require distinguishing between reliance on semantic priors (knowledge acquired during pre-training) and symbolic reasoning (learning arbitrary mappings from the prompt). Research indicates a divergence based on model scale: smaller models tend to rely heavily on pre-trained semantic priors, often failing tasks where prompt demonstrations contradict these priors (e.g., flipped labels). In contrast, larger models demonstrate an emergent ability to override semantic priors and follow in-context input-label mappings, effectively performing a form of symbolic reasoning unconstrained by prior knowledge [395].

Finally, proprietary and closed-source models exacerbate the "black box" problem. Since model weights are not disclosed, gradient-based XAI methods (such as Grad-CAM or IG) are rendered inapplicable, while model-agnostic alternatives like SHAP become too expensive due to the high volume of API queries required for estimation. It also complicates uncertainty quantification, forcing researchers to estimate confidence via prompts (i.e., asking the model to verbalize its certainty) or sampling strategies. However, relying on verbalized confidence is complicated by issues of calibration, hallucination, and *sycophancy*, defined as the tendency of instruction-tuned models to generate answers that align with the user's stated views or misconceptions rather than objective facts [422]. These issues are aggravated by the integration of LLMs into complex pipelines, such as RAG systems. In these setups, the model's output depends not only on its internal weights but also on the provenance and quality of retrieved data, rendering the reasoning process even more difficult to describe.

3.5.4 Future Directions

In response to the challenges of producing and evaluating explanations, the future of XAI is being shaped by several paradigm shifts. A significant direction is a renewed focus on *ante-hoc* systems, which are transparent by design [191, 8, 249]. In these "white-box" models, the explanation follows from the system architecture, mitigating the faithfulness problem that plagues post-hoc methods. While this approach may involve a trade-off, sacrificing some predictive power by using simpler architectures

compared to state-of-the-art black boxes, it offers a direct path toward building trustworthy models that can be fully validated by humans.

Alongside this, there is a growing recognition of the user-centric nature of the explanation process, leading to a focus on *Human-in-the-Loop* (HITL) systems [482]. Here, the focus moves from producing static outputs to enabling interactive dialogues, often facilitated by well-designed XUIs that enhance the *usability* of the overall system [366, 156]. Designing these interfaces effectively requires an interdisciplinary approach; experts from multiple fields must collaborate to determine what information is relevant for a specific audience, how it should be formalized, and how it can be conveyed clearly through the XUI [414]. By treating explanation as a collaborative and interdisciplinary process, the HITL paradigm aims for a deeper cognitive alignment between the system and its users, enabling bidirectional feedback loops that can ultimately improve the AI model itself.

Finally, a third direction seeks a pragmatic middle ground by designing hybrid systems that combine opaque and transparent components. This approach strategically limits the role of black-box models in favor of more transparent components. For instance, an opaque deep neural network can be used for complex feature extraction (e.g., creating embeddings), while a simpler, transparent model (like a Decision Tree) makes the final prediction using only those high-level features. The trade-off is managed by shifting decision-making power from the opaque feature extractor to the transparent predictor, which is then used to generate a faithful explanation. This modular approach is particularly relevant for systems leveraging modern LLMs. Since perfect transparency in such models is likely unattainable, recent frameworks use them for tasks like knowledge extraction while integrating that information into more verifiable pipelines [419, 431]. This focus on building transparent pipelines from constrained, opaque parts is a highly promising path for deploying advanced AI safely.

II

Classification for Multilingual and Noisy Text

This part presents my contributions to the development and application of novel models and benchmarks for text classification (TC) in various settings. Chapter 4 describes a comprehensive benchmark of language models (LMs) across non-English languages, created to evaluate their performance in less-resourced contexts. It analyzes the capabilities of popular LMs in classifying diverse textual content (such as news articles and encyclopedia entries) in Italian and French, compared to English. This foundational work, conducted at the start of my doctoral studies, helped to identify challenges and establish baselines for TC, providing the necessary groundwork for the subsequent research. Chapter 5 describes a system for automating the classification of customer support tickets. This domain presents a unique challenge for TC due to the noisy, jargon-rich text generated by users on highly technical topics. Furthermore, as ticket categories are often organized hierarchically, this problem serves as a practical instance of hierarchical text classification (HTC).

4

Text Classification on Non-English Languages

While deep learning (DL) has rapidly advanced TC, state-of-the-art approaches have been developed for and benchmarked against English datasets, leaving other languages to deal with inevitable linguistic challenges. Using Italian and French as case studies, this chapter reviews the availability of annotated task-specific datasets and presents practical and computational linguistic considerations concerning the application of modern LMs to these languages. To substantiate our analysis, we compile and compare inventories of datasets available for both languages at the time of writing. Furthermore, to simulate a real-world scenario, we apply representative TC methods to two new multilabel classification datasets in Italian, French, and English, adapted from existing corpora containing hierarchically-organized data. The chapter concludes by discussing results and future research directions from a linguistically inclusive perspective.

4.1 Introduction

Modern natural language processing (NLP) methods rely on ingesting massive amounts of text to effectively model a language, meaning the limited availability of such corpora in many languages constitutes a serious obstacle. While multiple recent works have reviewed TC from a generic perspective [350, 189, 306], they remain mostly English-centered and often do not assess the linguistic challenges involved when applying these methods to other languages.

To understand how language-specific resources influence the development of machine learning algorithms, it is useful to discuss the *resource categorization of languages*. In computational linguistics, languages are often classified as low-, mid-, or high-resource. These resources include not only raw text data but also linguistic tools, such as lemmatizers and part-of-speech (PoS) taggers, that are often required for preprocessing. While no standard categorization exists, English and Mandarin Chinese are widely considered to dominate the high-resource end of the spectrum. Other languages often cited as high-resource include Arabic, French,

German, Portuguese, and Spanish.

Despite the existence of multiple high-resource languages, the vast majority of research implicitly focuses on English [202]. This reality highlights the need for NLP models that can generalize to languages and data beyond their initial training sets [45]. This goal is closely related to the concept of *language independence* [61], an attribute of models designed to work comparably well across multiple languages (e.g., cross-lingual and multilingual models). In practice, applying TC procedures to non-English languages can be challenging, either due to a lack of suitable benchmarks or because the required linguistic tools are lacking or perform differently.

Throughout this chapter, we use Italian as our primary case study to investigate these challenges. It is important to situate this work in its original context, as the analysis and experiments in this chapter were conducted in 2022 and are largely derived from my earlier publications on this subject [2, 1]. While the foundational challenges discussed remain relevant, the data landscape for many languages has evolved since that time, largely driven by initiatives related to the development of generative LLMs.

4.1.1 Task and Approach

The primary objective of this chapter is to provide a quantitative and qualitative analysis of the challenges in developing classifiers for text in non-English languages. In practice, these challenges come down to two core issues: the scarcity of high-quality labeled data for a variety of classification tasks, and the performance disparities of TC methods between languages.

Our approach involves three main steps. First, we compile and compare inventories of publicly available datasets for Italian and French to measure the disparity in resource availability. Second, we create two new datasets for topic labeling and news classification in Italian, French, and English. Finally, we use these datasets to conduct a benchmark study, applying a set of representative algorithms to quantitatively measure and discuss the performance differences between English and the two less-resourced languages.

4.1.2 Contributions

The contributions of this work can be summarized as follows:

- We discuss the issue of labeled data availability for 7 distinct and common TC subtasks, substantiating our analysis by comparing dataset inventories across three languages;
- We synthesize two new multilabel datasets: one from the RCV1/RCV2 collection of Reuters articles and another, It/Fr/EnWiki-100, using Wikipedia articles and their topics in the three respective languages;

- We assess the practical applicability of TC methods by comparing the performance of 7 LMs, including both traditional and modern pre-trained paradigms, on the tasks of news classification and topic labeling;
- We analyze the link between model performance and specific issues caused by the availability of language resources and differences in model architectures or pre-training.

The datasets and code used for the experimental part of this work are publicly released (where legally permitted) for reproducibility[1].

4.2 Availability of Annotated Data

The availability of annotated corpora is essential for NLP research. While the development of deep LMs mainly leverages self-supervised strategies, labeled datasets are required for supervised tasks like TC, a necessity that has been further emphasized by the recent focus on LLMs. This section presents a survey of resources that were available at the time of writing for two European languages: Italian, which we regard as a mid-resource language, and French, a high-resource one. We compare their offerings to the landscape of English NLP resources available at that time.

While a consistent number of written English annotated corpora is available and heavily referenced in the literature, we find that the quantity of easily accessible resources in the languages we considered is still lacking (especially in Italian), limiting research on this theme. To clarify, this is a fundamentally different issue from the one posed by low-resource languages, usually characterized by insufficient unlabeled data to even be able to perform the self-supervised procedures that are essential for pre-training LMs.

4.2.1 Overview

Text classification tasks We conduct a scientific literature search of annotated textual corpora presented or employed in research contributions. Reflecting Li et al. [350], we focus on the following common TC sub-domains:

- **Sentiment analysis (SA)**: the task of understanding the emotional content of a piece of text, usually mapping it to predefined categories representing specific emotions. We include in this category the tasks of stance and polarity detection, as well as the identification of rhetorical devices, like irony, or linguistic properties, like subjectivity.
- **Topic labeling (TL)**: the task of extracting the topic (or theme) of a document, for example, an article. This task is often related to content recommendation, as it can be used to map textual content to user interests.

[1] https://gitlab.com/distration/dsi-nlp-publib/-/tree/main/text-class-survey-22

- **News classification (NC)**: classification of news into specific categories, like user interests or topics.
- **Question answering (QA)**: extractive question answering can be framed as a classification problem where the model, given a list of candidate sentences extracted from text and a target question, must decide which sentence contains the answer.
- **Natural language inference (NLI)**: given a pair of statements, the task is to determine if one is entailed by the other.
- **Named entity recognition (NER)**: the task of locating and classifying named entities mentioned in unstructured text into predefined categories.
- **Syntactic parsing (SP)**: the task of predicting the morpho-syntactic properties of words in sentences, like part-of-speech (PoS) tagging, speech dependencies, and semantic role labeling.

These tasks can be adapted to various domains, and numerous subtasks with different formulations are possible. They are commonly used as benchmarks in NLP research, especially as part of multitask natural language evaluation initiatives, like the General Language Understanding Evaluation (GLUE) benchmark [164].

Search criteria To balance search time and effectiveness, our search strategy is composed of three steps:

1. Search for datasets on Google Dataset Search [2];
2. Search for publications on Google Scholar using keywords along the lines of "Italian text corpus" and "Italian Text Classification".
   (a) The first two pages of results sorted for pertinence are explored;
   (b) The same is repeated by filtering results based on their publication date, by looking at contributions published from 2019 onwards;
3. Search on PapersWithCode [3] for contributions using the same keywords.

For every publication, we explore all publicly referenced and accessible data sources.

4.2.2 Datasets in Other Languages

In this section, we present the results of our search, omitting corpora that are not public or that are unavailable at the time of our search. We further filter out datasets with fewer than a few thousand labeled samples, unless they are highly specialized or heavily referenced.

[2] `https://datasetsearch.research.google.com`
[3] `https://paperswithcode.com/datasets`

4.2.2.1 Italian and French datasets

We list monolingual corpora for Italian and French in Tables 4.1 and 4.2. Table 4.3 describes multilingual classification corpora containing one or both of these languages, and possibly others. We mark with a single asterisk (*) datasets available through a request for access. Additionally, we mark with (**) datasets that are not distributed for free or require a specific affiliation. When defining tasks, we use the abbreviations introduced in the previous section and otherwise use TC to indicate a generic "text classification" task that does not fit any of the defined categories. For the sake of comparison, we give an estimate of the dataset size, based on the published documentation. Size can be expressed as the number of labeled sentences ("S"), tokens or words ("T"), or documents ("D") available in the corpus, and is comprehensive of all training, test, and evaluation splits. For multilingual datasets in Table 4.3, it refers to the number of samples in Italian and French (or, when similarly sized, an average of the two).

Table 4.1: Italian datasets.

Name	Paper	Task	Size	Unit
ABSITA	[136]	SA	10,000	S
SENTIPOLC	[105]	SA (irony, subjectivity)	9,400	D
ATE_ABSITA	[283]	SA, TL	4,300	D
AMI 2020	[280]	SA (misogyny)	9,900	S
R-ITA	[289]	SA (stance)	1,000	D
ChroniclItaly	[274, 302]	NER, SA	8,600	D
IHSC	[145]	SA, SP	6,900	D
HaSpeeDe*	[281]	SA, SP	8,500/3,500	D
SQuAD-it	[162]	QA	61,000	D
Fact-Ita Bank*	[106]	NER, SP, NC	65,000	T
FLaIT*	[71]	NER, SP	1,500	S
PAISÀ	[76, 72]	SP	250,000,000	T
KIPoS*	[233]	SP	200,000	T
iLISTEN 2018	[137]	SP	22,000	T
PoSTWITA	[105]	SP	6,700	D
TUT	[28]	SP	3,500	S
TE-EVALITA 2009	[54]	NLI	800	D
GxG	[138]	TC (gender)	11,000	D
DaDoEval	[291]	TC (date)	2,800	D
AcCompl-It*	[282]	TC (acceptability, complexity)	1,680/2,530	S
ITAmoji*	[176]	TC (emoji prediction)	275,000	D

* Available through an access request.

Table 4.2: French datasets.

Name	Paper	Task	Size	Unit
French Twitter SA[1]	-	SA	1,500,000	D
Allociné	-	SA	200,000	D
French Sexism Detection	[243]	SA (sexism)	11,800	D
Event2018*	[247, 244]	SA (stance), TC (event)	15,000/137,000	S
CAS*	[261]	SP, SA (uncertainty, negation)	4,900	D
E-FRA	[289]	SA (stance)	2,000	D
FQuAD*	[265]	QA	26,000	D
PIAF	[245]	QA	3,800	D
Quaero Broadcast News XT**[2]	-	NER	1,500,000	T
Quaero Old Press XT**[3]	-	NER	1,300,000	T
La Recherche**[4]	-	SP	447,000	T
FTB*	[213]	SP	644,000	T
ParisParl[241]	-	SP, TC (affiliation)	203,000,000	T
French Corpus MWE	[46]	SP	166,000	T
French FraCaS	[242]	NLI	346	D

[1] https://www.kaggle.com/datasets/hbaflast/french-twitter-sentiment-analysis
[2] https://www.islrn.org/resources/074-668-446-920-0
[3] https://www.islrn.org/resources/864-217-681-552-4
[4] https://www.islrn.org/resources/798-363-116-656-4/
* Available through an access request.
** Require payment or specific affiliation.

4.2.2.2 English datasets

A comprehensive list of English TC datasets is provided by Li et al. [350]. To facilitate comparison, we report in Table 4.4 some of the most popular English datasets, along with their sizes and related tasks. Because of the large number of results, our search here is limited to popular datasets used within PapersWithCode's recent contributions.

4.2.3 Findings

Many of the Italian datasets listed in Table 4.1 were created for the EVALITA initiative, a periodical campaign organized by AILC for the evaluation of NLP and speech tools for the Italian language. The most recurrent tasks proposed in this initiative fall into the sentiment analysis and syntactic parsing domains. While most datasets assembled by participants in this initiative are made openly available and provide great value to the Italian NLP research, they are often small in comparison

Table 4.3: Multilingual datasets (M=millions, B=billions).

Name	Paper	Languages	Task	Size	Unit
Webis-CLS-10	[56]	Fr, En, +2	SA	69,000	D
Amazon Reviews ML	[266]	Fr, En, +4	SA	210,000	D
SemEval-2016 Task 5	[99]	Fr, En, +6	SA, TL (aspect)	2,400	S
Reuters Corpus Volume 2 (RCV2)	[33]	It, Fr, +11	NC	28,406/85,393	D
MLSUM	[269]	Fr +4	NC	425,000	D
KB Europeana Newspapers NER	[82]	Fr +3	NER	-	-
WikiAnn	[120]	It, Fr, +280	NER	7.5M	T
DAWT	[111]	It, Fr, En, +3	NER, EDL	1.5M	D
WikiNER	[67]	It, Fr, En, +6	NER, SP	260,000	S
NewsReader MEANTIME	[96]	It, En, +2	NER, SP, NC	15,000	T
Universal dependencies	[97]	It, Fr, En, +100	SP	~1M	T
XL-WiC	[268]	It, Fr, +1	SP	2,000/70,000	S
Aranea	[81]	It, Fr, +20	SP	120M/1.2B	T
PANACEA	[58]	It, Fr, +2	SP	-	-
MKQA	[329]	It, Fr, En, +23	QA	10,000	D
CLEF QA Test Suites**[1]	-	It, Fr, En, +7	QA	160,000	D
XLNI	[161]	Fr, En, +13	NLI	7,500	D

* Available through an access request.
** Require payment or specific affiliation.
[1] https://www.islrn.org/resources/394-993-527-034-7

to English datasets for the same task, and are always fewer in number. SemEval is a similar international workshop for evaluating semantic analysis systems [314]. Multilingual datasets provided for the proposed tasks tend to be small, and although access is provided, they are not easily accessible or usable outside the context of these initiatives.

We hereby discuss the availability of task-specific datasets as compared to analogous English counterparts (Table 4.4). One should note that some of the datasets presented, especially in French, are made available through the ELRA-ELDA catalog[6]. This resource requires an ELRA membership plan and/or the payment of a fee

[6] Available at http://www.elra.info/en

Table 4.4: English datasets.

Name	Languages	Task	Size	Unit	Reference
20 Newsgroup	En	NC	18,800	D	[23, 350]
Reuters	En	NC	10,700	D	[18, 350]
R8	En	NC	7,600	D	[350]
R52	En	NC	9,100	D	[350]
RCV1	En	NC	804,000	D	[33]
AG News	En	NC	127,600	D	[38]
TREC-6	En	QA	5,500	D	[160]
SQuAD 2.0	En	QA	150,000	D	[155]
Yahoo!Answers	En	QA	1,460,000	D	[91]
Yelp-2 / Yelp-5 [4]	En	SA	8,600,000	D	-
Amazon-5	En	SA	3,650,000	D	[350]
Amazon-2	En	SA	4,000,000	D	[350]
IMDb	En	SA	50,000	D	[59]
DBpedia[5]	En +124	TL	630,000	D	[350]
MultiNLI	En	NLI	433,000	D	[147]
CoNLL-2003	En +1	NER	301,000	T	[32]

in order to be accessed.

Syntactic parsing Corpora for syntactic parsing (SP), like PoS-tagging and lemmatization, are well-resourced in both languages reviewed in this chapter and are featured in several multilingual treebanks (like Universal Dependencies [97] and Panacea [58]). Furthermore, the PAISÀ corpus stands out as a large monolingual dataset in Italian for these tasks.

News classification On the other hand, we noticed a lack of news classification (NC) datasets in Italian and French. The only notable exceptions are the MLSUM and RCV2 datasets. The MLSUM multilingual dataset contains news extracts labeled with their summaries and topics, and it is available in French but not Italian. On the other hand, the Reuters RCV2 dataset for multilingual news classification contains both Italian and French sub-corpora. This dataset can only be obtained by sending an access request and signing an organizational agreement.

Topic labeling Likewise, topic labeling (TL) datasets are also scarce in Italian and French. Wikipedia dumps and DBpedia represent a remarkable source of crowdsourced semi-structured data that can be employed for topic labeling and TC in general, and are available in hundreds of languages. However, labels must first be extracted and merged following some criteria that have not been standardized. Though these corpora have already seen use in the literature [121, 150], there is no

consistent set of annotations to be used as a reference. While categories assigned to Wikipedia pages by contributors are often used as predictive targets, these frequently contain spurious or improper information that can be treated in many different ways, or should arguably be removed entirely.

Sentiment analysis Multiple sentiment analysis (SA) datasets are available in Italian and French — often targeting user-generated content — for the detection of polarity, political stance, or specific rhetorical devices (like irony). More than 1/3 of the Italian datasets are scraped from social or e-commerce platforms, especially Twitter.

Question answering There is at least one large question answering (QA) dataset for both Italian and French, as well as several multilingual ones that contain both languages. For this task, the size of these datasets is comparable to the main English QA datasets.

Named entity recognition Similarly, there are multiple large multilingual corpora for named entity recognition (NER) tasks, at least for the most classic formulation of this task aimed at recognizing "person", "location", and "organization" entities.

Semantic entailment We found two semantic entailment (NLI) datasets containing the Italian and French languages, the largest containing 1,000 and 7,500 samples, respectively. By comparison, one of the most popular English NLI datasets (MultiNLI) is more than 50 times the size of the French one mentioned.

4.2.4 Cross-Lingual Benchmarks

Multitask evaluation benchmarks like GLUE, SUPERGLUE (for English), and FLUE (for French) are increasingly popular tools to evaluate models across a wide variety of NLP tasks. This incentivizes the development of models that share knowledge across different tasks and gain sufficient language understanding to generalize to a wide range of applications. Notably, the recent publication of the XTREME and XGLUE benchmarks introduced support for multilingual and cross-lingual cross-task evaluation. Still, for some tasks, not all languages are available. For example, among the classification tasks we are interested in, XGLUE supports only PoS-tagging (included in our SP category) and web page ranking in the Italian language. Additionally, they do not provide training data in every language, and, for some tasks, data is extracted from other multilingual datasets (namely XNLI, Universal Dependencies, and WikiAnn). Table 4.5 summarizes popular language-specific and cross-lingual benchmarks.

The goal of these initiatives is not to provide more monolingual data for languages with fewer resources, but rather to encourage the evaluation of cross-lingual models capable of transferring knowledge across different languages, even those with little or no training data. Their contribution is important in that it provides a standardized evaluation environment for DL models that could alleviate the common low-resource issue [267].

Table 4.5: Linguistic benchmarks.

Benchmark	Paper	Languages	Tasks	Cross-Lingual
GLUE	[164]	En	QA, SA, TL, NC	
SUPERGLUE	[210]	En	TC, NLI, QA	
FLUE	[246]	Fr	TC, SP, NLI, PAR*	
XTREME	[253]	It, Fr, En, +37	NER, SP, NC, QA	✓
XGLUE	[267]	It, Fr, En, +8	NER, SP, NC, QA	✓

* Paraphrasing.

4.3 Applicability Evaluation

We have previously mentioned the rising computational costs of developing state-of-the-art NLP solutions. In this section, we simulate a practical case by synthesizing two custom multilabel classification datasets. In our research, we have found that multilabel TC in the Italian language (and, to some degree, in French) is understudied, in contrast to binary and multiclass TC. In a similar vein to Tamburini [301], our study is aimed at gauging how easily and how well these methods can apply to new tasks and datasets with a constrained amount of resources (i.e., only fine-tuning).

4.3.1 Data

We use one monolingual dataset per language for each studied task. We decided to tackle the multilabel classification task, as it was more interesting for our research work and, to some extent, is less documented in the literature. An empirical evaluation of the labels in our TL dataset finds that the categorizations utilized are overlapping and cannot be easily decomposed into binary sub-classification tasks (e.g., “sports” vs “winter sports”) [30]. A similar consideration can be made of the selected NC dataset, which was multilabel by construction.

We chose to synthesize these datasets mainly because of the scarcity of other options in Italian. In the first case, we found that synthesizing a TL dataset from Wikipedia was the only way of obtaining a reasonably large, general-domain, an-

notated corpus in Italian. Similarly, the RCV2 dataset was chosen for the NC task because no other public collection of annotated news articles was available in Italian.

4.3.1.1 Topic labeling

We synthesized a dataset for the TL task using Wikipedia dumps in all three languages. For each dump, articles and related topics are extracted using a modified version of the popular WikiExtractor tool[7]. After an exploration of the data, we concluded that Wikipedia categories are ill-suited for a TL task since they are often too specific and hardly provide a good topic indication. Therefore, we decided to use a different approach and annotate extracted articles with the Wikipedia portals[8] to which they are assigned.

At the time of writing, there are roughly 500 portals within the English version of Wikipedia, while there are more than 500,000 categories. Wikipedia itself states that portals serve as entry points for articles that belong to the same broad subject, thus making them better targets for our task. Our final datasets contain only the 100 most populous portals, and article frequency has been limited to a maximum of 50,000 articles per label. This was done both to contain the dataset size and to reduce class imbalance.

4.3.1.2 News classification

For the NC task, we utilized the Italian and French subsets of articles in the Reuters multilingual collection (RCV2), as well as the English monolingual collection contained in RCV1. The English Reuters collection has been, for a long time, one of the staple TC corpora utilized for experimental purposes [30, 55], though RCV2, its multilingual version, has not seen as much attention. The articles are not "parallel", in the sense that they do not contain the same content in different languages, but are different articles altogether.

The RCV1/2 articles are labeled with a variety of tags that describe their content at varying degrees of specificity. The most consistent and interesting tags across languages were topic codes; within such codes, subjects are ordered hierarchically. However, articles are often tagged inconsistently — the depth of hierarchy within tags ranges between two and four levels, and documents are sometimes only partially tagged. For this reason, we decided to retain topics at the second level of specificity; each article is tagged with all second-level topic codes it contains and stripped of any others. Only topics that had at least 500 representatives were included in the final datasets. Articles are deprived of all topics excluded in this way, and the article is discarded if it contains no topics after this process.

[7] `https://github.com/attardi/wikiextractor`
[8] `https://en.wikipedia.org/wiki/Wikipedia:Portal`

4.3.1.3 Analysis of datasets

Final statistics of the described datasets are reported in Table 4.6. The main difference between the TL and NC datasets is the length of the articles; an average over the number of tokens per document reveals that Reuters articles are comparatively much shorter than extracted Wikipedia articles. Indeed, Wikipedia articles are usually much more descriptive, while Reuters articles are presented in a very concise and to-the-point format. As the LMs we worked with allow for a maximum of 512 tokens as input, we can expect a truncation to be much more frequent in the case of Wikipedia articles. The information loss should, however, not be dramatic, as most discriminative information is usually found at the beginning, where the article is introduced.

Figures 4.1 and 4.2 depict the distribution of the number of topics per article for the ItWiki-100 and RCV2it datasets, respectively. Unsurprisingly, most articles have few labels (between one and three), with a large amount having only one label. The larger Wiki datasets have a longer tail-shaped distribution, with a few outliers having a large number of labels, but that overall makes up a small part of the datasets (for instance, articles with four or more labels make up less than 1.4% of the ItWiki-100 dataset).

All datasets follow a similar distribution of topics, with many well-represented classes and a lower bracket of comparatively under-represented classes. We point out that class imbalance was much more severe in the raw data we processed, with a much smaller number of dominant classes and a much larger number of topics with next to no representation. As a result, the dataset remains unbalanced but makes for a realistic evaluation case, containing both prevalent and underrepresented topics.

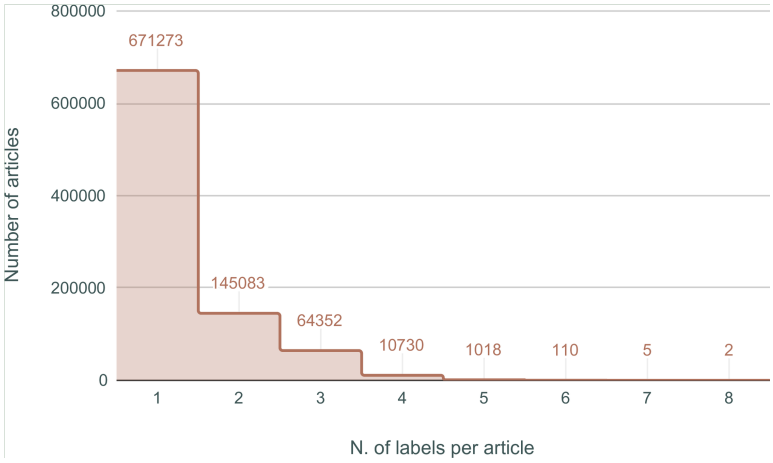


Figure 4.1: Distribution of the number of labels in ItWiki-100.

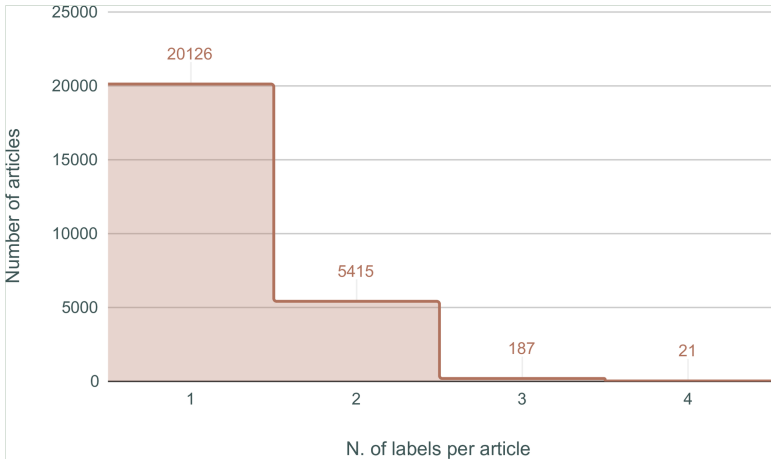


Figure 4.2: Distribution of the number of labels in RCV2it.

Table 4.6: Statistics on the examined datasets.

Name	Classes	Avg n. tokens	Samples	Task
ItWiki-100	100	354	892,573	TL
FrWiki-100	100	362	1,494,761	TL
EnWiki-100	100	741	329,626	TL
RCV2it	15	123	25,750	NC
RCV2fr	38	224	79,173	NC
RCV1en	57	216	758,149	NC

4.3.2 Experimental Setup

For all methods, excluding FastText and classical approaches, the input documents are truncated, keeping only the first 512 tokens (or padded to that size). For training, every dataset is split into a training, validation, and test set: 40% of the data is reserved for testing, and 20% of the remaining samples are used for validation. Splits are produced in a way to preserve the distribution of labels, through a stratification strategy [60, 123].

4.3.2.1 Evaluation metrics

One of the most adopted evaluation metrics for multilabel classification tasks is that of F_1 score, defined as the harmonic mean of precision and recall, as in the equation below (see § 2.2.3 in Chapter 2 for more details on evaluation metrics).

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.1}$$

In our tests, we report the F_1 score with macro-averaging across all categories

along with the accuracy score, as the latter is an interpretable measure of the overall fraction of correct predictions. In its computation, the predicted set of labels must exactly match the ground truth for it to be considered a true positive (this score is sometimes referred to as "*subset accuracy*").

Hierarchical metrics Given that the RCV1 dataset uses hierarchically organized labels (topical news codes), we additionally report model performance on the original, unaltered dataset using hierarchical metrics. Specifically, we adopt the approach of Kiritchenko et al. [40], which augments the sets of predicted and true labels to include all their respective ancestors. Standard evaluation metrics are then computed on these newly expanded sets. We also report the Average Hierarchical Cost (AHC), which measures the average distance between predicted and correct labels in the hierarchy tree. The theoretical basis for these metrics was discussed in detail in § 2.2.3.

4.3.2.2 Methods applied

We present quantitative results for an array of models that either are or have been state-of-the-art approaches to solving the task of TC. We start by providing a strong baseline with classical methods, of which we test Naïve Bayes and linear Support Vector Machine (SVM) classifiers as representatives. As examples of neural networks preceding the Transformer era, we showcase the results of FastText [110], XML-CNN [121], and a BiLSTM-based classifier. We then fine-tuned BERT [192] and XLM-R [251], two open-sourced Transformer-based methods that have been pre-trained on language modeling tasks over large corpora. This last set of methods currently achieves the best results on the vast majority of downstream NLP tasks.

Every method is trained and tested 4 times per dataset, each time on a newly generated train and test split, and we report the final average test metrics, along with the standard deviation (SD) over all runs, in Tables 4.7 and 4.8. As an exception, and because of technical constraints, we train XLM-R only once per dataset, as this model is much heavier than the already expensive BERT.

Hierarchical evaluation As a baseline experiment to quantify the difficulty of the original RCV1 dataset with respect to our simplified version, we assess model performance on the full, hierarchically organized set of 103 labels. For this task, we only evaluate the BERT and XML-CNN models alongside a linear SVM baseline configured in a multilabel (OVA) setting. All models are assessed using both flat and hierarchical metrics. To ensure a fair comparison, they were trained using the same cross-validation strategy, optimization method, and preprocessing pipeline. The results of this evaluation are presented in Table 4.9.

Table 4.7: Test set macro F_1 score (%) for the tested TC methods ± standard deviation (SD), with the best results in bold.

	Italian		French		English	
Model	**ItWiki**	**RCV2it**	**FrWiki**	**RCV2fr**	**EnWiki**	**RCV1en**
Naïve Bayes	62.0 ± 0.0	76.5 ± 0.4	55.1 ± 0.1	66.1 ± 0.3	63.6 ± 0.1	56.3 ± 1.4
Linear SVM	82.4 ± 0.0	79.6 ± 0.8	73.7 ± 0.0	72.4 ± 0.3	80.3 ± 0.1	71.7 ± 0.3
FastText	81.5 ± 0.1	76.7 ± 0.7	75.7 ± 0.1	64.1 ± 0.7	74.4 ± 5.4	69.6 ± 0.8
BiLSTM (GloVe)	83.6 ± 0.1	80.5 ± 0.2	76.9 ± 0.1	70.0 ± 1.4	81.2 ± 0.5	76.6 ± 0.7
XML-CNN (FastText)	82.7 ± 0.1	77.6 ± 0.9	78.9 ± 0.2	66.9 ± 1.1	78.2 ± 0.4	73.0 ± 0.7
BERT (base)	**87.0** ± 0.1	**84.0** ± 0.6	**84.0** ± 0.1	**76.8** ± 0.5	**85.5** ± 0.2	**78.1** ± 0.4
XLM-R (base)	86.8	83.6	83.2	73.9	84.6	77.2

Table 4.8: Test set subset accuracy score (%) for the tested TC methods ± SD, with the best results in bold.

	Italian		French		English	
Model	**ItWiki**	**RCV2it**	**FrWiki**	**RCV2fr**	**EnWiki**	**RCV1en**
Naïve Bayes	43.2 ± 0.1	62.9 ± 0.2	28.7 ± 0.1	47.5 ± 1.8	39.2 ± 0.7	47.3 ± 1.9
Linear SVM	74.4 ± 0.1	71.7 ± 0.5	58.7 ± 0.1	65.6 ± 0.3	66.9 ± 0.1	67.7 ± 0.2
FastText	74.1 ± 0.2	67.8 ± 0.5	60.3 ± 0.1	61.1 ± 0.6	68.2 ± 5.3	67.0 ± 0.4
BiLSTM (GloVe)	76.3 ± 0.2	72.7 ± 0.7	63.7 ± 0.1	65.7 ± 0.8	68.0 ± 0.9	72.5 ± 0.5
XML-CNN (FastText)	76.4 ± 0.2	71.2 ± 0.5	66.1 ± 0.3	64.4 ± 0.8	66.6 ± 0.5	71.0 ± 0.1
BERT (base)	**80.8** ± 0.2	**77.3** ± 0.6	**72.4** ± 0.2	**69.6** ± 0.7	**75.3** ± 0.3	73.5 ± 0.2
XLM-R (base)	**80.8**	**77.3**	71.6	68.8	74.3	**74.0**

Table 4.9: Performance (%) on the RCV1 original dataset ± SD, with the best results in bold.

Model	**Acc** ↑	**F_1** ↑	**h-F_1** ↑	**AHC** ↓
BERT (base)	**63.9** ± 0.4	**67.2** ± 3.1	**67.2** ± 3.0	19.6 ± 4.8
XML-CNN	55.2 ± 0.2	49.2 ± 1.2	49.3 ± 1.3	**18.2** ± 1.0
Linear SVM (MultiL)*	49.7	54.6	54.7	23.4

* SVMs run on fixed splits have deterministic results if run for enough iterations.

4.3.3 Discussion on Results

The results obtained for the classical methods are quite favorable. The one-vs-rest ensemble of linear SVCs proved to be the strongest baseline, while our application of multinomial Naïve Bayes had substantially lower performance (though it was considerably faster). Among preprocessing operations, we found that lemmatization and n-gram discovery did not produce significant differences in results, so we do not report their effect in the final tables.

Neural networks that predate the Transformer era demonstrated strong performance, usually surpassing traditional methods. In our experiments, the exceptions were likely attributable to the size of the training data; NNs had better results for larger datasets, giving ground to classical methods whenever training samples were scarcer. Moreover, on smaller datasets (like RCV2it), we observed a noticeable variance between the results of different runs, whereas the networks were very consistent on larger datasets (like our "Wiki-100" datasets).

For these models, we experimented with different pre-trained embeddings (i.e., Word2Vec, GloVe, and FastText) and found that FastText embeddings yielded the best result for XML-CNN. In the case of BiLSTMs, however, the best results were instead obtained with GloVe embeddings, despite their comparatively restricted vocabulary size. Furthermore, BiLSTMs benefited from the removal of stopwords in their input text, something that we did not find to be true for the XML-CNN. Nonetheless, the gap in the results between the two models was noticeable but not dramatic.

Unsurprisingly, the attention-based Transformer architectures outperformed the other methods on all our datasets. An important aspect of these models that is worth reiterating is their ease of application to downstream tasks. Only a few epochs of fine-tuning were necessary to obtain these results, though they were still the most computationally complex and required the longest time to train. While monolingual BERT models performed best, XLM-R also showed very strong performance, even though it is a multilingual model whose vocabulary capacity is shared across many languages. On average, we observe that the Italian models perform at the same level as (or slightly better than) the English models. The French tasks, on the other hand, proved to be slightly harder on both datasets. The similarity in results is not at all surprising, considering how close the three chosen languages are. Indeed, English has Germanic roots, while Italian and French are Romance languages (derived from Vulgar Latin), yet they have developed very closely and have strong influences on each other. Many differences could be pointed out (gendered nouns and pronouns, liaisons, accents, etc.), but it is fair to consider them morphologically similar languages since they all belong to the fusional family.

Finally, the evaluation on the original hierarchical RCV1 dataset, although limited to a smaller set of models, shows a significant drop in performance for all methods when compared to their results on our simplified, flattened version. As previously explained, our version of the dataset retains only the 57 second-level

labels from the original 103, favoring fewer but more consistent annotations. As the results in Table 4.9 indicate, predicting the full, original set of labels is a more challenging task. BERT's performance suffered a 13% drop in both accuracy and macro F_1, with an even more substantial drop for the XML-CNN and SVM baselines (over 22%). This outcome is not unexpected, as our initial analysis revealed the challenging and highly specific nature of many of the original labels, which motivated the dataset simplification. Note that in this more complex setting, methods designed specifically to handle label hierarchies would likely provide better results. The hierarchical F_1 score did not provide significant additional insight, as these scores did not differ substantially from the standard flat metrics in our experiments. In contrast, the AHC metric reveals how distant the incorrect predictions were within the label hierarchy. The high AHC scores show that the models' predictions were, on average, significantly far from the true labels in the hierarchy tree, further confirming the difficulty of the task.

This analysis demonstrates the ease of application and strong performance of pre-trained LMs, including multilingual variants like XLM-R, in settings with fewer resources and without extensive, task-specific tuning.

4.4 Limitations and Opportunities

While this study demonstrates the practical effectiveness of pre-trained LMs on non-English tasks, it has several limitations, which we address here to highlight valuable opportunities for future research.

The primary limitation is the presence of confounding variables that make it difficult to isolate the impact of language alone on performance. The monolingual models used in our benchmark differ significantly in their pre-training corpora, architectural parameters, and training objectives. For instance, the Italian BERT was pre-trained on 81GB (13B tokens) of data, taken from OPUS and OSCAR corpora[9]. The English BERT model is trained on the BookCorpus and Wikipedia dump (13GB) [192]. The French model FlauBERT is trained on a mixture of French documents, extracted from Project Gutenberg (a collection of e-books), OPUS, Wikimedia, and Common Crawl, amounting to 71 GB of data [246]. This latter model is also trained with a MLM objective only, while the others use both MLM and NSP, and it has more learnable parameters: 138 million instead of 110. Finally, XLM-R was trained on 2.5TB of data in 100 languages, extracted from the Common Crawl corpus [251]. In it, the amount of data per language is variable: 300 GB for English, 30 GB for Italian, and 57 GB for French text. Furthermore, the classification datasets themselves vary in size and the number of target labels. For example, the RCV2 dataset contains a relatively small number of Italian articles when compared to both French and English. The number of target labels is also variable across languages, resulting in TC tasks that are likely to be on a slightly different level of difficulty. Given

[9] `https://github.com/dbmdz/berts`

these differences, our results should be interpreted not as a definitive statement on the inherent difficulty of these languages for TC, but as a pragmatic evaluation of how a selection of publicly available LMs perform on realistic, monolingual tasks across three languages with varying levels of resource availability. Future work could ascertain the role of language and morphology in these LMs. Considering that tokenization is the most language-dependent step, such a study would need to control for confounding factors by testing several tokenizers and pre-training LMs from scratch on balanced monolingual corpora, similar to the methodology of Rust et al. [315].

We also acknowledge potential limitations in our dataset creation. It is possible that parts of the Wikipedia dumps used to create our datasets were also present in the pre-training corpora of the English and French models, which could have led to some degree of information leakage. Furthermore, our dataset curation involved several study-specific decisions, such as discarding categories based on our initial exploration. While this was a pragmatic step for our analysis, we acknowledge that future work should prioritize the use of standardized benchmarks for these tasks, such as those in the GLUE family [164].

4.5 Conclusion

In this chapter, we provided an overview of existing models for TC and studied their applicability to non-English languages, using Italian as our main point of reference. Our analysis began with a survey of existing TC datasets across three languages, which we categorized according to the tasks they address. We compared the availability and scale of Italian TC datasets to equivalent corpora in French and English, confirming that while both benefit from a greater availability of large, labeled datasets, English remains significantly better-resourced than either. Furthermore, we presented new quantitative results on multilabel news classification and topic labeling tasks in Italian, French, and English, domains that are underrepresented outside of English-centric research.

Building on this benchmark, the next chapter will apply this knowledge to a more complex problem. While the datasets in this chapter were treated as having flat labels with no explicit relation to one another, a key observation is that they are derived from inherent category hierarchies, which are very common in real-world scenarios. Our preliminary results on the original RCV1 dataset further showed a significant performance drop when the full label hierarchy was considered. This presents an opportunity to investigate classification methods that can explicitly leverage such information. The following chapter will therefore address this challenge by applying Hierarchical TC to a specific industrial use case.

5

Hierarchical Classification for Support Ticket Automation

Modern service providers must often process large amounts of customer requests swiftly, yet few research contributions have approached this problem leveraging the latest advances in pre-trained LMs, and the field still largely relies on traditional classification methods. This chapter addresses this gap by proposing an LM-based approach for this task, demonstrating a practical application of HTC that leverages the hierarchical relations between ticket categories, a subject I focused on during the first part of my research [6, 4, 7]. We first provide an overview of support ticket automation strategies, focusing on recent DL proposals for ticket classification. We analyze commonly used datasets and conduct experiments on two public corpora characterized by a two-level hierarchy of labels. The first is a collection of 20,000 customer complaints, and the second comprises 35,000 issues crawled from a bug reporting website. Using this data, we focus on classifying tickets with a pre-trained BERT model. The experimental section has two primary objectives: first, to demonstrate the impact of different document representation strategies on performance, and second, to present an effective method for improving classification performance by injecting information from the label hierarchy into the classifier. Our findings reveal that the choice of embedding strategy is critical, improving the F_1 score by over 28% compared to the standard approach. Furthermore, injecting hierarchical information provides an additional performance gain, with our proposed ML-BERT model outperforming the best baseline by up to 5.7% in F_1 score and 5.4% in accuracy on the bug reports dataset.

5.1 Introduction

A *support ticket* is a formal request for help from a customer to a service provider's support team. These requests, which include service tickets, customer complaints, and incident reports, are fundamental tools for customer relationship management [327]. As one of the most direct points of contact between users and service staff, tickets are crucial for resolving issues and incidents. Such interactions are ubiquitous

across nearly every industry, from IT support and bug reporting [224] to healthcare [232] and governmental institutions [239].

Tickets are typically textual messages, often written directly by customers or technicians. Consequently, they consist mainly of natural language text that is frequently noisy and concise. They commonly originate from channels like emails, web forms, live chats, and social media platforms [346]. A ticket usually comprises a short title and a description of the issue, and may sometimes include supplementary data like screenshots [223]. Once a ticket is created, its swift categorization and routing to the appropriate expert are paramount for ensuring customer satisfaction, maintaining high productivity, and complying with Service-Level Agreements [66]. In this context, *Ticket Automation* (TA) refers to the set of automated systems designed to minimize the steps between ticket submission and resolution. Among TA tasks, accurately classifying incoming tickets with a descriptive label is one of the most widely studied and impactful [383].

5.1.1 Task and Approach

This chapter first discusses the most common TA sub-tasks in § 5.2, focusing on the most recent developments that have been applied to this field. The experimental part in § 5.3 then addresses the most common TA framing, which is the automatic categorization of a service request within a shallow hierarchy of topics. The task is therefore that of TC, which requires extracting meaningful representations from the text of the support tickets.

Our approach involves using a pre-trained BERT model [192] to learn semantically meaningful representations from the noisy text of support tickets. Specifically, we first explore how different document embedding strategies impact classification performance. We then devise and evaluate a specialized multi-level model that leverages the label hierarchy by combining representations fine-tuned to predict the different levels of that hierarchy. We validate this approach on two public, two-level hierarchical datasets: one containing customer complaints and another of bug reports.

5.1.2 Contributions

The primary contributions of this work are as follows:

- We provide an overview of Ticket Automation, including a categorized list of recent methods and a comprehensive list of public datasets;
- We demonstrate how different strategies for deriving document embeddings from a BERT model significantly impact performance on ticket classification;
- We propose a novel, global approach to this classification sub-task that exploits the hierarchical structure of the labels to improve performance;

- We benchmark our proposed model against several baselines on two public datasets, showing its effectiveness on noisy, real-world data.

Our work is among the first to experiment with a pre-trained Transformer-based LM for the classification of support tickets. Despite the noisy nature of the data, we show that these LMs can perform better than more traditional (i.e., non-DL) methods that are often still proposed in the recent TA literature. As such, we hope the insights provided in this work can help practitioners consider the usage of pre-trained LMs for industrial applications in the TA domain. The code and datasets used in our experiments are publicly available[1].

5.2 Ticket Automation

Automatic ticket classification can be seen as a specific field of application of TC [324, 293, 2]. The processing of support tickets is made challenging by the nature of the bodies of text involved: many of these help requests are very brief, and almost always contain technical jargon that should be taken into careful consideration [203].

In this section, we provide an overview of TA, describing its most prominent subtasks as well as listing notable and recent work in this field. We will put particular emphasis on ticket classification, as our research reveals it to be the most common automation procedure in practice. At the end of this section, we also provide a list of datasets often used in this domain's literature. First, however, we describe the similarities between TA and HTC, a subfield of TC.

5.2.1 Relation to Hierarchical Text Classification

It is common for ticket categories to have a *hierarchical structure*; these levels identify the incident, or help request, at different degrees of specificity. In practice, most real-world ticketing scenarios will have at most a shallow hierarchy of two or three levels [215].

In this context, HTC methods are a group of approaches especially devised to be applied to TC environments characterized by a hierarchical label structure. As such, these methods are indeed applicable to these scenarios, though most of these approaches are devised for more complex hierarchies and are often framed as multilabel tasks, which we found to be less common in TC. Nevertheless, many concepts within HTC literature are still useful for TC, and we introduced them in § 2.2 of Chapter 2. Specifically, our work includes a comparison of global HTC methods against local and flattened approaches, utilizing the taxonomy introduced in § 2.2.1.

[1] `https://gitlab.com/distration/dsi-nlp-publib/-/tree/main/ticket-automation-survey-app-22`

5.2.2 Ticket Automation Tasks

As previously mentioned, multiple automation procedures have been devised in the context of TA. In many cases, the straightforward "topical" classification of tickets is sufficient to assign tickets to a specialized group that can deal effectively with the issue. However, depending on the circumstances and resources available, more nuanced and refined techniques can also be applied, often solving more complex tasks. We briefly outline these tasks, though we will focus mostly on the former approach. Whenever appropriate, we will discuss methods within these classes if they provide useful insight into our objectives.

First off, ticket classification itself need not be limited to a topical categorization; some approaches attempt to capture facets other than the topic of a ticket, most commonly a *priority* to determine the urgency of the incident and a *type* which, in a related fashion, is utilized to determine its importance (e.g., information request *vs* incident report) [53, 171]. Other methods attempt instead to match tickets directly with an individual expert, rather than groups of experts based on topics (a task related to *expert finding*) [219]. While similar, this task is rendered more complex by the necessity of clearly defining the skills of the expert and the ones required to solve the issue, often also having to consider the availability of the expert in question. Some approaches seek to find the optimal (minimal) "routing" of tickets through the network of experts [50, 286]. While this might reduce to expert finding whenever only one expert is necessary, this is not always the case. In a medical scenario, for example, it is often important to gather a sequence of opinions from different specialists, therefore requiring a "path" traversing the network of experts. Lastly, in some cases, it might be possible to match tickets directly with their resolution without the need for human intervention [135]. There are various approaches to this latter task, the most common being either retrieval-based (i.e., match most suitable historical solution) or generative (i.e., generating an entirely new response, learning from previous ones). Table 5.1 provides a summary of the different automation procedures that may be applied in the context of support ticket resolution.

Table 5.1: Common TA frameworks applied to support tickets.

Task	**Application**
Ticket Classification (TiC)	Categorize tickets in terms of topic, sometimes also type and priority
Expert Finding (EF)	Automatically assign the resolving expert to a ticket
Ticket Routing (TR)	Direct a ticket through the shortest path in a network of experts
Ticket Resolution (RE)	Find an automatic resolution to a ticket, usually based on past solutions

5.2.3 Related Work

Most research studies in the context of ticket classification propose new methods or analyze their functioning within a particular domain. There are a few reviews and surveys that discuss the subject. Revina et al. [293], for instance, deal with tickets in the IT domain, exploring text representation techniques, and the performance of various text classifiers. However, they limit their review to more traditional classification methods, such as SVMs and Random Forests. They discuss the need for explainable TC, as well as which factors are relevant for prediction quality. Kubiak and Rass [170] discuss ticket classification as a part of a larger work on data-driven techniques for IT-Service-Management. They also discuss its relatedness to hierarchical classification and thoroughly discuss performance evaluation methods suitable for hierarchical approaches. Again, they mostly discuss traditional methods, such as SVMs, Bayesian models, k-NN methods, and Decision Trees. In Fuchs et al. [383], a literature review of technologies in the field of automated support ticket systems is provided. This review is largely aimed at automated ticket resolution, rather than classification. Young et al. [232] review TC procedures in the context of healthcare incident reporting and adverse event analysis. The authors list a wide selection of methods that have been utilized in the healthcare environment and discuss how they can be effectively utilized. The reviewed works mostly consist of traditional classification methods.

Among TA tasks, expert finding may be considered the most closely related to ticket classification, depending on the specific situation and the required ticket routing solution. A generic overview is provided by Husain et al. [219], which review work in the period 2010 – 2019. They describe the finding of experts for technical support as an early formulation of expert finding. In Xu and He [182], trouble ticket routing is framed as an expert recommendation task that also integrates TR components, by learning social profiles for the experts in order to suggest other experts in case the current one is unable to solve the issue. They devise several two-stage expert recommendation algorithms to determine appropriate resolvers for a ticket. They also argue that the more standard approach of classifying tickets within single problem types might be limiting, due to the negative impact of misclassifications. Lin et al. [124] review expert finding methods, focusing on the parts of this task involved in expertise resource selection (extracting expertise-related data for experts) and expertise modeling (building models on the data to identify experts). They examine state-of-the-art algorithms for expert identification up to 2015, by which they rank previously modeled experts based on the probability of them being an expert on a query topic.

5.2.4 Recent Ticket Automation Methods

As mentioned, TA refers to a set of algorithmic solutions that automatically process support tickets and customer requests. In this section, we first describe our study

selection procedure; then, we list recent advancements in Tables 5.2 to 5.4. The tables contain a high-level overview of these methods and the contributions they bring to the literature. Then, to later compare it with our proposal, we analyze in more detail the most relevant works we have found in the ticket classification field. Because of the large number of works, we apply strict constraints in determining their relevance; in particular, we require that an existing code implementation has been made available for a proposed method.

5.2.4.1 Study selection

We reviewed literature on TA using Google Scholar[2], DBLP[3], Scopus[4], Web of Science[5], and PapersWithCode[6]. We focused on research published from 2018 up to the most recent available in 2023, but also included influential earlier works when they were frequently referenced. The keywords queried were "*ticket automation*", "*ticket classification*", "*support ticket*", "*trouble ticket*", "*expert finding*" and "*ticket routing*". We found that a considerable number of matching results are articles regarding the application of machine learning algorithms to real-world ticketing systems, but do not provide new solutions or interesting research insights. Moreover, many of these works report the results of relatively few classification methods, most of which are traditional or otherwise not very recent or use paid API-based services. As they do not contribute to our goal, we have excluded them from our search. Much of the work found matching the keyword "expert finding" relates to applications of recommendation systems or ranking methods [3] as applied to several domains; we limit our selection to the methods tested on customer-care data. We summarize the results of our search in Tables 5.2 to 5.4, which list publications related to TiC, EF, and TR/RE, respectively. In these tables, the dataset is omitted for studies that used proprietary, unreleased data.

5.2.4.2 Description of recent methods

In the remainder of this section, we introduce some of the works contained in the previous tables. We focus in particular on those manuscripts that provide a public code implementation, as these can be helpful to other researchers to make comparisons and test baselines. However, as we will discuss, we were not able to reproduce all the methods selected this way.

Mani et al. [224] propose DeepTriage, a deep bidirectional RNN enhanced with the attention mechanism for ticket classification. An initial preprocessing step removes stopwords and tokenizes text through Stanford's NLTK package [39]. Em-

[2] `https://scholar.google.com`
[3] `https://dblp.org`
[4] `https://www.scopus.com/`
[5] `https://www.webofscience.com`
[6] `https://paperswithcode.com`

Table 5.2: Ticket Automation literature on the TiC sub-task.

Article	Data	Contribution	Code
[130]	-	Clustering and topic modeling methods for TC with BoW	✗
[157]	-	Ticket escalation prediction with oversampling, feature engineering based on domain knowledge	✗
[139]	-	Ticket partition and signature-based construction algorithm, problem type classification	✗
[172]	-	Ensemble (voting) classifier, bagging, and boosting	✗
[169]	-	Entity (software product name) extraction and linking for support ticket routing	✗
[171]	Chromium, Linux	Hierarchical Attention model, comparison with traditional classifiers	✓
[178]	-	TC modeled as a contextual multi-armed bandit problem	✗
[179]	-	Two novel multi-armed bandit algorithms	✗
[158]	-	Comparative analysis of five classifiers to label customer issues with predefined topics	✗
[223]	-	Enrichment of tickets with data from attached images	✗
[203]	-	HTC, ticket similarity method, multilinguality	✗
[232]	-	Systematic review of classification methods in the context of healthcare incident reports	✗
[221]	GitHub	Plugin app to automatically label GitHub issues through FastText, issues dataset	✓
[228]	-	Comparison of string matching and similarity measures	✗
[231]	-	Comparison of several algorithms for the prediction of ticket escalation based on the extracted sentiment	✗
[226]	-	Assignment scheme based on a Petri Net model	✗
[295]	-	Integration of several similarity measures through data-driven policy	✗
[293]	-	Comparative analysis of text representation techniques (TF-IDF, Word2Vec, FastText) and classifiers, linguistic feature extraction	n/a
[322]	-	Ticket opening prediction using time windows for feature engineering	✗
[333]	-	Binary bug classification, genetic algorithm optimization of MLP hyperparameters	✓
[343]	-	Fuzzy SVM to handle outliers and noise	✗
[346]	Endava	Oversampling of imbalanced classes, ensembles	✗
[340]	-	TC of German tickets, clustering to detect similar classes	✗
[330]	-	Learning char/word-level representation with bi-LSTM	✗
[335]	-	Prediction of ticket reopening, comparison of different methods, development of efficient BN	✗
[384]	AMS, LTI	Improve k-NN performance with distance function and determination of a suitable k value	✗

Table 5.3: Ticket Automation literature on the EF sub-task.

Article	Data	Contribution	Code
[224]	Chromium, Mozilla, Firefox	Bidirectional LSTM with attention, comparison with baselines, new datasets	✓
[219]	n/a	Review of main research on applications of EF	n/a
[198]	-	Ensembles of classifiers to assign tickets to sparse and frequent resolver groups	✗
[197]	StackExchange	Combination of question raiser, content, and answerer embeddings for EF	✗
[287]	-	Combination of graph- and text-matching to score tickets and expert groups	✗
[262]	-	Siamese LSTM to support classification of tickets with infrequent labels, RF with Gini impurity for explainable rule mining	✗
[328]	-	Combination of topic, reputation, and authority based on QA network	✗
[336]	StackOverflow	Intern candidates ranking using community QA data, ensemble method	✗
[303]	StackExchange	Usage of social information and document information for EF in community QA	✗
[382]	StackExchange	Learning of semantic relationships between words in answers and experts' skills	✓
[386]	StackExchange	Combination of experts' topic interests and an authority-based model to improve ranking	✗
[380]	LegalCQA	Creation of query-dependent user profiles using their answers and the comments to their answers	✓
[388]	StackExchange	Transformer-based multi-view encoders to learn the relation between experts and answered questions	✓

beddings for each word are then initialized using the Word2Vec algorithm [68] and fine-tuned for a few epochs. These word embeddings are then passed to a bidirectional LSTM, which acts as an encoder, effectively performing a language modeling task. The outputs of the LSTM layer are aggregated and weighted using the attention mechanism [92] to produce the final ticket representations, which are then classified using a linear layer with softmax activation. The cross-entropy loss is used during training. The method is validated on three public bug report datasets, which are extracted from the list of reported issues on the Chromium and Firefox browsers, as well as within Mozilla Core software components. To construct the dataset, the authors only consider fixed bugs, and the target label for each ticket is the developer ID who has resolved it.

Kallis et al. [221] propose TicketTagger, a GitHub plugin for the automatic

Table 5.4: Ticket Automation literature on the RE and TR sub-tasks.

Article	Data	Contribution	Code	Task
[135]	-	Mapping tickets to resolution actions using a quality-model estimation	✗	RE
[131]	n/a	Framework for the evaluation of TR and human-assisted algorithms	✗	TR
[134]	-	Retrieval of tickets and solutions from noisy hardware tickets with recall- and precision-oriented classifiers	✗	RE
[182]	-	Combination of expertise profiles and social profiles from ticket resolution sequences	✗	RE
[183]	-	Routing models using textual descriptions and historical routing sequences	✗	TR
[180]	-	Resolution action suggestion with sentence alignment strategy	✗	RE
[230]	-	Combination of CNN for feature learning and traditional methods for the classification step	✗	RE
[286]	-	Ranking method for TR, group feature engineering	✗	TR,EF
[284]	-	Ensemble model for ticket resolution using supervised and unsupervised methods	✗	RE
[379]	-	Flexible framework for action suggestion, action extraction module	✓	RE
[312]	-	Multi-task model for expert support in problem resolution with multi-task training	✗	RE
[383]	n/a	Review of machine learning applications in Automated Ticketing Systems	n/a	TR,TiC

assignment of labels to GitHub issues. The tool uses the FastText [114] library to assign one of three categories based on the title and description of each issue. The labels, used by repository maintainers to organize open issues, can be either "*bug report*", "*enhancement*", or "*question*". The FastText classifier extracts n-grams from documents and learns representations fine-tuned on the target dataset. It then averages these embeddings and feeds them to a linear classifier with a softmax output to obtain the final probabilities for each label [110].

Lyubinets et al. [171] describe a hierarchical attention model, combining hierarchical RNNs with attention blocks. As a preprocessing step, the dataset is lowercased, cleaned of noisy words and stopwords, and tokenized using the NLTK package [39]. Then, a bidirectional layer with Gated Recurrent Units (GRUs) [80] is used to learn word representations based on the context of each word in the sentence (this is the word encoder module). An attention mechanism is used to weigh the contribution of each word to obtain a latent sentence representation. Subsequently, another encoder identical to the previous one is used at the sentence level, and attention is applied to

learn overall document embeddings. Finally, similar to the previous works, they use a linear layer with softmax activation for classification (in a multiclass setting). The authors validate this approach on the Linux Bugs dataset using the "Priority" and "Product" fields, and on the "Type" attribute on the Chromium dataset that was used by DeepTriage. Unfortunately, we were unable to reproduce this framework using the published code because of undocumented dependency issues.

5.2.5 Datasets Used in Research

Table 5.5 showcases datasets utilized in the works we have reviewed in this research. The table only reports openly available datasets, describing their size, whether they are multilabel and hierarchical, the generic automation task as previously defined, and the topical domain that describes their content. It is worth noting that many works in TA literature apply their methods to proprietary datasets, which are therefore not available.

Table 5.5: Public TA datasets referenced in the reviewed works.

Dataset		N. samples	Multilabel	Hierarchical	Task	Domain
Financial[1]		78,313	✓	✗	TiC	Finance
Endava[2]		48,549	✓	✓	TiC	Helpdesk
IT company	[116]	21,348	✓	✓	TiC	Helpdesk
GitHub	[221]	30,000	✗	✗	TiC	Issues/bugs
LTI[3]		48,543	✗	✗	TiC	Helpdesk
AMS[4]		12,810	✗	✗	TiC	Helpdesk
Chromium	[224]	383,104	✗	✗	EF	Issues/bugs
Mozilla	[224]	314,388	✗	✗	EF	Issues/bugs
Firefox	[224]	162,307	✗	✗	EF	Issues/bugs
Enron	[35]	500,000	✗	✗	MC	E-mail
StackExchange	[360]	(77GB)	✓	✗	TiC, EF	(QA) Coding
LegalCQA	[380]	9,897	✗	✗	EF	(QA) Legal
Linux	[171]	16,456	✓	✓	TiC	Issues/bugs
Bitext[5]		20,000	✗	✗	TiC	Helpdesk
StackOverflow	[168]	2,2M	✗	✗	EF	(QA) coding

[1] www.kaggle.com/datasets/venkatasubramanian/automatic-ticket-classification.
[2] https://github.com/karolzak/support-tickets-classification.
[3] https://github.com/umeshdeorukhkar/LTI_Citi_Hackathon_My_ML.
[4] https://www.kaggle.com/datasets/karthikcs1/amsticketdata.
[5] https://www.bitext.com/blog/free-customer-support-dataset

5.3 Classification Approach

In previous sections, we provided an overview of the automatic support ticket resolution landscape. In this section, we propose a novel method for the specific task of automated topical classification of tickets within shallow hierarchies. Our approach is based on pre-trained Transformer-based LMs, which are currently state-of-the-art in terms of text representation. In particular, while the LMs are tasked to extract a semantically meaningful representation, our main addition is a multi-level framework that exploits hierarchical information contained within the label structure to perform a more accurate classification. In addition, we wish to explore various alternatives proposed in the literature for the creation of unified document embeddings to be used for the purpose of classification. These experiments were essential to our investigation, as they allowed for much better results in practice. Moreover, they provide useful insights into the development of individual document embeddings in this specific domain.

The following section will introduce our experimental approach. We first detail the datasets utilized and the baselines implemented. Then, we discuss multiple strategies to combine word embeddings as derived from a BERT model. Note that, though our specific approaches utilize BERT as a basis, they are agnostic to any LM capable of producing embeddings for words in a document. We conclude the section by describing our proposed approaches in more detail.

5.3.1 Data

We evaluated our methods on two datasets. The first, which we refer to as Linux Bugs, is a custom collection we created by extending the scraping script from Lyubinets et al. [171]. The second is a public Financial dataset of customer complaints available on Kaggle[7]. General statistics for the datasets, both before and after preprocessing, can be found in Table 5.6. A sample of the label hierarchies is shown in Figures 5.1 and 5.2. Both datasets adhere to the Mandatory Leaf Node Prediction (MLNP) setting. This means that the ground truth for each ticket consists of a single leaf-level label along with all of its parent labels up to the root of the hierarchy.

Financial dataset The Financial dataset contains 78,313 anonymized customer complaints from a financial company, which are essentially support tickets, although only 21,071 of these have a valid message written in natural language. All tickets in the dataset have been annotated by customers and/or helpdesk personnel with a "Product" and a "Service" category. While the original dataset has been used for topic modeling tasks based on the products/services of the tickets, we use these labels as the prediction target of our models. We utilize the product field as the main label (e.g., "*Debt collection*", "*Mortgage*", etc.), and the sub-product field as a sub-label (e.g.,

[7] `https://www.kaggle.com/datasets/venkatasubramanian/automatic-ticket-classification`

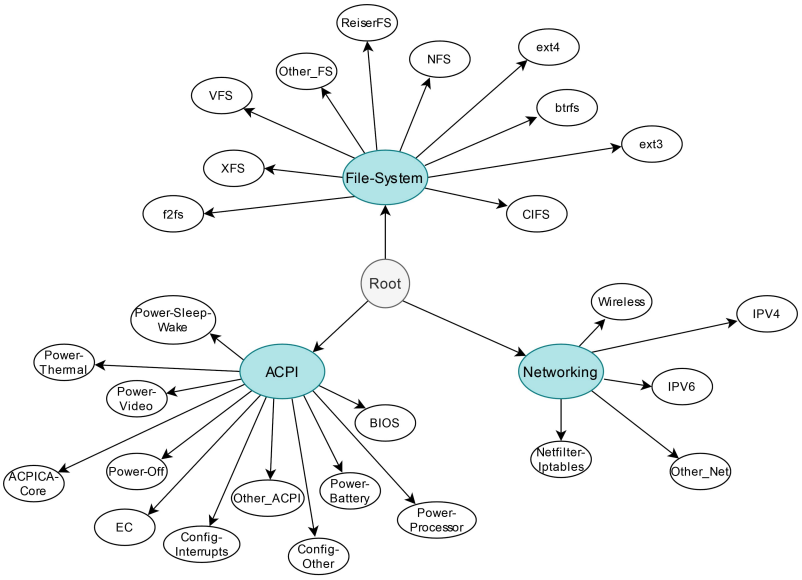


Figure 5.1: Example of hierarchical structure of 3 labels in the Linux Bugs dataset.

"*Credit card debt*", "*Checking account*", etc.). It is worth noting that the hierarchy of this dataset is rather weak, and we found the classification task to be hard in terms of predicting categories. There are multiple labels with similar or ambiguous meaning; for instance, there are pairs of labels such as "*Credit card*" and "*Credit card or prepaid card*", or even multiple similar categories such as "*Mortgage*", "*Mortgage debt*", "*FHA mortgage*", "*Other mortgage*", and "*Other type of mortgage*". Choosing the correct label would be non-trivial even for a human labeler. Below is an example

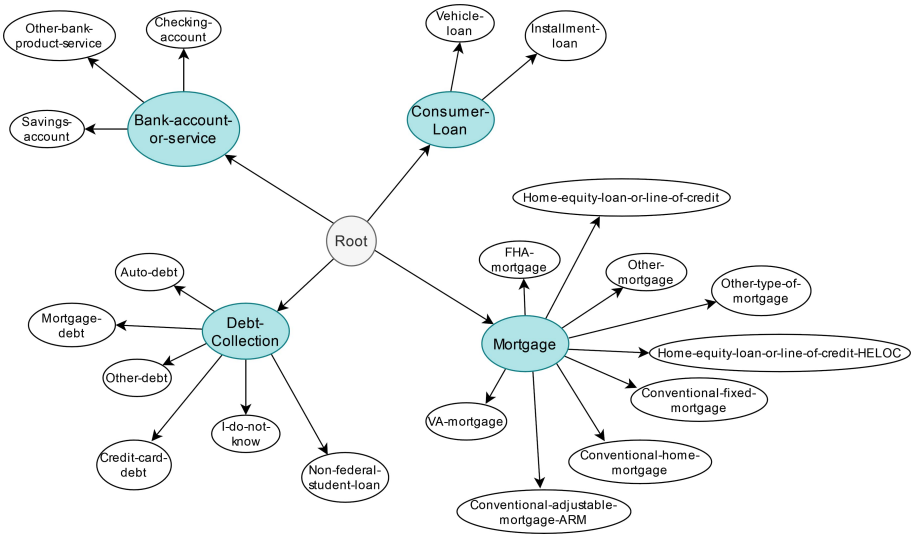


Figure 5.2: Example of hierarchical structure of 4 labels in the Financial dataset.

extracted from this dataset:

> **Product** (label): Vehicle loan or lease
> **Sub-product** (sub-label): Loan
> **Description** Vehicle was financed on [XXX] and last payment was [XXX] balance was [XXX]. Chase auto never release the Title of the vehicle. I called chase Auto finance division in year their answer was last payment was not clear. I asked them if they sent any written notice by mail if they did it send that but they never replied back.

Linux Bugs dataset The Linux Bugs dataset contains bugs from the Linux kernel bug-tracker [8], and was first proposed by Lyubinets et al. [171]. The one used in this work is an expanded version that we obtain by further crawling the online bug-tracking portal, and contains more than double the number of bug reports with respect to the original version. The support tickets are classified by users in terms of importance, related product, and specific component. We utilize the "Product" field as the main label (e.g., "*Network*", "*Drivers*", etc.), and the "Component" category as a sub-label (e.g., "*BIOS*", "*Scheduler*", etc.). Again, below is an example:

> **Product** (label): SCSI Drivers
> **Component** (sub-label): Other
> **Title**: MPT2SAS drops all HDDs when under high I/O
> **Description**: I have this issue that refused to be solved no matter what I do. My ASRock comes with onboard SAS controller (LSI 2308), since I recieved it always does this one thing: Drops all HDDs connected to it. It happens only under heavy IO operations after a few minutes. I can recreate it easily by running either dd, md5deep or even btrfs scrub. Kernel locks, can't even shut it down from console and a quick ls /dev/disk/by-id shows that all the HDDs connected to the SAS controller have disappeared. It happens with the stable kernel (3.9 and 3.10.3) and the mainline (3.11-rc2) as of this day. It's not a hardware issue, because I installed a Windows Server 2012 on the same machine with a few HDDs I have laying around and beat the controller to the ground and it never hanged. So I know it's a Linux-specific issue. Dmesg logs before and after the issue are attached. Thank you.

5.3.1.1 Preprocessing

We followed a process similar to the one adopted by Mani et al. [224] for the initial cleanup of both datasets. This preprocessing is aimed at removing noise and non-informative bits of text while maintaining sentence structure intact as much as possible. Indeed, this is necessary for recent, contextualized LMs to be effective [2].

[8] https://bugzilla.kernel.org

Table 5.6: Statistics for the datasets utilized in the experiments.

Dataset	Version	Samples	Avg chars	Labels	Sub-labels	Flattened labels
Linux Bugs	Original*	38,194	2,541	20	177	190
	Final**	35,050	2,536	17	73	85
Financial	Original	20,925	1,342	17	67	81
	Final	20,576	1,344	13	35	42

* After removal of tickets that are unlabeled, missing a message body, or duplicated.
** After the removal of rarest classes and the preprocessing operations shared across methods.

The raw datasets are filtered by removing duplicates and any entry where the main body of the ticket is void. Titles and descriptions are concatenated to generate a single content descriptor for tickets in the case of the Linux dataset (as Financial tickets have no title). Furthermore, to reduce the already severe imbalance within these datasets, we apply a threshold constraint: labels and sub-labels with fewer than a certain number of representative tickets are excluded. The threshold chosen for the comparatively smaller Financial dataset was 30, while it was 100 for the larger Linux dataset. Additionally, to disambiguate between second-level categories, which sometimes overlap in the Financial dataset, we concatenate the first level with the second level. For example, the bug report in the above example would be labeled as “SCSI-Drivers_Other”.

The DeepTriage-inspired [224] text sanitation procedure consists of the removal of URLs, some HEX codes, as well as a lowercasing of all words. Following the procedure from Lyubinets et al. [171], which initially proposed the Linux Bugs dataset, we also experimented with a more aggressive preprocessing approach to remove “garbage” text from Linux bug reports, which mainly consists of the removal of memory addresses. Though the filter works well, we did not register improvements whenever including this procedure, and therefore decided to exclude it from the finalized pipeline. We also note that traditional methods showcased an improvement when excluding stopwords from the list of processed tokens. The effect of our basic preprocessing can be visualized in the sample ticket below, which is the same as the one shown in the previous section, annotated with the label used as the target for classification. The size and statistics of the dataset we used in our experiments are summarized in Table 5.6.

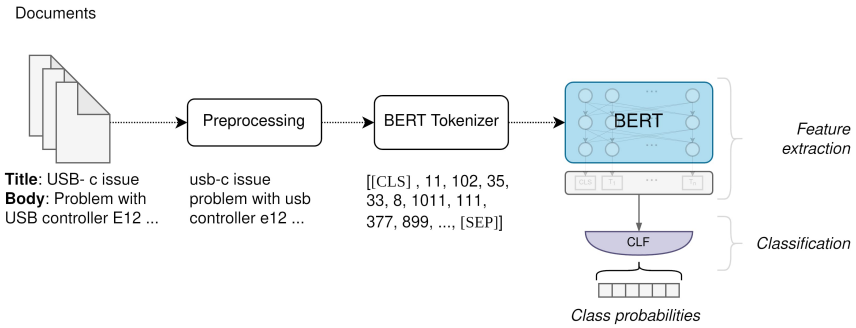


Figure 5.3: Tickets processing pipeline used with our BERT-based models.

Category (flattened label): SCSI-Drivers_Other
Ticket body: mpt2sas drops all hdds when under high i/o i have this issue that refused to be solved no matter what i do . my asrock comes with onboard sas controller (lsi 2308) , since i recieved it always does this one thing : drops all hdds connected to it . it happens only under heavy io operations after a few minutes . i can recreate it easily by running either dd , md5deep or even btrfs scrub . kernel locks , can't even shut it down from console and a quick ls /dev/disk/by-id shows that all the hdds connected to the sas controller have disappeared . it happens with the stable kernel (3.9 and 3.10.3) and the mainline (3.11-rc2) as of this day . it's not a hardware issue , because i installed a windows server 2012 on the same machine with a few hdds i have laying around and beat the controller to the ground and it never hanged . so i know it's a linux-specific issue . dmesg logs before and after the issue are attached . thank you .

After preprocessing, all models require a tokenization step. In the case of LMs like BERT, the tokenizer utilized was the one associated with the original work (i.e., WordPiece for BERT) [64]. In the case of DeepTriage and traditional methods, we utilized NLTK's *word_tokenize* function [39]. In short, this approach is an improved word-level tokenization based on regular expressions to split text as in Treebank-3 [27]. The overall classification pipeline for our experiments is visualized in Fig 5.3.

5.3.2 Baselines

In order to validate our results, we tested a set of baselines drawn from similar works that have obtained state-of-the-art results on the specific ticket classification task, as well as some broader state-of-the-art TC approaches. We report results for the following:

- **SVM** [21]: Similarly to Lyubinets et al. [171], we test an approach based on a TF-IDF [13] document representation, which is then fed to SVM classifiers with a linear kernel. The method utilizes a one-vs-rest approach, creating a classifier for each node of the hierarchy, but only on the flattened representation. The

best parameters are sought through a grid search prioritizing macro F_1, and are then utilized to retrain the model (test data is never seen during this procedure). As additional preprocessing, this method removes stopwords and seeks bi-grams within tokens.

- **TicketTagger/FastText** [221]: We follow the approach of the authors of TicketTagger, which is a straightforward application of FastText [114]. The classifier itself is quite simple and consists of a MLP optimized to utilize FastText's embeddings [110]. In our experiments, we use the "autotune" option provided, extracting a selection of 20% of the training samples for validation (as with other methods).

- **DeepTriage** [224]: The main architecture is based on a Deep Bidirectional Recurrent Neural Network (DBRNN-A), enriched with the attention mechanism and LSTM units. Text representation is achieved by creating fresh Word2Vec embeddings, first trained and then fine-tuned on the recurrent architecture. In their work, the goal is to assign tickets to available developers who can quickly resolve the bug, therefore resembling the EF task more closely. Nonetheless, as an (extreme) multiclass method developed in the same context, we applied it to our setting.

- **BERT** (*flattened*) [192]: This is a straightforward application of the BERT LM for sequence classification, which is based on the attachment of a classifier head on top of the LM. The classifier head is a single feed-forward layer, and the entire model is trained to predict the flattened, second-level labels. This is the standard approach to classification with BERT. We experimented with various document embedding summarization strategies, as discussed before.

- **XLNet** (*flattened*) [211]: The approach is the same as with BERT. XLNet is an autoregressive model, more akin to traditional LMs, but devises a clever pre-training approach based on word token permutations to introduce bidirectional information (i.e., both left and right context). We experimented with a few document embedding summarization strategies, but only displayed the results of the best-performing one, which utilizes the last token as a representative of the entire document.

5.3.2.1 Other methods considered

In addition to the ones reported, we also tested a number of other approaches. The results are not listed because the methods either obtained unsatisfying performances or we were not able to reproduce the same results as reported in the original work. Other traditional approaches, such as Naïve Bayes [113] and Random Forests [24], resulted in below-average performances. We were unable to run the code published by Lyubinets et al. [171], and thus were unable to verify their proposed method.

5.3.3 Document Embedding Summarization Strategies

Before a body of text can be embedded in any way, it must first be split into atomic units (words or parts thereof); this task is performed by specialized modules called *tokenizers*. In the case of BERT models, tokenization is based on the WordPiece algorithm [64, 192], which was discussed in Table 2.1. These algorithms operate on the assumption that common words should be kept in the vocabulary as-is, while rare words should be split, for a more significant representation of their composing segments to be learned.

A trained tokenizer truncates each input document to a certain threshold of maximum tokens as determined by the specific model, also padding any shorter sequences to the same length (in this case, 512). The BERT tokenizer additionally prepends to each input sequence a special symbol, the `[CLS]` token, which is expected to predict the target binary label of the Next Sentence Prediction task (NSP) during pre-training [192].

Tokenized text is presented as a sequence of ids, each one mapping to a word (or special symbol) in the BERT vocabulary — which is composed of about 30,500 English words and symbols in the HuggingFace model we adopted[9]. In its smallest version (*BERT-base*), BERT consists of twelve stacked encoder blocks, each one producing contextualized embeddings for input tokens, including the `[CLS]` token. Being pre-trained to capture sequence-wise information for a binary classification task, the `[CLS]` representation of the last layer is considered a good candidate to be used in a general classification task.

This was the strategy adopted by the authors of the BERT architecture, which also tested the model in a multiclass setting. In their work, the `[CLS]` embedding is passed through the NSP prediction head and to a final linear layer with softmax activation [192]. This strategy has been widely accepted as the "default" way to adapt BERT-like models to downstream classification tasks. However, some researchers suggest that other strategies may be preferred. Tanaka et al. [204] test several strategies to enrich BERT embeddings with a BoW feature vector, as well as averaging together word embeddings in the last and second-to-last layers; they report considerable improvements in classification over the default strategy. Similarly, Reimers and Gurevych [206] also argue that the `[CLS]` embedding makes for a suboptimal sentence representation, and propose SentenceBERT for the generation of sentence embeddings. Moreover, some researchers have suggested that the different encoder layers within the architecture can become "specialized" in the extraction of particular linguistic features, like syntactic and semantic ones, hence potentially providing additional information for classification [275, 196].

In this work, we test several strategies to obtain document embeddings from word embeddings and compare them with the standard approach using the `[CLS]` token. The tested strategies are described in detail in Table 5.7. Most of them are inspired by experiments in previous work [204, 331]. On the other hand, the

[9] `https://huggingface.co/bert-base-uncased`

sum_sum_norm and *sum_norm* strategies were devised following the intuition that embeddings could be considered oriented vectors in a high-dimensional space, and that the meaning of a document could be approximated by summing these vectors and normalizing the sum in different ways. Our findings will be outlined in § 5.4.

5.3.4 Multi-Level Models

Ticket classification datasets most commonly contain several labels organized in a hierarchical structure. For instance, the two ticket datasets used in this work are labeled with two levels of categories: a macro category and a secondary, more fine-grained topic indicator. These datasets are derived from a real-world ticketing system and bug report repository, respectively; thus, they provide a good indication of common structures within these environments.

In the following paragraphs, we will describe how the categorization of documents in this two-level hierarchical setting can be improved by combining models that work on different levels into a single *global model*. All models use pre-trained LMs (in this case, BERT) to extract features from each document's text. Classification is achieved by adding a single linear layer with softmax activation and fine-tuning the model, a widely utilized approach [192, 2]. Unless specified otherwise, we assume that BERT embeddings are also trainable during the fine-tuning procedure — again, as it is standard in these cases.

As a first step, we define two multiclass classification objectives with their respective set of target classes:

1. **T1**: prediction of first-level target class (i.e., the macro-label);

2. **T2**: prediction of second-level target class (i.e., the sub-label).

Like in the case of the flat classifiers, to avoid overlapping labels in T2, we flatten the tree of categories to obtain the final sub-labels. Through this process, any duplicate pair of sub-labels is transformed into two separate classes (e.g., if two labels both have the "*other*" sub-label). Notably, all sets of sub-labels are already disjoint, so this procedure does not increase the number of sub-labels.

We experiment with three multi-level classifier frameworks:

- The first one combines two classifiers that were previously trained on T1 and T2, respectively (`ML-LM`);

- The second one is trained on T2 and is supported by a classifier pre-trained on T1 (`SupportedLM`);

- The third classifier is similar in spirit to the first one but is trained end-to-end on both tasks with a single LM (`DoubleHeadLM`).

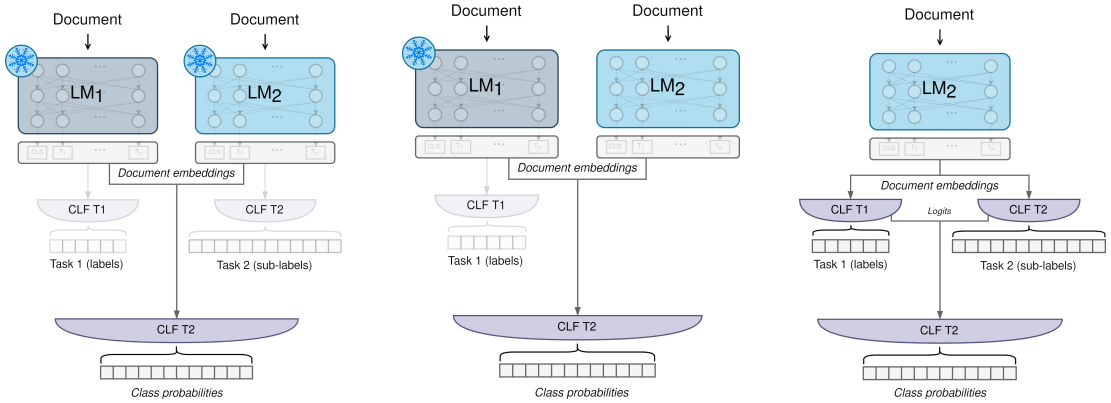
Table 5.7: Summarization strategies for document embeddings.

Basis	Strategy	Emb. size	Description
cls	*last*	d	`[CLS]` token embedding from last layer (default strategy)
	avg_h		Average of the `[CLS]` token embeddings from the last h layers
	$concat_h$	$d * h$	Concatenation of the `[CLS]` token embeddings from the last h layers
avg	*last*	d	Average of all token embeddings[1] from the last layer
	avg_h		Average of the average of token embeddings from the last h layers
	$concat_h$	$d * h$	Concatenation of the average of token embeddings from the last h layers
max	*last*	d	Column-wise maximum of all token embeddings[1] from the last layer
	avg_h		Average of the max of token embeddings from the last h layers
	$concat_h$	$d * h$	Concatenation of the max of token embeddings from the last h layers
max_min	*last*	$d * 2$	Concatenation of the max and min of embeddings from the last layer
	avg_h		As above, but averaging vectors from the last h layers
max_avg	*last*	$d * 2$	Concatenation of the max and average of embeddings from the last layer
	avg_h		As above, but averaging vectors from the last h layers
sum_sum_norm	*last*	d	Sum of token embeddings divided by the sum of the norm of embeddings
	$concat_h$	$d * h$	Like *last* but concatenating the last h layers
sum_norm	*last*	d	Sum of token embeddings divided by its norm (i.e., normalized sum)
	$concat_h$	$d * h$	Like *last* but concatenating the last h layers

[1] Excluding special symbols (e.g. `[CLS]` and padding).

5.3.4.1 Multi-level language model

The first Multi-Level LM, which we call `ML-LM`, is shown in Fig 5.4a. It utilizes two LMs. The first one is trained to predict the first level of labels (T1), while the second



(a) ML-LM. (b) SupportedLM. (c) DoubleHeadLM.

Figure 5.4: Proposed two-level classification models.

one is trained to predict second-level target classes (T2). Again, task T2 operates on a flattened representation to avoid sub-label duplicates.

After the models have been trained disjointly on the separate tasks, their respective weights are frozen. In `ML-LM`, the document embeddings obtained by these models (using one of the summarization strategies of Table 5.7) are concatenated together and fed to a linear layer, which is then trained on T2. To reiterate, at this point, only the parameters of the classification head are learnable, meaning computational costs are much more affordable. Note that the classifier outputs from both base models are discarded since we are interested in the pre-classification embeddings only.

5.3.4.2 Supported language model

The `SupportedLM` approach utilizes a LM previously fine-tuned on T1 as "support" to a secondary LM, which has not yet been fine-tuned. The second model is trained on the T2 task as before, but with additional information derived from the support model. The document embeddings from the two models are concatenated before the final classification layer. In this case, this step effectively adds extracted features from the first-level label, giving the second model a chance to rectify its prediction on T2 based on the feedback from the first model. The architecture of the model is showcased in Fig 5.4b.

5.3.4.3 Double-head language model

In the `DoubleHeadLM` approach, a single LM is used to produce document embeddings that are fed to two intermediate linear classifiers, one for each task. The two outputs of this prediction step are then concatenated and fed to a final linear classifier trained on the second task. The whole model is trained end-to-end, with three loss objectives, one for each classifier. The goal of this approach is to enforce a

regularization effect on the produced embeddings so that they better reflect all the information needed to predict both levels of labels. The final classifier combines the predictions of the intermediate classifiers to predict the correct labels. Fig 5.4c showcases the architecture of this model.

5.4 Experimental Results

In this section, we discuss the performance metrics and report the results. A thorough discussion of our findings and implications is provided in § 5.5.

Experiments are run on a machine with an Intel i9-9900K CPU, an Nvidia GeForce RTX 2080 Ti GPU, and 64 GB of RAM, with CUDA 10.1 and Python 3.10 used at runtime. The SVM algorithm is based on Scikit-learn's [63] linearSVC implementation, while DeepTriage is developed in Keras [93]. All contextualized language models (including BERT, XLNet, and their custom variations) are developed in PyTorch 1.11 [227].

To benchmark the selected methods, we utilize standard classification metrics, i.e., accuracy, precision, recall, and F_1 score, which are detailed in Chapter 2, § 2.2.3. The latter three metrics are *macro* averaged, meaning that all classes contribute to the average in the same manner (i.e., without weighing for class imbalance). We did not use hierarchical metrics in this analysis, as they did not provide significant additional information in our previous experiments.

5.4.1 Results

This section contains the results of our experiments. We compute metric values using two repetitions of 3-fold cross-validation: in every run, 33% of the data is used as a test set, and the remaining part is used for training. We select this specific number of folds to reduce the time needed to train and test all models, since the procedure is repeated twice for each one. This allows us to account for the variability of results and still contain the overall training time. Splits generated for testing or validation are sampled using stratification to ensure that labeled documents are selected in the same proportion as they appear in the whole dataset. All results reported in this section are obtained by averaging the measured metrics over all six runs. Before testing, we used 20% of the training split as a validation set to select the models' hyperparameters.

5.4.1.1 Document embedding summarization strategies results

Table 5.8 lists results obtained by training BERT-based classifiers on the T2 task with different document embedding summarization strategies. Again, before measuring performance on the test split, we used 20% of the training set to determine the optimal values for the following hyperparameters:

- Number of fine-tuning epochs;
- Learning rate (choosing between $5e^{-6}$, $1e^{-5}$, $2e^{-5}$, $5e^{-5}$);
- Whether to apply stronger preprocessing procedures;
- Whether to train the model with weighted cross-entropy.

The learning rate values experimented with were inspired by the ones used in the original BERT paper, scaling them to suit our reduced training batch size. We apply the same preprocessing used in DeepTriage [224], and we verify whether the additional cleaning operations proposed by Lyubinets et al. [171] are beneficial. We additionally try an approach that weighs each class's contribution to the cross-entropy value according to their support, so that less occurring classes contribute the most. Finally, to select the best number of epochs, we train using early stopping based on the loss value, with patience set to 2 epochs (as fine-tuning procedures usually run for very few epochs). We test all the combinations of the above-mentioned parameters on the validation set using a 3-fold CV. We validate using both the *cls last* and *avg last* strategies (among the ones listed in Table 5.7).

In both cases, the best results were achieved by training for no more than 3 epochs, with the learning rate set to $2e^{-5}$, unweighted loss, and no additional preprocessing. Therefore, we used these settings for all tests reported in Table 5.8. As the *cls concat*$_2$ strategy was the best among strategies that utilized multiple hidden layers, we further tested this strategy for varying values of h. As it turns out, the *cls concat*$_3$ strategy outperforms the previous one.

5.4.1.2 Multi-level models results

We present in Table 5.9 results obtained with the multi-level classifiers. We test these configurations using the "bert-base-uncased" model [192] available on HuggingFace [270]. Models are tested using only the best-performing averaging strategy, based on the results presented in the previous section.

As with the previous tests, we use the unweighted cross-entropy loss and make no additional preprocessing. The base LMs and respective classifiers utilized in the first and second architectures are first trained with the same hyperparameters chosen for the T2 task. When specified, the LMs then have their weights frozen for the remainder of the training process. To reiterate, these are LM_1 and LM_2 in the first architecture, while this applies only to LM_1 in the second architecture (Fig 5.4). We perform a separate hyperparameter tuning for learning rate and number of epochs for the final classifier of these architectures (i.e., the bottom CLF T2 in the figures), testing them as before on the validation set and utilizing a 3-fold CV. During tests, the models are trained for 2 epochs with learning rate set to $2e^{-5}$ for `SupportedLM` and `DoubleHeadLM` and $2e^{-4}$ for `ML-LM`.

Table 5.8: Test set results (%) with BERT classifier on T2 comparing summarization strategies on the Linux Bugs dataset (± SD). Best results are outlined in bold.

Basis	Strategy	Acc	F_1	Prec	Rec
cls	*last* (p)*	51.8 ± 0.6	35.4 ± 0.9	38.6 ± 0.9	36.5 ± 0.6
	last	56.6 ± 1.2	44.6 ± 1.8	47.9 ± 2.7	45.2 ± 1.7
	avg_2	53.1 ± 1.0	39.3 ± 1.2	42.0 ± 1.8	39.8 ± 1.2
	$concat_2$	53.5 ± 1.0	40.0 ± 1.4	42.6 ± 1.5	40.1 ± 1.2
	$concat_3$	**57.1** ± 0.8	45.6 ± 1.3	**49.8** ± 1.3	**45.8** ± 1.5
	$concat_4$	56.8 ± 0.9	**45.7** ± 1.4	49.0 ± 1.3	45.8 ± 1.7
	$concat_5$	56.5 ± 1.2	45.6 ± 1.3	48.6 ± 1.1	45.8 ± 1.8
	$concat_6$	56.0 ± 1.1	45.3 ± 1.5	48.6 ± 1.3	45.3 ± 1.7
	$concat_{10}$	56.2 ± 0.8	45.5 ± 1.4	48.5 ± 1.5	45.7 ± 1.6
avg	*last*	52.5 ± 0.8	38.7 ± 1.0	42.0 ± 1.5	38.8 ± 1.0
	avg_2	52.2 ± 0.5	38.3 ± 1.3	40.9 ± 2.1	38.7 ± 1.0
	$concat_2$	52.3 ± 0.7	39.0 ± 0.9	41.1 ± 1.5	39.4 ± 1.1
max	*last*	52.2 ± 1.1	38.5 ± 1.4	41.5 ± 1.1	39.1 ± 1.4
	avg_2	51.9 ± 0.7	37.5 ± 1.3	39.5 ± 2.1	38.7 ± 1.4
	$concat_2$	51.8 ± 0.6	37.3 ± 1.5	40.1 ± 2.1	38.3 ± 1.2
max_min	*last*	52.2 ± 1.0	37.7 ± 1.1	39.5 ± 1.9	39.5 ± 1.0
	avg_2	52.2 ± 0.9	37.4 ± 1.0	39.5 ± 1.9	38.5 ± 1.1
max_avg	*last*	51.6 ± 0.7	38.1 ± 1.2	40.6 ± 1.7	39.0 ± 1.2
	avg_2	51.9 ± 0.6	37.9 ± 0.7	40.2 ± 1.1	39.2 ± 0.7
sum_sum_norm	*last*	27.8 ± 1.8	7.0 ± 0.6	7.6 ± 0.6	8.7 ± 0.7
	$concat_2$	25.2 ± 3.2	5.0 ± 0.7	4.6 ± 0.7	7.1 ± 0.8
sum_norm	*last*	40.6 ± 1.8	17.1 ± 1.7	19.2 ± 2.3	20.6 ± 1.6
	$concat_2$	37.9 ± 1.3	13.5 ± 1.9	14.9 ± 2.2	17.9 ± 2.1
	$concat_5$	38.8 ± 1.5	13.5 ± 1.5	14.9 ± 1.5	18.0 ± 1.4

* Pooled, as in the standard approach for BERT classification (i.e. *cls pooled*).

5.4.1.3 Baselines results

We report in Table 5.10 the results obtained using the baselines outlined before. We perform a hyperparameter search for all neural models to select the best learning rate. For FastText, we report test results after performing the auto-tune procedure for 25 minutes on 20% of the training set. Before testing, the model is retrained on the entire training set with the best hyperparameters. XLNet is validated on the same learning rates used for BERT, and we use the same early-stopping strategy to select the number of epochs. We used the loss value as the target score in validation with neural networks, and we used the F_1 score for the other baselines. For XLNet, we utilize the last token as document representation, as suggested by the authors.

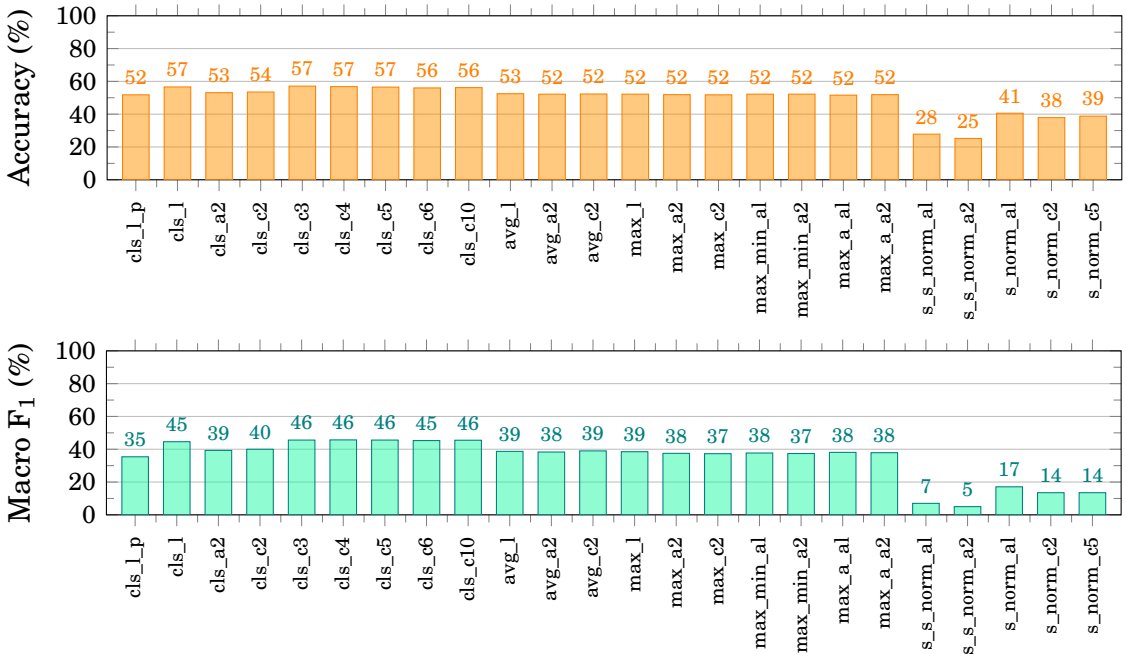


Figure 5.5: Visual comparison of the accuracy and macro F_1 score (%) of various approaches to document embedding summarization on the Linux Bugs dataset. Abbreviations: l = last, a = average, c = concat, s = sum.

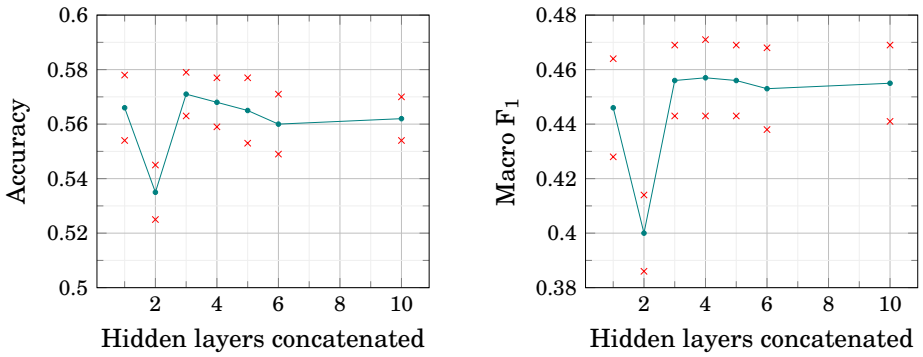


Figure 5.6: Test set results on the Linux Bugs dataset, utilizing BERT fine-tuned with a linear classifier. The graph represents the change in accuracy (left) and macro F_1 (right) scores with the number of hidden layers concatenated for the *cls_concat* strategy. In this case, 1 stands for the raw `[CLS]` token (not pooled). The "x" marks represent the upper and lower bounds given by the SD.

This strategy provides the best results on our task and is the one reported in this section.

5.5 Discussion

We summarize in Fig 5.7 the results of both baseline methods and our proposed approaches. The results demonstrate that our proposed methods can be more effective than baselines trained on the "flattened" version of the datasets. We observed an improvement on the Linux Bugs dataset, especially, and on the Financial dataset as well (though to a lesser extent), despite the latter not being strictly hierarchically labeled. Our findings confirm that models trained on the T2 task can

Table 5.9: Test set results (%) with multi-level classifiers on T2 with *cls concat*$_3$ averaging strategy (± SD). Best results are outlined in bold.

Model	**Acc**	**F$_1$**	**Prec**	**Rec**
		Linux Bugs		
ML-BERT	60.2 ± 1.0	**50.0** ± 1.4	51.8 ± 1.2	**50.1** ± 1.3
SupportedBERT	**61.1** ± 0.7	48.5 ± 1.3	**52.5** ± 1.0	48.7 ± 1.4
DoubleHeadBERT	54.9 ± 0.8	40.0 ± 1.3	44.2 ± 1.0	41.0 ± 1.4
		Financial		
ML-BERT	63.3 ± 2.9	**26.7** ± 2.0	**31.1** ± 2.7	**26.8** ± 1.9
SupportedBERT	**64.9** ± 2.1	24.9 ± 1.0	30.3 ± 1.9	25.1 ± 1.0
DoubleHeadBERT	57.6 ± 1.1	18.4 ± 0.9	22.1 ± 2.3	19.3 ± 1.0

Table 5.10: Test set results on baseline algorithms (± SD). Best results are outlined in bold.

Model	**Acc**	**F$_1$**	**Prec**	**Rec**
		Linux Bugs		
DeepTriage	51.6 ± 0.4	44.7 ± 0.6	49.3 ± 0.7	43.2 ± 0.6
SVM	55.1 ± 0.4	**47.3** ± 0.6	**55.2** ± 1.3	**45.9** ± 0.6
FastText	43.3 ± 0.3	29.1 ± 0.3	38.1 ± 1.4	28.1 ± 0.2
XLNet	55.6 ± 0.6	45.3 ± 0.6	48.0 ± 0.9	45.3 ± 0.1
BERT (*cls pooled*)	51.8 ± 0.6	35.4 ± 0.9	38.6 ± 0.9	36.5 ± 0.6
BERT (*cls concat*$_3$)	**57.1** ± 0.8	45.6 ± 1.3	49.8 ± 1.3	45.8 ± 1.5
		Financial		
DeepTriage	49.4 ± 2.7	20.3 ± 1.1	26.2 ± 2.3	19.4 ± 1.3
SVM	52.8 ± 1.0	25.4 ± 1.0	**36.4** ± 1.9	22.7 ± 0.7
FastText	54.2 ± 0.6	15.7 ± 2.2	20.2 ± 2.8	15.3 ± 1.9
XLNet	59.2 ± 2.1	**26.1** ± 3.0	31.7 ± 3.0	**25.8** ± 3.8
BERT (*cls pooled*)	59.1 ± 0.4	23.7 ± 1.1	26.4 ± 2.1	24.0 ± 1.2
BERT (*cls concat*$_3$)	**59.5** ± 0.9	24.0 ± 0.7	29.0 ± 3.2	24.2 ± 0.9

benefit from the integration of information from a model specialized in the T1 task.

5.5.1 Our Models

On the Linux Bugs dataset, our experiments show that `ML-BERT` and `SupportedBERT` respectively achieve 9.7% and 6.4% of F_1-score improvement over the flattened classifier with the best averaging strategy. The improvements in terms of accuracy instead amount to 5.4% (`ML-BERT`) and 7.0% (`SupportedBERT`). The results on the Financial dataset follow a similar trend; the two models achieve 11.3% and 3.8% improvement in F_1 score over the flattened classifier, while the accuracy score improves by 6.4% and 9.1%, respectively. Overall, `ML-BERT` and `SupportedBERT` achieve the highest F_1 and accuracy scores among all baselines, with a slight tendency by `ML-BERT` towards higher F_1 and, conversely, by `SupportedBERT` to favor accuracy.

The best baseline method in terms of accuracy is the cls_concat_3 flattened classifier. In terms of F_1 score, the SVM classifiers are the best on the Linux Bugs datasets, while XLNet is the best on the Financial dataset. Still, `ML-BERT` outperforms them by 5.7% and 2.3%, in each dataset, respectively. A notable exception can be seen in the performance of XLNet on the Financial dataset. When gauged against F_1 score, XLNet performs better than both the cls_concat_3 classifier and `SupportedBERT`, and only 2.2% worse than `ML-BERT`. The non-strict hierarchical structure of the Financial dataset most likely affects the performance of our framework, which targets the dependency among labels directly.

The overall performance of `DoubleHeadBERT` is lower than that of the other methods. In this regard, we can assume that the embeddings contain the most useful information classification-wise. Our experiments suggest that combining the classification output (i.e., the logits) of two separate classifiers does not provide enough semantic information for a third classifier to make the required adjustments to the prediction.

5.5.2 Baselines

Performance metrics across baselines are quite close; in some cases, we found that more recent approaches would perform worse than the SVM-based classifier, which instead performed remarkably well on both datasets. The most critical advantage that we would expect BERT to have over models based on traditional text representations is the ability to extract more expressive features that can embed both contextual and sequential information from the tokens. However, the Linux Bugs dataset is very noisy, with many grammatical inconsistencies and technical readings, like stack traces or memory addresses. These likely make little sense for a LM pre-trained on more structured natural language. On the other hand, the SVM-based approach utilizes BoW features weighted with TF-IDF; therefore, the classifier only looks at global word frequency without considering any structural information. Indeed, it is conceivable that the strength of the SVM classifier can be explained by

the lack of particular expressiveness in the structural information of these datasets. While the Financial dataset is less cluttered in terms of technical jargon, it is still rather noisy, while also being characterized by many structurally unsound sentences. Still, sentence structure is more expressive here, as demonstrated by the stronger performance of both our baseline LMs and our proposed approaches. However, as mentioned in § 5.3.1, the labeling in this dataset is less consistent than in the other, which explains the generally lower precision and recall scores.

The FastText classifier obtained worse performances than other methods on both datasets. Given the amount of noise in the datasets we experimented with, it is possible that fine-tuning the embeddings before applying them to the classification task may improve the results of this approach — similar to what we do in our Transformer-based approaches.

DeepTriage obtains decent results, though not on par with BERT-based models. This was to be expected, as Transformers are capable of higher semantic and syntactic understanding as compared to recurrent models, and therefore create more meaningful representations for the documents. A noteworthy disadvantage of DeepTriage is indeed its recurrent nature; the computational expense for training becomes quickly unmanageable when attempting to process longer sequences. While the original authors limit sentence length to 30 tokens, our preliminary tests showcased major improvements when allowing for longer sentences in the model, leading us to increase this threshold to 200 in our tests. The results are still noteworthy, as they are rather close to the best-performing flattened classifier (more so in terms of F_1 score).

In terms of framework, the approach based on XLNet is quite similar to the standard BERT approach. In our experiments, it was only tested on the flattened dataset, but it can be adapted for use with the proposed Multi-Layer architectures. However, XLNet is not pre-trained on a NSP task like BERT, and has no `[CLS]` token ready for classification; therefore, it needs an alternative summarization strategy. In our experiments, we observe that XLNet performs better than BERT with the *cls pooled* strategy, and its metrics are very similar to those obtained with BERT's *cls last* method. XLNet likely suffers from the same issues that BERT has on these datasets, i.e., is hindered by technical jargon and very concise sentence formulation. Still, as mentioned before, the model manages to beat our `SupportedBERT` approach on the Financial dataset in terms of F_1 score, and is rather close to `ML-BERT`. As the hierarchical structure of this dataset is rather weak, this likely showcases that a better understanding of the documents (i.e., better document embeddings) is more important than integrating the already inconsistent structure of the labels.

5.5.3 Document Embedding Summarization Strategies

The choice of document embedding summarization strategy has a considerable impact on classification performance, a fact that is confirmed by our tests on the Linux Bugs dataset (results in Table 5.8 and Fig 5.5). Using the *cls concat*$_3$ strategy

improves F_1 score and accuracy by 28.8% and 10.2% with respect to the standard classification method, which utilizes the `[CLS]` token after pooling from the last hidden layer only. We point out that the "raw" `[CLS]` token (not pooled) is superior to its pooled counterpart on this particular dataset; we have discussed the reasons behind the usage of such "pooled embeddings" in § 5.3.3. We also observe that both the *avg last* and *avg concat*$_2$ strategies improve in precision and recall over the widely used *cls pooled* approach, with the latter being the second-best strategy for F_1 score. However, in our experience, the *avg* strategy does not seem to work well with an approach based on concatenating the average of multiple hidden layers, and therefore we only report the result obtained by concatenating two hidden layers.

Our experiments aimed to measure the importance of the summarization strategies for word embeddings. By analyzing combination strategies based on normalized sums, averages, and other operations, we sought to verify whether we could effectively "follow a path" in the high-dimensional vectorial space to obtain a meaningful representation for a document. In more detail, word embeddings can be seen as points in an n-dimensional space; sequences of words can then be interpreted as a path of oriented vectors in the same space. Under this assumption, averaging embeddings is a reasonable way to obtain a single document vector representing the overall direction of a sequence of words. However, an average also accounts for the magnitude of word embeddings. To reduce its importance and focus on the direction of vectors, we also normalize the sum of vectors in two different ways. We obtain worse results, suggesting that, in our case, the length of vectors should be considered for a good document representation. We also report results using other approaches that have been used in the literature, like maximum and minimum, even if we find the geometrical interpretation of these operations to be less theoretically justifiable.

An interesting takeaway of these results is the effectiveness of the `[CLS]` token as a summarization of the document in BERT models. Moreover, this is true for all (or at least, a majority of) hidden layers of the model, as demonstrated by the effectiveness of combining multiple `[CLS]` tokens. Overall, we found the difference in results utilizing different summarization strategies to be quite striking. On these grounds, it is possible to hypothesize that providing the model with information from multiple hidden layers could allow it to leverage a richer set of features, therefore leading to better classification. We test this hypothesis in an experiment, the results of which are reported in Fig 5.6. In this experiment, we progressively concatenate the `[CLS]` representation from the deepest (last) layer backwards with the shallower ones. Upon inspection, we observe that the addition of more layers fails to provide a steady improvement. This could be attributed to at least two factors: *(a)* the deeper hidden layers already capture the most useful features for this task, or *(b)* simple concatenation is not the ideal approach for combining the information provided by these different layers.

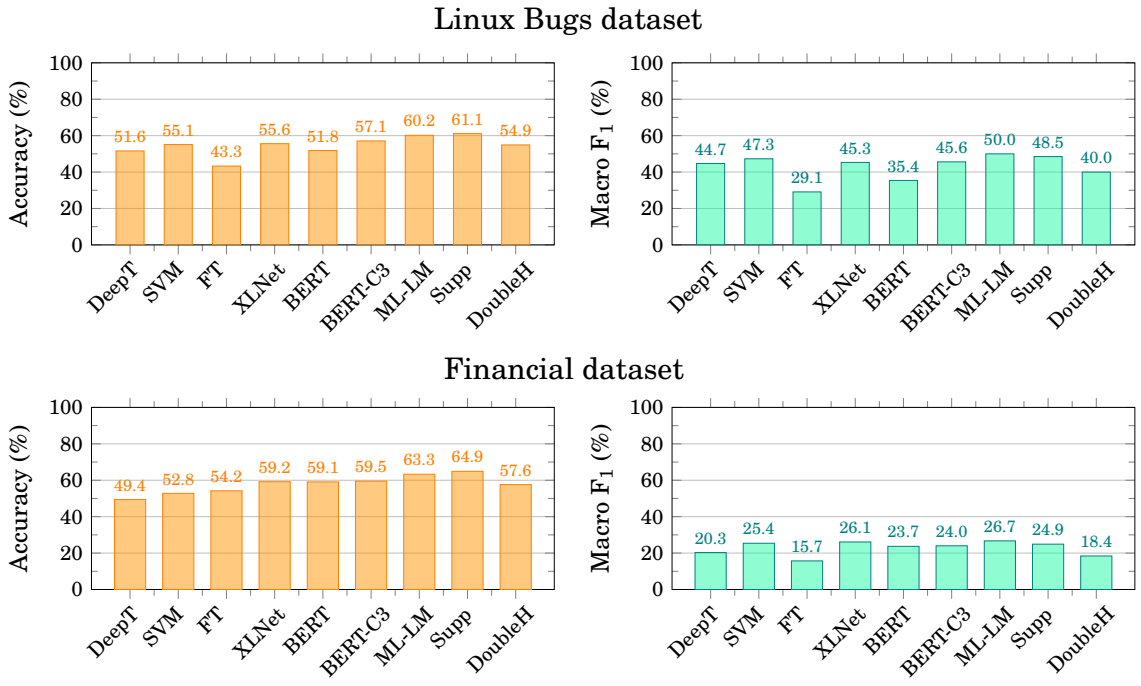


Figure 5.7: Visual comparison between the tested methods in test set accuracy (left) and macro F_1 (right) for the Linux Bugs and Financial dataset. Abbreviations: DeepT = DeepTriage, FT = FastText, ML-LM/Supp/DoubleH = our proposed strategies. As before, BERT and XLNet refer to the standard usage of those models with a single-layer classifier head.

5.6 Limitations and Opportunities

From an architectural standpoint, the primary limitation is that our multi-level model requires manual modifications to support arbitrary hierarchy depths; future work could focus on adapting it to handle varied hierarchical structures dynamically. Furthermore, we acknowledge that contemporary generative or larger encoder-only LMs would likely offer a more performant solution today.

We also point out that a fully supervised approach may not be the most suitable for all real-world applications, since companies may not have a fixed set of target labels or may want to change them dynamically. It would therefore be interesting to compare the effectiveness of hierarchical clustering methods, which could group tickets without predefined categories [130]. Additionally, the zero-shot capabilities of modern generative LLMs offer a compelling alternative that could achieve strong performance without the need for expensive fine-tuning, making them a worthy first benchmark for any future industrial application.

Further architectural improvements could involve validating other LMs within the `SupportedLM` and `ML-LM` architectures. For example, a model like ByT5, which operates on raw bytes, could be particularly effective for noisy ticket data by mitig-

ating out-of-vocabulary issues [349]. It would also be valuable to investigate if the performance gains from our hierarchical approach scale with larger, more general LMs.

Another interesting opportunity is to conduct a deeper analysis of the hidden layers within contextualized LMs. Our study found that simply concatenating more than three layers was not beneficial, which highlights the need to investigate more effective strategies for combining these internal representations. While our work focused on performance, advanced interpretability tools like the Language Interpretability Tool [264], Errudite [200], and iSEA [391] could be used to perform a more meaningful semantic error analysis of these factors. This could also provide important insights into how layer-specific fine-tuning contributes to global predictions, potentially leading to further architectural improvements.

5.7 Conclusion

In this chapter, we provided an up-to-date view of recent research in Ticket Automation, categorizing the literature according to the sub-task it addresses. We identified four primary sub-tasks and delved into one of the most critical, Ticket Classification. We explored the application of contextualized LMs — specifically, BERT and XLNet — on two public hierarchical TC datasets. The first, containing bug reports, features a well-organized label hierarchy, while the second, a collection of customer complaints addressed to financial companies, has a less clearly structured hierarchy.

We investigated several strategies for producing document-level representations from BERT's word embeddings, and our results on both datasets show that the chosen strategy significantly impacts performance. This underscores the importance of treating the embedding strategy as a key hyperparameter to be tuned during validation. Furthermore, we introduced and tested three multi-level classifiers designed to leverage hierarchical label dependencies. Two of our proposed model-agnostic frameworks demonstrated solid improvements over standard "flattened" classification approaches. This analysis concludes the first part of my work related to the development and practical application of TC methods.

III

Explainable AI Design and Language Understanding

The previous chapters discussed my work on pre-trained LMs and their application to text classification (TC) problems. While these deep learning (DL) models achieved strong performance, their inherent opacity motivates the research presented in this part, which is dedicated to building and analyzing more transparent classification pipelines. The first two chapters demonstrate solutions to practical tasks using Explainable Artificial Intelligence (XAI) methodologies, while the last provides an analysis of language understanding in LLMs, evaluated through classification and self-rationalization performance. Chapter 6 establishes that, for some tasks, transparent models should be preferred over opaque black boxes. Inspired by seminal works advocating for simplicity [191, 293], this chapter uses a computer vision task — classifying the ripeness of banana crates — to argue that significant gains in explainability can be achieved with only a minimal loss in accuracy. From this basis, Chapter 7 addresses the necessary compromises when transitioning from simple problems to complex ones. Recognizing that the ideal of a single, fully transparent model is often impractical in Natural Language Processing (NLP) due to the complexity of language, this chapter proposes a novel "layered" approach to explainability. The framework operates on the principle that explanations are most valuable at specific levels of granularity; it therefore permits the use of opaque models for semantic feature extraction while ensuring the final decision-making layer remains transparent. Finally, Chapter 8 investigates the ability of generative Large Language Models (LLMs) to identify puns and to explain their own predictions, framed as a study on subtle language understanding and rationalization capabilities. In this study, we analyze how even the most sophisticated LLMs struggle with this nuanced task of humor detection, revealing that their understanding is often superficial.

6

Prioritizing Transparent Models in Classification

The success of DL has made complex, opaque models the default choice for many tasks solved with Artificial Intelligence (AI). While powerful, their intrinsic opacity complicates the explanation of their predictions, which can hinder debugging, prevent meaningful audits, and reduce user trust. This chapter, largely based on our research published in Rizzo et al. [8], challenges the assumption that this complexity is always necessary. As the first investigation in this section, we showcase how a problem can be effectively solved by pairing data processing strategies with traditional, inherently transparent models. To this end, we demonstrate an often-neglected, knowledge-based approach to problem-solving. Our contribution is threefold. First, we introduce a set of guidelines for creating "explainable-by-design" systems. Second, we apply these guidelines to the task of classifying the ripeness of banana crates, and conduct a direct comparison between our transparent model and an opaque DL baseline. The results demonstrate a minimal loss of accuracy for a significant gain in the interpretability of the classifier's decisions. Finally, we validate our approach through a user study evaluating the perception and usefulness of the generated explanations.

6.1 Introduction

Over the past decade, DL has driven significant advances in machine learning (ML), achieving high accuracy across various cognitive tasks. However, as detailed in Chapter 3, this complexity creates challenges in interpreting model decisions and guaranteeing accountability, fairness, and transparency — requirements that vary by system criticality. These regulatory obligations often conflict with the intuitive, trial-and-error design process typical of DL, where formalized metrics for such criteria are frequently lacking. Building on the framework established in Chapter 3, we view an explanation as the translation of computational *evidence* into a semantic *interpretation*. This process requires balancing faithfulness (adherence to the model's actual reasoning) and plausibility (alignment with the end-user's reasoning) of an

Figure 6.1: Ripeness stages for crates of bananas from least ripe (1) to ripest (4).

interpretation, which must be conveyed through an effective Explanation User Interface (XUI). This holistic perspective on *explanation* as a *process* challenges the common research focus on maximizing accuracy at the expense of interpretability. While rigorous techniques like SHAP [133] can support this process, they cannot cover all the necessities of a realistic system, which invariably requires domain knowledge, user validation, and precise requirements definition. To address this, this chapter proposes a solution-oriented approach grounded in established design principles for explainability. Through stakeholder consultations and comparative performance benchmarks, we empirically select the simplest models capable of meeting system requirements. This strategy facilitates a comprehensive design and analysis of model explainability, which we illustrate through a practical industrial application.

6.1.1 Task and Approach

To showcase our design strategy, we analyze a straightforward real-world scenario in which stakeholders required a degree of transparency into an automated classification system. Our target task is the classification of the ripeness of banana crates on a scale from 1 (least ripe) to 4 (ripest) (see Fig 6.1 for an example).

In our approach, we design for accuracy and explainability simultaneously. To tackle the classification task, we select a pool of three DL methods: (i) a simple Convolutional Neural Network (CNN) model with three convolutional blocks, (ii) a pre-trained convolutional model based on the MobileNetV2 framework [149], and (iii) a pre-trained Vision Transformer (ViT) [325]. As we will show, the latter allows for almost perfect results and is the best neural model across our proposed methods; however, none of the XAI methods we have tried seem capable of explaining its predictions meaningfully. On the other hand, we show that an alternative approach, based on simple color features and a Decision Tree classifier (DT), can provide competitive accuracy while exposing the information needed (i.e., the *evidence*) for producing satisfactory global explanations.

6.1.2 Contributions

Key contributions of our work include:

- We establish design principles for ML problems that prioritize explainability from the outset;
- We analyze various DL methods for a classification task that requires a balance between accuracy and explainability;
- We demonstrate the effectiveness of a simpler, transparent DT model with minimal feature engineering on the same task;
- We conduct a user study to verify that the generated explanations align with stakeholders' needs.

Our experiments reveal that three neural models can easily achieve high classification accuracy, which prompts us to question the necessity of complex, opaque methods for a seemingly simple task. This finding highlights the viability of simpler, more transparent models that can achieve comparable performance without sacrificing explainability. Based on these results, we outline a set of practical design principles to help practitioners build for explainability, emphasizing the importance of matching model complexity to the complexity of the problem itself.

We release our code and self-collected dataset[1] to support reproducibility and the extension of our research.

6.2 Related Work

Our research intersects two significant areas: (i) enhancing explainability in AI and (ii) optimizing fruit ripeness grading.

Explainable AI DL models are often criticized for their opacity, which makes explaining their predictions challenging. Chapter 3 presented a taxonomy of XAI techniques, highlighting their reliability issues [166, 222, 165] and discussing why such methods cannot function as holistic solutions to the explainability problem [129]. Despite these limitations, model-agnostic tools like LIME [100] and SHAP [133] remain widely adopted due to their ease of integration and visually compelling outputs. However, these visualizations can often produce deceptively plausible explanations that mask underlying inconsistencies. Beyond reliability concerns, we believe the primary limitation of these techniques is their failure to bridge the semantic gap: they typically provide low-level evidence as explanation, such as a heatmap of important pixels, but do not guide the user toward a faithful interpretation of that evidence. This leaves a significant cognitive burden on the

[1] https://gitlab.com/distration/dsi-nlp-publib/-/tree/main/stop-overkilling-24

end-user, who must attempt to fill the gap between the low-level explanation and a meaningful, high-level understanding. This process can lead to interpretations that are misaligned with the model's actual behavior.

In this study, we therefore compare our transparent, explainability-by-design approach against SHAP [133], which remains a popular and rigorous benchmark in saliency-based explainability, applicable well beyond the computer vision domain. Aligning with the argument from Rudin [191] for using inherently transparent models in high-stakes tasks, our research advocates for a strategy where explainability is a core design principle. We argue that simply applying some "magical" XAI technique on top of an arbitrary black-box model is unlikely to meet the practical needs of end-users who require genuine explanations.

Fruit ripeness recognition Grading fruit ripeness has been addressed using statistical methods [41], traditional ML [250], and DL approaches [365]. DL methods, known for high accuracy and minimal feature engineering, are currently among the state-of-the-art. Rizzo et al. [5] provide a comprehensive survey on fruit ripeness grading. However, a common trend in recent work is a focus on minor accuracy improvements, often at the expense of explainability.

6.3 Designing for Explainability

We propose some guidelines to identify the discriminative features that are most intuitive for stakeholders and to process them using the simplest ML method adequate for the task. In this context, "simplicity" refers to the model's complexity (e.g., number of parameters, the lower, the better) and its reliance on a human-understandable process.

We propose a general, explainability-first design process that follows these high-level steps:

i *Define the task* to be solved using ML, the *input data*, and the *stakeholders* of the final system;

ii Engage with stakeholders to *identify the attributes* they consider relevant for solving the task and which of those attributes should form the basis of an explanation;

iii *Select candidate ML models* that can effectively process the defined attributes. The primary selection criterion should be the model's ability to provide expressive evidence that supports an interpretation that is faithful to its internal logic and aligns with the stakeholders' intuition. It is also useful to select a state-of-the-art, potentially opaque model to serve as a performance benchmark.

iv *Evaluate* both the model's *performance* and the effectiveness of the generated explanations. The chosen model should be competitive with the state-of-the-art while also meeting the stakeholders' expectations for explainability. This may require testing for faithfulness and plausibility; user-centric metrics can be assessed through a user study, which can help gather qualitative evidence of the efficacy of the proposed XUI.

Step (ii) is perhaps the most challenging, especially when very little problem-specific knowledge is available to the stakeholders. In this scenario, benchmarking against a state-of-the-art black-box model can help establish the task's difficulty. If the task involves intuitive features that can be leveraged, a less complex and more transparent model is worth trying. The intuitiveness of both the features and the model is critical to creating an explanation that is faithful to the model's behavior and plausible to the human stakeholder. We acknowledge, however, that finding meaningful, human-understandable features can be challenging for some tasks. In NLP, for example, the inherent complexity and ambiguity of language often make handcrafted features impractical. That is why we advocate for a pragmatic approach that involves testing a variety of ML algorithms and XAI techniques, especially when the task appears simple. For some problems, a simple and explainable solution may not yet exist, but it is always worth seeking one first. The following subsections showcase how we applied each step of these guidelines to our problem.

6.3.1 Task Definition, Stakeholders, and Data

From a practical perspective, this work deals with a multiclass image classification task. Our stakeholders are workers at the wholesale fruit market of Treviso, Italy, who are interested in automating the ripeness grading of banana bunches. Currently, operators manually label bunches on an increasing ripeness value (1 to 4, least to most ripe, see Fig 6.1). All the bananas within a crate are assumed to be in the same ripeness stage. The ML classifier resulting from this work would aid operators in labeling large numbers of incoming crates. Moreover, this is the first step in digitalizing the fruit processing pipeline, from inspection and assessment of fruit quality to online sales. Given the impact of the assessment step on fruit pricing, our stakeholders stressed the importance of maintaining transparency in the grading process to allow human supervision and validation.

We collected an *ad-hoc* dataset comprising 927 images to develop the ML solution with a reasonable balance between the four ripeness classes. The dataset was manually labeled by the operators who perform the quality assessment of incoming products. To understand human performance on this classification task, we also asked three operators to re-label a subset of images from the dataset. More technical details on the data are provided in § 6.4.1, while human performance is reported in § 6.5.

6.3.2 Feature Selection

After consultation with the stakeholders, we determined that color is the most reliable and intuitive factor in determining the ripeness of banana bunches. Images are encoded using the RGB color space, a well-known color model backed up by solid theory based on the human perception of colors. Since color is the most important feature of our dataset, we process images to extract valuable color information and train our classifier to recognize the ripeness stage by considering such color. § 6.4.1 details how this information is extracted and used in the proposed solution.

6.3.3 On the Choice of Models

For this task, we selected both state-of-the-art DL methods and simpler, more transparent classifiers. Testing DL models gives us an idea of the best performance that can be achieved, as well as the problem's difficulty. As stated, our primary objective is to identify the least complex model that achieves adequate performance, thereby maximizing transparency. To this end, we compared a DT, a Support Vector Machine (SVM) with a polynomial kernel, and a multinomial Naïve Bayes (NB) classifier. Ultimately, we selected the DT as the most suitable transparent model, given its comparable performance and high transparency.

We point out that the DT learns discriminative rules that partition the feature space into regions corresponding to each target class (i.e., the ripeness stage). By extracting color information in the RGB space, we can obtain a *global* explanation that maps each ripeness stage to specific areas of that color space. We highlight that this explanation is *faithful* by design, in the sense that it directly describes the DT "reasoning" process. It is also *plausible*, as it is grounded in features (the RGB color) that are both relevant to the task and understandable to stakeholders, and it relies on the DT's intuitive rule-based logic. These characteristics make this strategy effective with respect to point (iii) of our guidelines.

6.3.4 Testing for Accuracy and Explainability

As for step (iv) in our guidelines, we compare the performance of the baseline models (DT, SVM, NB) and select the DT to be the best compromise between the complexity and intuitiveness of the explanation that can be derived from it, as discussed in the previous section. The NB classifier achieves lower performance than the DT. Conversely, the SVM with a high-degree polynomial kernel achieved slightly better results (less than 0.5% accuracy and F_1-score improvement). However, given the minimal difference in results and considering that the decision boundaries of the polynomial SVM are more difficult to understand because of their complexity, the DT appears to be a better choice. Additionally, we compare the DT with some state-of-the-art DL models that would be the obvious off-the-shelf solutions for this task, in line with recent computer vision trends. All results are reported in Table 6.1, showcasing

that the DT achieves competitive performance and is well above human classification performance. Finally, we want to assess the efficacy of the generated explanations for our stakeholders. To do this, we conducted a user study to investigate the users' preferences about the generated explanations, which is documented in § 6.5.3.

6.4 Methods and Explanations

6.4.1 Data and Preprocessing

Our dataset consists of 927 RGB images of banana crates captured at 4160 x 3120 pixels using a CZUR Shine Ultra scanner under consistent lighting. Images were resized to 224 x 224 pixels for compatibility with pre-trained models and to facilitate reasonable inference times. The dataset underwent augmentation with random transformations such as rotation, affine and elastic morphology transformations, cropping, Gaussian blur, and perspective changes to mimic real-world scenarios like smartphone photography. About half of the dataset was augmented and incorporated into the training set. The dataset's visual inspection indicated noise, especially around crate boundaries. To address this, we applied semantic segmentation using the SLIC algorithm [65], which improved results for all methods (an example of a segmented image from our dataset can be seen in Fig 6.2). All reported results refer to the performance when using these segmented images. For the selected DL models, feature extraction is automated from the raw RGB input. However, we employed minimal feature engineering for the DT, focusing on color features. Each image was represented by its average color's normalized R, G, and B channel values. We also adjusted luminance by converting images to YUV space, normalizing the Y channel, and reverting to RGB. This process is important since RGB embeds luminance within its channels, unlike spaces like YUV, with a separate luminance channel.

6.4.2 Deep Learning Approach

In addressing the task of banana ripeness classification, we run and compare three neural approaches. The first architecture consists of a simple CNN using three convolutional blocks, each characterized by two bi-dimensional convolutions and max pooling interleaved by ReLU activation functions. The convolutional layers extract features fed to a three-layer feed-forward neural network, which outputs the final prediction. Before being processed by the CNN, the data is normalized to the mean and standard deviation (SD). The second architecture we consider is the pre-trained MobileNetV2 network [149]. Still convolutional by nature, the strategy at the core of this method is based on depth-wise convolutions and inverted residual connections. The designers aimed to build a powerful, pre-trained model for low-tier devices. The third architecture we examine is the Vision Transformer (ViT) [325]. Transformers [128] are neural architectures based on multi-head attention [92],

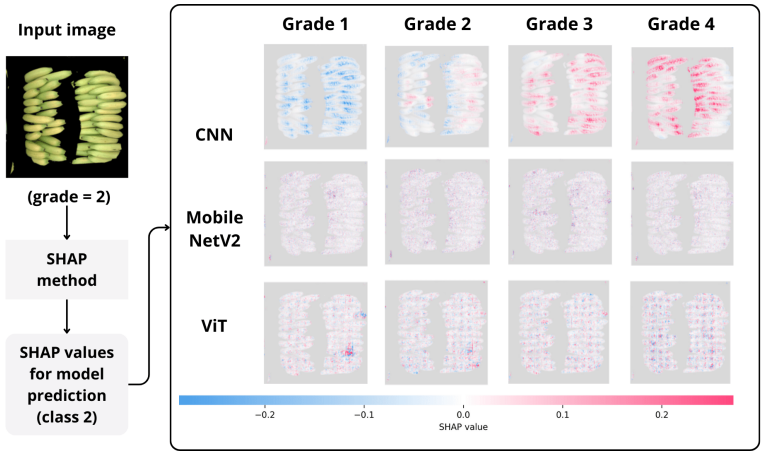


Figure 6.2: Examples of explanations for DL models generated using SHAP.

widely studied and employed by the NLP community [1]. This architecture, which was introduced in Chapter 2, has been successfully applied to a variety of computer vision tasks [369]. Briefly, ViT splits an image into fixed-size patches and linearly embeds them. Positional embeddings are then added to the patch embeddings to retain spatial information. The resulting sequence of vectors is then fed to a standard Transformer encoder. For classification, a learnable "classification token" is prepended to the sequence, analogous to the `[CLS]` token used in BERT [192]. In our experiments, we use the `vit-base-patch16` model [271], which was pre-trained on ImageNet.

6.4.2.1 Explanation strategy

As previously mentioned, we employ SHAP [133] to explain the predictions of the DL models. Unlike architecture-specific methods such as Grad-CAM (which is strictly tailored to CNNs) or attention mechanisms (specific to Transformers), SHAP is model-agnostic and easily applicable to a variety of modalities, including images, text, and tabular data. This versatility makes it the most popular first choice for many practitioners. When dealing with images, SHAP allows for the generation of heatmaps as the primary XUI. These heatmaps are intended to describe the importance of each pixel for the model's prediction. Intuitively, warm colors indicate the regions of the image that contributed most positively to the prediction, while colder colors indicate areas that contributed negatively. Example explanations generated with SHAP are presented in Fig 6.2. As discussed in § 6.2, such feature importance heatmaps can lend themselves to ambiguous or even misleading interpretations by end-users. Specifically, as can be seen in Fig 6.2, the heatmap appears to highlight *where* the model is mostly looking, but not *what* it is seeing. This creates an alarming condition where the explanation can convey to the user a "convincing lie" about how the model behaves. The next sections show how our design addresses the challenges

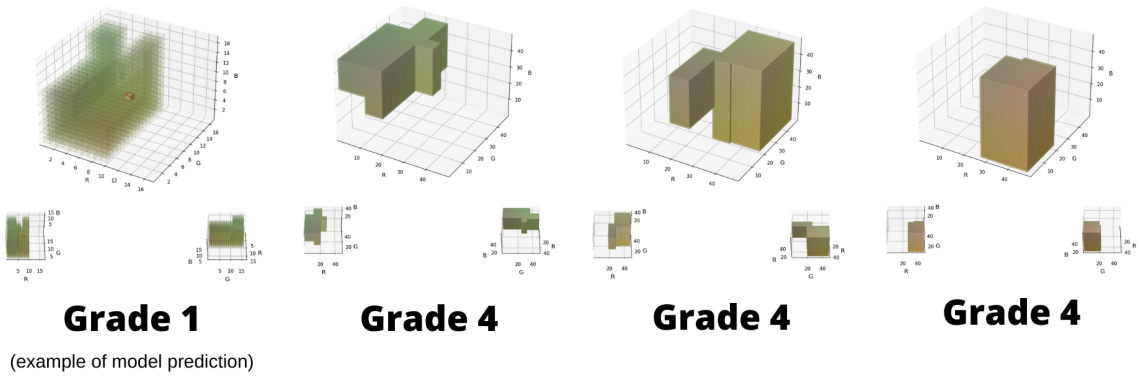


Figure 6.3: Explanation generated from the DT's learned constraints on the RGB color gamut. The four ripeness grades identify different areas within the gamut, while the highlighted red square within the "Grade 1" plot represents the average color for the specific input example, providing a local explanation.

of achieving both faithfulness and plausibility.

6.4.3 Decision Tree

In contrast to the examined DL methods' inner complexity, we propose tackling the same task using a simple, more transparent model based on a DT classifier. In particular, we adopt the implementation offered by scikit-learn, based on the CART algorithm [16].

6.4.3.1 Explanation strategy

Achieving explainability is more straightforward with specific models, such as those with fewer parameters, as they tend to simplify the process of assigning an intuitive and faithful interpretation. For example, the DT provides us with the possibility of inspecting its decision rules. These rules are the *evidence*, which can be trivially interpreted as a series of binary decisions based on learned splits on the RGB features. For every non-leaf node, a DT learns a threshold for one of its features, producing two children based on whether the feature value is above or below that threshold. In our case, each instance is classified by following a path to a leaf node labeled with a specific ripeness class. Conveniently, the set of rules from the traversed path defines a specific area within the RGB color space, which forms the basis of our explanation. Basing the explanation on the intuitive process of discriminating between banana crates by color (as our stakeholders do) makes it inherently *plausible*.

However, while the decision paths of shallow trees are simple to follow, the decision paths can grow exponentially for features with complex interactions. Furthermore, the raw numerical splits within the DT can still appear opaque to an

Table 6.1: Macro-averaged performance metrics for the models averaged over ten random seeds (± SD). The performance of the discarded baselines is also reported in the last three rows.

	Accuracy	*Precision*	*Recall*	F_1
Decision Tree	97.16 ± 1.04	97.23 ± 1.06	96.78 ± 1.19	96.97 ± 1.10
CNN	93.49 ± 1.15	92.98 ± 1.31	93.08 ± 1.23	93.77 ± 1.23
MobileNet V2	97.43 ± 0.46	97.26 ± 0.46	97.17 ± 0.54	97.18 ± 0.49
ViT	99.67 ± 0.15	99.60 ± 0.20	99.66 ± 0.17	99.62 ± 0.18
Human Perf.	75.00 ± 5.89	75.88 ± 4.53	75.00 ± 5.89	75.19 ± 5.24
SVM (linear)	88.12 ± 4.57	90.84 ± 2.65	85.74 ± 4.61	86.76 ± 5.01
Naïve Bayes	85.18 ± 2.09	84.41 ± 2.11	84.24 ± 1.95	84.21 ± 1.96
SVM (poly)	97.62 ± 0.71	97.61 ± 0.63	97.34 ± 0.84	97.46 ± 0.74

average user without a proper XUI to convey the explanation clearly. ML experts, for instance, may be satisfied with understanding the range of RGB values mapped to each target class, but non-expert users will likely need these rules to be processed and visualized more clearly.

Thus, we take our explanation a step further by devising an XUI that is designed to be human-understandable. We use the rules extracted from the decision paths as constraints on the RGB gamut to identify and visualize the portions of the color space corresponding to the four ripeness classes. This allows any new input to be represented by its average color and easily located within one of these regions. This plot, exemplified in Fig 6.3, is our proposed explanation for the DT's behavior. It is worth stressing that since the colored regions are extracted from the complete set of rules learned by the DT, this visualization is, by definition, a global explanation. As such, our strategy allows us to unequivocally see which colors are associated with each label class. One of the key benefits of such an interpretable explanation is the ability to validate the classifier's behavior; any unexpected colors appearing in the XUI would immediately point to a potential bias in the model.

6.5 Experiments

In this section, we compare the performance achieved by our employed methods. First, we analyze the classification metrics achieved by the three DL models and the DT. Then, we study the explanations generated according to the strategies proposed in § 6.4.2.1 and § 6.4.3.1 and compare them through a user study involving our stakeholders.

6.5.1 Performance

We measure the predictive performance of our selected models using four standard classification metrics: accuracy, macro-averaged precision, macro-averaged recall, and macro-averaged F_1 score. All methods are tested using 5-fold cross-validation, repeated ten times with different random seeds to strengthen the results. Table 6.1 showcases the results achieved with both DL methods and the DT. We additionally report the human performance, which is the average of the scores obtained by three stakeholders on the classification of a balanced dataset of 300 randomly sampled images from the original non-augmented dataset ($\sim 20\%$). It is easy to see that all methods achieve excellent results, with all metrics surpassing the 90th percentile scores and improving on a human baseline. It is worth remembering that these results are achieved on the datasets augmented with images that have gone through various augmentations, which makes them more robust at the cost of small decreases in performance.

A manual error analysis also reveals that mistakes always occur because the classifiers select an adjacent class (e.g., class 2 instead of class 1). The ViT model achieves a near-perfect score among the selected methods for all metrics. The DT also obtained outstanding results, though this required comparatively more effort (including the standardization of the luminance and extensive hyperparameter tuning). Nevertheless, this process allows the DT to have results comparable to those of MobileNetV2.

6.5.2 Explainability

We compare the SHAP explanations for the DL models with the handcrafted explanations based on RGB color designed for the DT. Figures 6.2 and 6.3 compare the two types of explanations for the same input. It is easy to see that the masks produced by SHAP do not highlight meaningful features of the image. Indeed, we can observe that the highlighted regions seem random, although some users might be tempted to artificially make sense of these heatmaps due to their visual appeal, which seems to suggest the model is actually looking at the right thing (i.e., the bananas). We agree that SHAP's heatmap XUI is very *understandable* in the sense of having a manageable cognitive load; however, this property alone does not make for a good explanation. As seen in Fig 6.2, in our case, SHAP's visualization for the CNN consistently presented a similar result for all classes, seemingly finding very important features for grade 4 (even when the CNN correctly classified other ripeness stages). The situation does not change when we visually examine the explanations generated by the other methods.

This does not necessarily mean that the explanations generated by SHAP are not faithful to the model's inner workings. Rather, the intuitive interpretation of the highlighted regions is not expressive enough to clarify how the model uses those features internally, for example, to detect edges or shadows. As such, we can

only conclude that, despite their plausibility, these visualizations are inadequate as meaningful explanations for this specific task. However, this may not always be the case. Assuming the visualization correctly reflects the model's reasoning, the position of important pixels would be more meaningful in coarse-grained classification tasks. For instance, in an object detection task on ImageNet, if the important pixels are located on an animal's muzzle, the intuitive interpretation is that the muzzle is a crucial feature. Nevertheless, even in this more favorable scenario, the interpretation of the heatmap comes from the user and is not guaranteed by SHAP. Different users might arrive at different interpretations, for example, thinking the heatmap is highlighting facial hair instead of the muzzle. This exemplifies why we believe saliency-based methods do not fully solve the explanation problem on their own but rather support the process of generating an explanation.

On our specific dataset, however, the position of these highlighted regions does not convey intelligible information. Conversely, our explanation (both local and global) for the DT is faithful to the model's inner workings by design. This strategy provides the user with a much more informative explanation that is understandable, plausible, and faithful to how the model works.

6.5.3 User Study

We designed a user study to investigate preferences for the generated explanations. The study involved 20 stakeholders in the banana ripeness grading process, representing a diverse range of backgrounds and AI expertise. Through an online questionnaire, participants were asked to self-report general demographic and professional characteristics (detailed in Table 6.2) and then compare two types of explanations for the same prediction: (i) a heatmap generated by SHAP for the top-performing ViT model, and (ii) a visualization of the input's average color in the RGB gamut, extracted from the DT model. The primary objective was to determine which explanation better showed why the model made its decision.

When asked about the value of explainability, all participants agreed that explanations were necessary, with most considering them essential. As for the preferred method, ten of the twenty respondents considered the DT's RGB visualization to be the most effective, while nine voted for the SHAP heatmap. Three declared that neither explanation was helpful to them[2]. This result is interesting; although SHAP's visualizations do not provide an unambiguous explanation (as they only show *where* the model was looking), their visual appeal was compelling enough for many participants to deem them trustworthy.

Finally, 80% of respondents declared that the chosen explanation would improve their trust in the model, and 70% were willing to trade approximately 5% of the classifier accuracy for a more transparent and human-explainable decision process. Considering that the accuracy loss between our DT and the most accurate model is

[2]Two participants selected both the RGB explanation or the "neither" option.

Table 6.2: Distribution of participant characteristics (self-reported) in the user study. Total participants $N = 20$.

Characteristic	**Category**	**Count (N)**	**Percentage (%)**
Age Group	18–25	1	5.0
	26–30	6	30.0
	> 30	13	65.0
Years of Experience	0–5	7	35.0
	> 10	6	30.0
	0–4	5	25.0
	5–10	2	10.0
AI Understanding	1 – No understanding	1	5.0
	2	1	5.0
	3	5	25.0
	4	2	10.0
	5	3	15.0
	6	3	15.0
	7 – Expert	5	25.0

only around 2.5% for this task, there appears to be little practical reason to prefer the more opaque model.

To determine if these preferences were influenced by user background, we conducted a Spearman's rank correlation analysis between the expressed preferences and user characteristics (age, domain experience, and AI familiarity). The analysis yielded no statistically significant correlations, indicating that the observed preferences were independent of the collected user characteristics.

We acknowledge, however, a key limitation. The relatively small sample size prevents us from drawing generalized conclusions, despite our efforts to include participants with varying levels of experience. Additionally, the XUI used to deliver explanations, shown in Fig 6.3, was basic and may have influenced the results. Nevertheless, the study is valuable in demonstrating that users may be drawn to explanations that are superficially appealing but potentially less faithful to a model's true reasoning. It highlights that the user's perspective is a central factor in designing truly usable and effective explanations.

6.6 Limitations and Opportunities

While our findings support the value of our explainability-by-design approach, this study has several key limitations that, in turn, point to valuable opportunities for future research. The primary limitation is the simplicity of the case study itself. The classification of banana ripeness relies on a single, highly intuitive feature (color),

which makes it an ideal candidate for a simple, transparent model. Applying our guidelines to more complex tasks, where discriminative features are not as easily defined or where domain knowledge is unavailable, would be significantly more challenging. Additionally, our use of a simple average color to encode each image, while effective here, is a simplification that would not suffice for problems requiring an understanding of attributes like texture, shape, or complex patterns. Future work could extend this design principle by incorporating a higher number of more complex features (e.g., color distribution) while accounting for explainability. More complex visual tasks could leverage keypoint descriptors such as SIFT [103] to encode information concerning shapes or gradients, preserving a degree of explainability that is lost in end-to-end DL models. In this work, we utilized SHAP because of its widespread adoption as a robust, "off-the-shelf" solution and its flexibility across various data modalities. While other image-specific techniques, particularly gradient-based methods, were not tested here, we point out that any saliency-based approach providing explanations in terms of input features would likely be unsuitable for this use case. Indeed, as illustrated in Fig 6.2 and discussed in § 6.2, raw feature masks fail to convey semantic meaning to the user; similar masks generated by alternative XAI methods would suffer from the same "semantic gap" issue.

As mentioned, the user study, while providing valuable preliminary insights, was limited by its small, relatively homogeneous sample size and an unrefined presentation of the explanations. A larger, more diverse study with a more sophisticated XUI would be needed to validate these findings more robustly. Finally, in line with the explainability-by-design principle, a promising avenue for future work is the exploration of regularization strategies to improve the inherent explainability of complex DL models, especially for producing more robust explanations [181, 144]. Investigating how to impose constraints on the features learned by neural networks could guide them to develop more human-understandable representations. This approach can help bridge the gap between high-performance black-box models and the need for transparent, unambiguous explanations [332, 406].

6.7 Conclusion

This chapter has discussed the explainability of ML models by proposing high-level guidelines to tackle ML problems with transparency as a primary goal. We demonstrated this approach by comparing three DL models to a DT for classifying bananas into four ripeness stages. While the DT yielded slightly lower accuracy, it produced inherently more explainable results. This task showcases how *model design* can be engineered to support an intuitive explanation strategy, rather than relying purely on a *post-hoc* approach. We argue that, where possible, working with a transparent model and stakeholder-understandable features provides satisfactory explanations with a minimal trade-off in accuracy. To validate this claim, we conducted a pilot user study with 20 participants, comparing explanations from SHAP against those

derived from color features and rules learned by a DT. The results indicate a user preference for more understandable models, even at the cost of minor accuracy losses, yet they also reveal a tendency among non-experts to favor simpler explanations, regardless of their faithfulness.

While this principle of prioritizing simpler, transparent models is powerful, many real-world problems are too complex to be solved by a single shallow model. The next chapter, therefore, addresses this challenge by proposing a method to build explainable systems for such tasks, strategically layering both simple and complex components to maintain transparency.

7

Designing Layers of Explainability

We build upon the premise that genuine explainability requires transparency in model design. This chapter addresses classification tasks where a single transparent model is insufficient, proposing a compromise between transparency and the need for opaque DL systems. Specifically, we introduce a novel, modular framework for TC that prioritizes explainability without a significant performance trade-off. The core of our methodology is the premise that low-level features can be extracted using powerful models, while the final aggregation and classification are performed by a transparent model. We show that this layered approach allows us to trace a decision path more effectively than using end-to-end DL methods with a post-hoc explainer. Our results demonstrate that this method yields effective and faithful explanations with only a marginal reduction in accuracy, presenting a compelling trade-off for applications where understandability is paramount.

This approach can be seen as a direct extension of the principles outlined in the previous chapter, and was adapted from our research published in Zangari et al. [9]. The main distinction lies in the complexity of the problem: while the previous work addressed a task solvable with a single, domain-specific feature (color), this chapter tackles a more complex TC problem. Here, the goal is to leverage a wide range of general, yet semantically meaningful, features to build a transparent and robust classifier. Additionally, we conduct a formal study of faithfulness, comparing our method's explanations against those derived from a standard, end-to-end fine-tuned LM.

7.1 Introduction

DL-based approaches differ from traditional ML methods in their innate ability to learn complex, high-order features. These encodings, developed through layers of compression and non-linear transformations, make it challenging for humans to understand their meaning or origin. In contrast, traditional ML relies on manually crafted features by experts, which are more straightforward but require intensive labor and time to create. As discussed in the previous chapter, the vocabulary of raw input features (e.g., pixels or words) often lacks the expressiveness to explain a model's reasoning, as these features are combined in deep layers into complex

representations that are inexplicable in terms of the original inputs. In other words, the autonomous feature extraction capability of DL models comes at the price of a significant gap, a *divide*, between the expressiveness of the model's internal representation of the world and the human capacity to comprehend it. More precisely, this divide originates from the discrepancy between the high-dimensional feature extraction and optimization process typical of DL and the human-scale reasoning needed to explain a decision effectively [107]. This consideration leads to an important argument against the unregulated usage of DL: the risk and likely possibility of unintentionally encoding various forms of bias into the model [323].

Despite these challenges, the effectiveness of DL models in solving complex problems cannot be overlooked. As seen in Chapter 2, NLP, in particular, has evolved from traditional methods requiring extensive preprocessing to modern LLMs that generalize language understanding and contextual reasoning far beyond manual feature engineering.

7.1.1 Task and Approach

To address the divide between DL models and human comprehension, we propose a layered design process that strategically utilizes DL and traditional methods to tackle smaller sub-problems. The core of the approach is to use capable pre-trained LMs as feature extractors to convert raw text into a set of high-level, semantically meaningful features (e.g., the presence of sarcasm, the main topic of the text). This has the advantage of tapping into LMs' vast knowledge that wouldn't be possible with traditional computational approaches. These features are then used as input for a simpler, transparent classifier, for which we use an XGBoost model.

We validate this approach on two binary classification tasks: sexism detection in social media posts and polarity detection in movie reviews, both framed as binary classification problems.

7.1.2 Contributions

The primary contributions of this work are as follows:

- We propose a layered framework that generates explanations based on semantic features rather than low-level raw inputs, aiming to bridge the "explanation divide" for end-users;
- We compare our approach against standard end-to-end DL classifiers, quantifying the trade-off between performance and explainability;
- We analyze the faithfulness of our method's explanations using functional metrics and compare them to those derived from the baseline DL models;
- We conduct a user study to gather evidence on the perceived quality and effectiveness of our explanation strategy.

We release the full code implementation for reproducibility[1].

7.2 Related Work

As detailed in our previous chapters, the dominant paradigm in XAI for NLP involves applying *post-hoc* methods to explain pre-trained "black-box" models. State-of-the-art approaches like LIME and SHAP typically provide local, feature-attribution explanations by assigning importance scores to individual input tokens. However, studies have demonstrated that both feature-attribution methods and other common techniques, such as those based on gradients or attention scores, can sometimes produce misleading or unfaithful explanations [387, 199, 375].

For these reasons, evaluating the quality of the explanation is a crucial area of study. The two most commonly measured criteria are *plausibility*, which assesses how convincing an explanation is to a human observer, usually measured through user studies, and *faithfulness*, which measures how accurately the explanation reflects the model's actual reasoning process [258]. Recent work has proposed various metrics for faithfulness, with a comparative analysis provided in Chan et al. [354].

As mentioned in our introductory review in Chapter 3, the XAI field is still debating the definition of fundamental concepts, with contrasting results and opinions among researchers [339]. Beyond these foundational issues, previous research has shown that explanations based on low-level features like pixels or words can fail to convey meaningful information to end-users [402]. Such explanations can result in vague, ambiguous interpretations, as they lack the conceptual richness required for a person to gain an understanding of a model's reasoning. The work in this chapter addresses these limitations, proposing an approach that provides evidence at a higher semantic level to produce explanations that are less ambiguous and more faithful.

7.3 Proposed Approach

In this section, we provide an overview of our proposed paradigm, detailing both the arguments that motivated its design and the technical specifics of its implementation for TC.

7.3.1 Argumentation and Overview

As the scale of LLMs (and DL methods, in general) is increasing at a far higher pace than the speed at which explainability methods are being developed, it has become

[1] `https://gitlab.com/distration/dsi-nlp-publib/-/tree/main/layers-of-explainability-24`

evident that different approaches to XAI might be warranted, especially whenever human awareness of the system is a key factor. At the same time, it is hard to discard DL methods entirely because of their outstanding performance. Thus, we propose and design a process that balances manual and automatic feature extraction.

At the core of this idea stands the argument that, at some point, we must ultimately rely on prior knowledge that may or may not be precise [15]. Indeed, every model trusts the data that is fed into it, which is assumed to come from reliable sources. For instance, a ML model processing sensor data has to believe the hardware performing such readings is itself working correctly and that the resulting data reflects reality. Likewise, we might utilize DL approaches to extract granular features for which we can afford a certain degree of trust [326], and then utilize these features to make more complex decisions. This approach also relies on the concept that an explainable model does not necessarily have to explain how all low-level features are extracted, but rather that explainability is a useful tool at the "last layer" of a decision system. By utilizing a model built on such features, we can create effective explanations (as they pertain to concrete textual properties with human-understandable semantics) without entirely discarding the effectiveness of neural approaches used as feature extractors. Moreover, these extractors need not be limited to DL approaches but can be complemented by utilizing more traditional feature extraction methods based on specific domain knowledge (whenever available).

In this work, we focus on TC tasks and showcase how a system built on semantic and grammatical features can achieve good results while providing a more interpretable decision path. Conceptually, this approach may also be utilized in other fields; in computer vision, for instance, one might aggregate extractors that produce features regarding an image's shapes, orientations, and composition.

7.3.2 Feature Extraction and Preprocessing

We extract various types of features, both leveraging pre-trained LMs to extract specific characteristics from the text (e.g., by classifying emotions) as well as more traditional statistical features (word count and several readability scores). The former category includes a RoBERTa model trained for irony detection, emotion recognition, topic extraction, and grammatical correctness. Overall, we obtain an initial set of 383 features.

We preprocess all the extracted features, removing constant and complementary ones. We then perform a feature selection using Recursive Feature Elimination. This results in 94 features for the CMSB dataset and 350 for the IMDb (the datasets are described in the next section). Selected features are min-max scaled to a range of $[0-1]$, where the resulting score indicates the magnitude of that feature's presence in the text.

A list of feature extractors utilized is provided in Table 7.1, along with a brief description and references to the original works or tools used.

Table 7.1: List of the main feature extraction methods used in our work.

Feature extractor	Based on	Description
Emotion LM	RoBERTa-base	Presence in the text of 28 human emotions. Based on the "roberta-base-go_emotions" fine-tuned on the go_emotions dataset [1].
Irony	RoBERTa-base	Presence (or not) of irony in the text. Based on "twitter-roberta-base-irony" LM, fine-tuned on TweetEval [272].
Sarcasm	BERT-base	Presence (or not) of sarcasm in the text. Based on "english-sarcasm-detector" [2] fine-tuned trained on the News Headlines Dataset For Sarcasm Detection [394]
Offensiveness	RoBERTa-base	Whether the text can be considered offensive (or not). Based on "twitter-roberta-base-offensive" LM, fine-tuned on TweetEval [272].
Evidence Type	MiniLM	Detects evidence types (or absence thereof) in text, such as assumptions, definitions, testimonies, statistics, or anecdotes. Based on the MiniLM fine-tuned model [3].
Topic	RoBERTa-base	The likelihood of a text belonging to various topics among 19 pre-set categories (e.g., sports, technology, relationships, etc.). Based on the model fine-tuned in [371].
Text Emotion	EmoLex	The likelihood of a text expressing certain emotions among 28 pre-set categories (e.g., joy, anger, disappointment, curiosity, pride, etc.). Based on the NRC Word-Emotion Association Lexicon [74].
Text Polarity	TextBlob	A polarity (e.g., negative, neutral, positive) score supplemented by a subjectivity score computed using the TextBlob Python package [4].
LIWC	LIWC-2007	Reveals over 100 linguistic and socio-psychological dimensions of textual data, using the Linguistic Inquiry and Word Count (LIWC) dictionary-based method [88].
Empath Features	Empath	Lexical categories assigned to words (e.g., violence, depression, femininity) computed using Empath [95].

[1] https://huggingface.co/SamLowe/roberta-base-go_emotions
[2] https://huggingface.co/helinivan/english-sarcasm-detector
[3] https://huggingface.co/marieke93/MiniLM-evidence-types
[4] https://textblob.readthedocs.io/en/dev/quickstart.html#sentiment-analysis

7.4 Experimental Settings

This section describes the datasets and methods used in our experiments and provides technical details.

7.4.1 Data

IMDb The IMDb dataset [59] consists of 50,000 movie reviews from the IMDb website, which are used for binary sentiment classification. Following the original work and established convention, reviews are labeled as positive (score ≥ 7) or negative (score ≤ 4), excluding neutral ones. It is evenly split with 25,000 reviews each for training and testing.

CMSB The "Call me sexist but" dataset [308, 338] contains over 13,000 texts from various sources, including tweets and surveys, for binary sexism detection. It features both original and minimally altered adversarial texts to shift sexist content to non-sexist. In our experiments, mentions to other users and URLs are replaced by standardized strings (i.e., "MENTION" and "LINK") to prevent the model from focusing on irrelevant patterns.

7.4.2 Classification Models

In our experiments, we want to compare the effectiveness of our proposed features against the current state-of-the-art models for TC. As such, we test a wide range of classification approaches, focusing on less opaque methods to maintain as much transparency as possible. We then compare the results with a Transformer model fine-tuned on the bodies of text in the dataset.

We test various classifiers on our features, including simpler methods such as Decision Trees, Logistic and Ridge Regressions, and various ensemble-based approaches. Among the tested methods, tree-based methods performed particularly well, with XGBoost delivering the best performance. As a Transformer baseline, we utilize a RoBERTa-base model pre-trained on tweets [357], which provides excellent results on both datasets[2]. We performed the same preprocessing on text for the feature extractors and utilized the pre-trained tokenizer for the corresponding model.

7.4.3 Explanation Approaches

This work focuses on local explainability in TC, emphasizing two essential aspects: evaluating feature importance in model reasoning and effectively communicating this to users via an XUI [402]. For feature evaluation, we employ SHAP [133], specifically utilizing TreeExplainer for XGBoost models and TranSHAP [305] for BERT-like

[2]We tested other RoBERTa models on the IMDb, without significant improvements.

models. TranSHAP is particularly effective for visualizing textual explanations, as it accounts for the sequential and structural nature of the text. We use the SHAP framework here for consistency with our previous work and due to its widespread adoption in the NLP domain. Regarding the XUI, we initially considered various visual and textual interfaces. Our previous user study showed that, while SHAP offers powerful visualizations, interpreting them correctly often requires a technical understanding that is unsuitable for non-expert users. Motivated by this finding, we propose a simplified interface that presents only the most essential information: the raw input text, the model's output, the ground truth label, and a clear list of the top features that contributed to the outcome. To make these feature contributions even more intuitive, the scaled feature values from the feature extractor described in § 7.3.2 (representing the magnitude or probability of each feature) are also discretized into "low", "mid", or "high" categories. Using Shapley values from SHAP, we also determine each feature's contribution percentage to the model's output, and add to the explanation only the top-k most influential features.

Explanations for both XGBoost and pre-trained LMs follow the same process, with SHAP used for LMs to highlight important words or symbols in the text, as it is commonly used in the literature. An example of the full explanation for XGBoost is shown in Fig 7.1, with the corresponding SHAP explanation for RoBERTa in Fig 7.3.

For the user study, these explanations were further simplified. We selected only the top five most influential features or words for each prediction, omitting their numerical importance values (which some users may find confusing) and using them only for ranking. Furthermore, to enhance intuitiveness, we embedded abstract feature values (e.g., low, mid, high) into natural language descriptions through simple rephrasing. The simplified versions of these explanations for the same examples are shown in Figures 7.2 and 7.4.

7.4.4 Faithfulness Evaluation

We adopt the metrics proposed by [256] to evaluate the faithfulness of our explanation strategies, specifically measuring both *comprehensiveness* and *sufficiency*. In the equations below, $c(x)$ is the predicted class (1 or 0) for input text x, and $p_{c(x)}$ is the model's confidence (probability) in the predicted class. $x_{:k\%}$ signals the top-$k\%$ most significant features for the model's prediction. B is the set of k values of choice, that we set to $B = \{1, 5, 10, 20, 50, 75\}$.

Comprehensiveness assumes that an interpretation is faithful if the important features are highly representative of the entire input, such that their removal would substantially change the model's confidence. The faithfulness score is determined by the change in the output probability of the originally predicted class once the important features are removed (Eq (7.1)).

1a. Feature-based textual explanation (our method) - extended

Example: "More frequently I see guys behaving straight out of their emotions. that's female behavior imo"
True class: 'sexist' - **Predicted class**: 'sexist' (confidence=74.1%)
Top features that contributed towards 'sexist':

- [imp=11.5%] LIWC words in the category 'student life & youths' (score=mid)
- [imp= 8.8%] LIWC words in the category 'fashion & style' (score=high)
- [imp= 7.0%] Text shows 'disgust' (score=high)
- [imp= 5.8%] Text contains assumptions (score=high)
- [imp= 3.4%] Text is ironic (score=high)
- [imp=27.0%] Other features (mean=0.52%)

Top features that contributed towards 'not sexist':

- [imp= 6.4%] LIWC words in the category 'human beings' (score=low)
- [imp= 2.6%] Text contains definitions (score=low)
- [imp= 1.5%] Text shows 'disapproval' (score=low)
- [imp=26.0%] Other features (mean=0.54%)

Figure 7.1: Example of an explanation extracted from the XGBoost predictor trained on our features.

1b. Feature-based textual explanation (our method)

Example: "More frequently I see guys behaving straight out of their emotions. that's female behavior imo"
True class: 'sexist' - **Predicted class**: 'sexist' (confidence=74.1%)
Explanation: The top characteristics of the text that contributed towards 'sexist' are the following (sorted by importance):

- Text is moderately likely to talk about 'student life & youths'
- Text is likely to talk about 'fashion & style'
- Text is likely to show 'disgust'
- Text is likely to contain assumptions
- Text is likely to be ironic

Figure 7.2: Example of an explanation extracted from the XGBoost predictor as shown in the user study.

2a. Word-based textual explanation (RoBERTa LM) - extended

Example: "More frequently I see guys behaving straight out of their emotions. that's female behavior imo"
True class: 'sexist' - **Predicted class**: 'sexist' (confidence=97.3%)
Top features that contributed towards 'sexist':

- [imp=22.4%] female
- [imp=17.7%] behavior
- [imp=14.4%] guys
- [imp=7.7%] imo
- [imp=4.1%] see
- [imp=9.0%] Other words (mean=1.76%)

Top features that contributed towards 'not sexist':

- [imp=7.3%] emotions
- [imp=5.0%] out
- [imp=3.3%] frequently
- [imp=2.5%] of
- [imp=6.6%] Other words (mean=1.91%)

Figure 7.3: Example of an explanation extracted from the RoBERTa LM.

2b. Word-based textual explanation (RoBERTa LM)

Example: "More frequently I see guys behaving straight out of their emotions. that's female behavior imo"
True class: 'sexist' - **Predicted class**: 'sexist' (confidence=97.3%)
Explanation: The top words in the text that contributed towards 'sexist' are the following (sorted by importance):

- female
- behavior
- guys
- imo
- see

Figure 7.4: Example of an explanation extracted from the RoBERTa LM as shown in the user study.

$$\text{COMP} = \frac{1}{|B|} \sum_{k \in B} (p_{c(x)}(x) - p_{c(x)}(x \setminus x_{:k\%})) \tag{7.1}$$

Sufficiency complements comprehensiveness and assumes that if some non-critical features are masked (only top-$k\%$ are kept), the model's confidence in the original decision should remain relatively stable (Eq (7.2)).

$$\text{SUFF} = \frac{1}{|B|} \sum_{k \in B} (p_{c(x)}(x) - p_{c(x)}(x_{:k\%})) \tag{7.2}$$

We employ these metrics as an objective measure to compare our explanation strategies. The strategy that achieves a *higher comprehensiveness* and *lower sufficiency* is considered the most faithful. For XGBoost, features are eliminated by assigning them the median value of the training set, thereby minimizing their informativeness by confining them to a commonly occurring range of values. With the pre-trained LM, words in the input sentence are masked using the `<unk>` token that is employed for unknown symbols. The evaluation is conducted on two test splits of approximately 3500 examples each. For the IMDb dataset, these splits were created by randomly sampling from the official test set.

7.4.5 User Study

To validate our approach, we conduct a preliminary user study to explore the users' preferences regarding the types of explanations described in § 7.4.3. This study involved 21 volunteers, who were asked to compare explanations of both kinds. The users were presented with 8 explanations, 4 based on the semantic features of the XGBoost model, and 4 from the LM approach. In each subset, 2 of the explanations concern the same data sample, while the other 2 differ from those of the other method. We measured subjectively reported quantitative metrics on a Likert scale from 1 to 7, building on existing work in the literature [351]. These metrics are the *(i) trust* in the model output (do you trust that the model made the right decision?); *(ii) plausibility* of the model output and its explanation (how much the provided explanation is realistically portraying the model's decision process?); *(iii) satisfaction* (does the explanation answer your "why" question?); and the *(iv) understandability* (how much the explanation is understandable based on how it is presented?).

7.5 Results and Discussion

7.5.1 Quantitative Results

We utilize standard evaluation metrics for binary classification tasks: accuracy, precision, recall, and F_1 score. Table 7.2 and Table 7.3 list the quantitative results of

the top-performing classifiers.

From a quantitative point of view, the LM-based approach has a definite edge. While this comes as no surprise, the results obtained by our approach are still quite competitive. On the CMSB dataset, XGBoost's accuracy, F_1 score, and recall decrease by 4.2%, 3.3%, and 7.2%, respectively, compared to the LM, despite having better precision (1.9% increase). Similar trends are observed on the IMDb dataset, with our method performing about 5% worse across metrics.

Table 7.2: Performance metrics (%) averaged over ten random seeds ± SD (CMSB dataset).

	Accuracy	*Precision*	*Recall*	F_1
Logistic Regression	88.1 ± 0.4	76.3 ± 2.2	61.4 ± 1.0	64.6 ± 1.3
Decision Tree	88.2 ± 0.3	74.6 ± 0.9	68.8 ± 1.3	71.0 ± 1.1
XGBoost	90.7 ± 0.5	81.5 ± 1.1	74.0 ± 1.5	77.0 ± 1.4
RoBERTa (base)	94.6 ± 0.1	79.9 ± 0.9	79.5 ± 1.8	79.6 ± 0.5

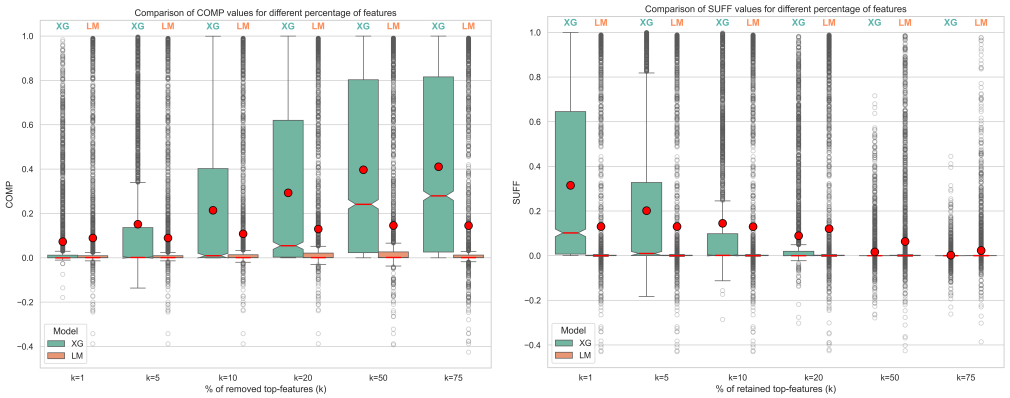
Table 7.3: Performance metrics (%) for a single run on the standard test split (IMDb dataset).

	Accuracy	*Precision*	*Recall*	F_1
Logistic Regression	87.3	87.3	87.3	87.3
Decision Tree	85.4	85.4	85.4	85.4
XGBoost	88.9	88.9	88.9	88.9
RoBERTa (base)	93.5	92.8	94.3	93.6

7.5.2 Faithfulness Analysis

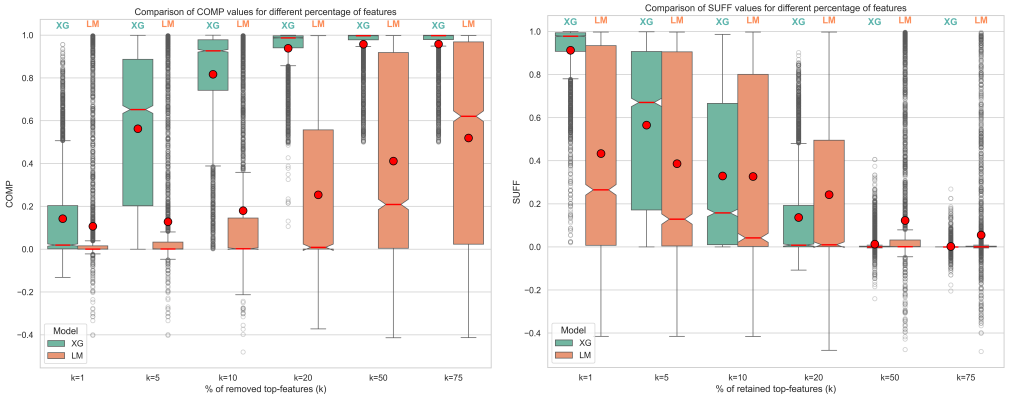
Box plots for faithfulness metrics on CMSB and IMDb are shown in Figures 7.5a and 7.5b and Figures 7.6a and 7.6b, respectively. A red line marks the median, and a red circle marks the mean. Our XGBoost classifier (XG) paired with SHAP's values yields higher COMP values on average than the RoBERTa model (LM) on both datasets (CMSB: XG COMP=0.257, LM COMP=0.118; IMDb: XG COMP=0.73, LM COMP=0.267). However, SUFF scores are closer (CMSB: XG SUFF=0.128, LM SUFF=0.1; IMDb: XG SUFF=0.327, LM SUFF=0.261), with LM performing slightly better.

We utilize the Wilcoxon signed-rank test to verify our hypothesis that XG has higher COMP and lower SUFF for each $k\%$. On the CMSB dataset, the hypothesis holds with statistical significance (p-values < 0.05) for COMP-5 to COMP-75 and SUFF-20 to SUFF-75, with an average effect size of 0.46. On IMDb, it holds for all



(a) Comprehensiveness (COMP ↑) (b) Sufficiency (SUFF ↓)

Figure 7.5: COMP and SUFF on the CMSB test set (teal = XGBoost, orange = LM).



(a) Comprehensiveness (COMP ↑) (b) Sufficiency (SUFF ↓)

Figure 7.6: COMP and SUFF on the IMDb test set (teal = XGBoost, orange = LM).

COMP measurements and SUFF-20 to SUFF-75. The difference is even more evident in the latter dataset, with the XGBoost model behaving significantly better than the LM in terms of COMP. The average effect size is 0.82, indicating that the XGBoost model tends to have significantly better COMP values than the LM. However, the SUFF comparison on both datasets shows that the LM is more consistent with the explanation when seeing only $< 20\%$ of the top words, and the effect size is also smaller in all these measurements.

According to both metrics, our XGBoost model aligns better with its explanation strategy than the LM, indicating more faithfulness. It is likely that XGBoost heavily relies on feature interactions, which our explanations do not fully capture. Indeed, most features have little individual discriminative power and can recreate context only when combined. Hence, keeping only the top-$k\%$ features with a $k \leq 10$ leads to a significant drop in confidence, negatively affecting the SUFF scores. While the

score is worse, we find it reasonable that confidence drops when too few features are retained. Conversely, LM is more stable, possibly inferring more context from fewer keywords than XGBoost can from key features. Due to its extensive general pre-training, this is also sensible since the LM has a clear advantage in filling in missing context when words are masked. The significant difference in text length between the two datasets must also be considered, as it likely affects the word-based explanations derived from the LM. CMSB samples are very short, averaging 15 words (as they are mostly tweets), while IMDb reviews are much lengthier, averaging over 200 words. The number of words retained or discarded is rounded up when considering a percentage of the sentences for the computation of these metrics; this implies that lower values of k in the CMSB will likely maintain or discard the same number of words. This partially explains why the bar plots for the CMSB dataset are more leveled in the case of the LM.

Overall, the COMP and SUFF values for the LM appear smoother, suggesting that the model heavily relies on hidden contextual representation not revealed by our explanation method. In contrast, the XGBoost-based strategy more clearly reflects the dependency on the features we identified as important, providing evidence of better faithfulness.

7.5.3 User Study Results

The results of the user study are reported in Table 7.5, which shows the average and SD for the self-reported metrics detailed in § 7.4.5. User characteristics are described in Table 7.4.

Overall, a positive trend across these metrics suggests that our proposed explanation method has potential benefits over the baseline approach. To investigate these preferences further, we performed a Spearman's correlation analysis (chosen for its suitability for ordinal and non-normally distributed data) between the self-rated user characteristics (see Table 7.4) and the average preference for our explanation strategy. We found a moderate, positive, and statistically significant correlation ($\rho = 0.51$, $p < 0.05$) between "AI understanding" and preference for our method (in terms of overall satisfaction with the explanation). This suggests that users with higher technical literacy appreciate the depth of our proposed explanations, whereas users less familiar with AI tend to prefer the simpler, but less expressive, word-based baseline. Correlations with other attributes were weak and not statistically significant, likely due to the limited variance in the sample, as the majority of participants reported similar age ranges and educational levels. Finally, we observed the same pattern when analyzing other reported metrics, such as the perceived understandability of the explanation.

However, these findings are subject to important limitations. First, the reliance on a small, homogeneous sample (primarily student volunteers from the Master's program) limits the generalizability of the results. Furthermore, a key finding was that neither explanation method significantly increased user trust. Therefore,

Table 7.4: Distribution of participant characteristics (self-reported) in the user study. Total participants $N = 21$.

Characteristic	**Category**	**Count** (N)	**Percentage** (%)
Age Group	18–25	4	19.0
	26–35	15	71.4
	> 40	2	9.5
Level of Education	Graduate	11	52.4
	High school	4	19.0
	PhD	1	4.8
	Post graduate	5	23.8
AI Understanding	1 – No understanding	4	19.0
	2	2	9.5
	3	4	19.0
	4	3	14.3
	5	3	14.3
	6	3	14.3
	7 – Expert	2	9.5

Table 7.5: Results of the user study ± SD (↑) (LM vs XGBoost explanations). MO and E stand for Model Output and Explanation, respectively.

	Trust (before)	Plaus. (MO)	Plaus. (E)	Trust (after)	Satisfaction	Underst.
LM	5.42 ± 1.52	5.50 ± 1.49	4.01 ± 1.70	3.83 ± 1.79	3.58 ± 1.97	3.83 ± 1.93
Ours	5.79 ± 1.09	5.85 ± 1.10	4.40 ± 1.35	4.51 ± 1.34	4.01 ± 1.49	4.06 ± 1.43

while the slight preference for our method is encouraging, it should not be taken as definitive proof of its superiority. We believe the primary contribution of this work lies in the improved faithfulness scores discussed in the earlier section; this user study serves as a source of preliminary, but positive, validation for our approach.

7.5.4 Discussion

Despite a 3–5% performance decrease in F_1 score compared to the end-to-end baseline, our layered approach provides more consistent and faithful explanations. Notably, our feature extractors were not fine-tuned, which suggests our framework has a degree of generalizability to other tasks and that performance could be significantly improved with task-specific tuning. The advantage of our approach was particularly evident in our faithfulness measurements, which measure an explanation's proximity to the model's reasoning. The end-to-end LM underperformed on the COMP metric, with up to 63% lower COMP than our proposed method, suggesting

its word-based explanations are less faithful than those produced with our approach. The SUFF metric results were less clear-cut, with our strategy showing higher sensitivity to top feature removal. This is reasonable, given our approach's reliance on feature combinations, whereas the baseline model draws on the vast internal knowledge from its pre-training process. Our pilot user study, while based on a limited pool of participants, indicated a preference for our high-level feature-based explanation strategy.

Our results also highlight that, while pre-trained LMs are undoubtedly state-of-the-art for TC tasks, simplistic word-based explanation strategies can be unreliable and misleading. Quite importantly, these explanations appeal to users due to their intuitiveness and lack of required interpretation, unlike the higher-level explanation we provided for the XGBoost model. Interestingly, in our user study, 42.9% of users found the LM explanations more plausible (on average) than the more faithful XGBoost model explanation. In contrast, 38% found it more understandable and stated that the LM explanation increased their trust in the model decision in at least one instance. These noisy results once again underline the difficulty of designing a good XUI and evaluating it effectively in user studies. Faithfulness measurement remains an open problem due to challenges in establishing unbiased ground truth (i.e., how can we compare faithfulness metrics without an objective ground truth?) [354]. Despite this, our benchmark provides preliminary quantitative evidence of the effectiveness of our explanation method.

7.5.4.1 Considerations on the proposed approach

The underlying principle of this explainable-by-design framework is that if a feature extractor has been sufficiently validated to make reliable predictions, its output can be trusted as a high-level input for a more complex problem that demands greater transparency. Modern LMs, for instance, have shown a remarkable ability to identify complex rhetorical figures like irony and humor. Once such a classifier is evaluated and deemed reliable, its output can be integrated as an interpretable feature in a more transparent pipeline. In this sense, this approach has strong similarities to other DL methods with *ante-hoc* explainability strategies, which also rely on a foundational assumption about an opaque component. Prototypical neural networks, for example, operate on the premise that a neural network can produce a semantically meaningful embedding space where similarity-based reasoning can occur [208, 300]. In this space, the distance between an input's embedding and a learned "prototype" is assumed to reflect a similarity that is reasonable to the end-user. While this similarity measure drives the final, transparent classification decision, the process of generating the embeddings themselves is not explained. Just as our framework accepts a validated but opaque feature extractor, prototypical networks accept a validated but opaque embedding space as the foundation for their similarity-based reasoning. This argument can be made for many other approaches, like Concept Bottleneck Models [254], and attention mechanisms [352]. In each of these cases, a

pragmatic trade-off is made: the inscrutability of a powerful representation learner is accepted in exchange for a final reasoning step that is engineered for transparency, a principle to which our framework adheres.

Furthermore, our framework is modular, which allows it to be applied recursively. The core assumption is that for many systems (and for our specific task), an explanation for the final classification layer is the most critical requirement. However, if a specific feature extracted by an opaque ML method also requires an explanation, the layered method can be applied to that sub-model as well. For example, if an irony detection score is used as a feature, the black-box irony classifier could itself be replaced with a more transparent, layered version.

7.6 Limitations and Opportunities

A primary limitation of this study is the modest performance trade-off, as the end-to-end LM achieves better performance, which might be critical in some applications. However, since our feature extractors were not fine-tuned, a clear opportunity exists to close this performance gap with task-specific tuning. Second, our task did not involve specific domain knowledge, whereas other practical applications could benefit from the inclusion of more targeted features, potentially improving performance.

Regarding our user study, while it provides valuable preliminary insights, it was limited by a small sample size drawn from a relatively homogeneous pool of users, as well as by the basic nature of the XUI employed. A larger, more diverse study would be needed to validate these findings more robustly. Such a study should also involve discussing the requirements of an effective XUI with a carefully selected sample of users. The insights from social sciences on the design of interactive user interfaces would also be helpful in this regard, as the clarity of presentation greatly influences understandability.

Furthermore, while our method provides a transparent final decision layer, it does not fully capture the “why” and “how” behind the feature extractors’ decisions. This is because our approach is still fundamentally based on feature importance. In other words, our method can only answer questions concerning the importance of its high-level features for the final prediction. While aware of this, our goal was to reduce the “divide” between the vocabulary of features used in the explanation and the low-level inputs the model processes. Despite its limits, we demonstrated that this approach can improve both faithfulness and plausibility.

Finally, a promising opportunity lies in combining our layered methodology with structured knowledge. Similar to how our approach extracts intuitive features, formally structuring data into a knowledge graph with a domain-specific ontology could provide another path toward explanations built on high-level, human-aligned concepts [273, 419, 359, 368]. A key opportunity here is to leverage the emergent abilities of modern LLMs for this very purpose, investigating their effectiveness at automatically extracting such structured knowledge from unstructured text.

7.7 Conclusion

This chapter proposed an approach to TC tasks based on a layered procedure that emphasizes explainability. We utilized a combination of DL and non-DL extractors to create a knowledge base of fine-grained features upon which our classifiers operate. This design makes it easier to follow the models' decision logic, as the features are at a higher level of abstraction. While not matching the raw performance of end-to-end LM-based approaches (3–5% drop in F_1), our methods perform competitively while offering more faithful explanations (up to +173% increase in *comprehensiveness* and +28% in *sufficiency*). Preliminary results from a small-scale user study indicate positive feedback toward our explanation strategy, but suggest that its higher complexity might not appeal to all user types.

The next chapter concludes this part on XAI by evaluating the ability of modern LLMs to genuinely understand a nuanced form of humorous language. This is tested by assessing their capacity to both recognize puns and produce coherent explanations for their decisions. Indeed, the impressive performance of LLMs on various linguistic tasks makes them prime candidates for use as the opaque feature extractors in a layered framework like ours; this has partly motivated the initial study in the next chapter, which evaluates the reliability and true depth of understanding of these models before they can be trusted in such a role.

8

LLMs and the Illusion of Humor Understanding

This chapter proceeds to test the limits of understanding in some of the most recent LLMs on a linguistic task [11]. We investigate whether the impressive performance of LLMs stems from a deep, human-like grasp of language or sophisticated pattern-matching. To explore this question, we examine the practice of prompting an LLM not only to solve a task but also to provide a justification for its answer, a process known as *rationalization*.

We investigate this capability through the lens of *puns*, a form of humorous wordplay that exploits polysemy and phonetic similarity. While LLMs have shown promise in detecting puns, we demonstrate that their understanding often remains shallow and lacks the nuanced grasp of human interpretation. By systematically creating and reformulating pun benchmarks, we show how subtle linguistic alterations are sufficient to mislead state-of-the-art LLMs. Our contributions include new, challenging benchmarks for pun detection, a comprehensive human evaluation of recent LLMs, and a critical analysis of the robustness challenges these models face when processing ambiguous language. To complete our analysis, we assess the quality of the rationales generated by the models, providing insights into how their reasoning aligns with the human interpretation of puns.

8.1 Introduction

Understanding linguistic nuances, such as hedges, idiomatic phrases, figures of speech, and metaphors, is crucial for effective communication [442, 481]. This requires deep contextual and cultural awareness, presenting ongoing challenges for LLMs that struggle with subtle, multi-layered language [409, 355, 436].

One notable example of nuanced language is the *pun* (paronomasia), a form of wordplay generating rhetorical (often humorous) effects through polysemy and phonetic similarity. Prominent in literature, poetry, and advertising [86], puns depend on the intuitive recognition of dual meanings (literal vs. figurative), creating the characteristic *pun effect* [12, 109]. The ability to automatically recognize puns is

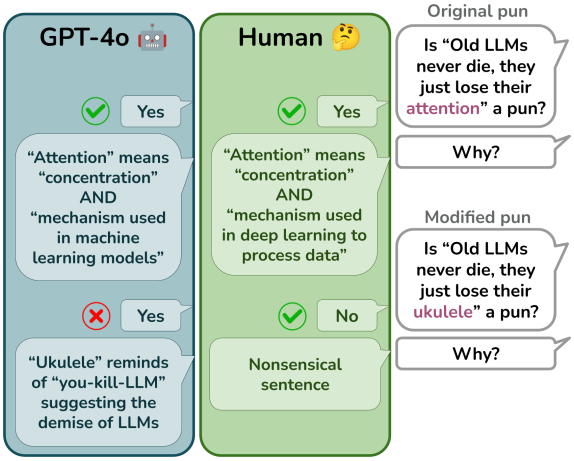


Figure 8.1: Example pun detection and explanation by GPT-4o vs. human. Subtle modifications to the pun (replacing the polysemous word, *attention*, with a random one, *ukulele*) are often sufficient to mislead LLMs.

relevant for digital humanities (e.g., for processing literary classics and ancient texts) and various NLP tasks, such as sentiment analysis and machine translation, which struggle with the ambiguity and non-literal nature of puns [94]. In comparison to other rhetorical devices, such as sarcasm, metaphors, or jokes, puns are structurally simpler [49, 122], and easier to formalize [376]. These qualities make them a good candidate for assessing linguistic understanding in LLMs.

While recent studies have examined pun generation [363, 378], detection, and explanation [377, 445, 122] using LMs, this line of research faces two key limitations. First, most studies have relied on a single dataset from SemEval [122], where shallow cues and data leakage may inflate performance [412]. Second, evaluations are typically focused on binary classification or hard-to-assess free-text rationales. A systematic analysis of more structured rationales would help clarify the capabilities and limitations of LLMs in interpreting rhetorical ambiguity [455, 319].

8.1.1 Task and Approach

Motivated by these limitations, we introduce two new collections of annotated short texts, PunnyPattern and PunBreak, designed specifically to evaluate the robustness of LLMs in pun detection. These collections move beyond simple detection, using targeted substitutions and common language patterns to probe whether models can accurately recognize a pun's context and structure, or if they rely on memorization and superficial cues. An example of a pun and an altered pun is in Fig 8.1.

We evaluate 7 open- and closed-weights generative LLMs on both identifying puns by prompting them to answer a "yes/no" question, and generating supporting *rationales*. We then analyze the quality of these rationales and report on their impact on detection performance. In contrast to earlier studies [377, 445], we design

prompts that elicit semi-structured rationales, enabling more systematic evaluation via automatic metrics and a detailed manual analysis guided by a new annotation protocol.

We publicly release both datasets and the code used in our experiments [1].

8.1.2 Contributions

We organize our discourse around the following research questions:

- RQ1: How accurately do LLMs detect puns, and how does their performance vary across new and existing datasets?
- RQ2: How robust are LLMs at detecting puns?
- RQ3: To what extent can LLMs explain puns?

Our results show that most LLMs superficially associate puns with common language patterns, which makes it difficult for them to distinguish genuine puns from structurally similar sentences that contain no pun. Automatic and manual error analyses further reveal that the models struggle to handle context and phonological properties effectively. Notably, prompting them to explain their reasoning slightly improves performance and helps counter this bias.

8.2 Related Work

SemEval-2017 Task 7 [122] targeted pun detection, location, and interpretation tasks by releasing a dataset of annotated puns and non-puns[2]. The participating systems were mostly based on traditional NLP techniques, with leading systems employing heuristics based on positional and semantic features, such as n-grams enhanced by Word2Vec embeddings [68] and phonetic distance measurements. The reported results showed strong performance in binary detection, moderate in pun location, but notably weak in correct sense association. SemEval-2021 [313] later expanded this dataset by adding new jokes, although without additional annotations.

Building upon this foundation, the JOKER workshop [413] introduced multilingual datasets for pun detection, location, interpretation, and translation, encompassing English, French, and Spanish. The JOKER corpus [412] aimed to address earlier criticisms of imbalance and reliance on superficial cues by providing 3,506 annotated short jokes in English and French, alongside 1,700 negative examples generated via word substitutions. Despite utilizing advanced LLMs such as GPT-3 [279] and BLOOM [420], a custom-trained T5 model [292] consistently outperformed

[1] `https://github.com/alezanga/punintended`

[2] A *non-pun* is a sample that does not contain a pun (a negative sample for the pun detection task)

others on all tasks, albeit with limited success (detection $F_1 < 0.6$). Other studies have further explored non-English puns [477, 411, 446].

Complementing these efforts, Sun et al. [377] investigated pun explanation by enhancing training data with rationales generated by T5 models. They measured explanation quality using simulatability, defined as the accuracy difference when explanations are provided as additional input vs. when they are not (see § 3.4 in Chapter 3 for more details). They found that high-quality human-annotated rationales can improve model performance, but automatically generating such explanations with LMs remains challenging. In a related work, Sun et al. [376] focus on pun generation by retrieving context and pun words. They propose a system that first retrieves a suitable pair of words, along with their senses, from given contextual words, and then employs a T5 model to generate a pun. However, the success rate of the generated puns was reported to be significantly lower than that achieved by humans.

To the best of our knowledge, Xu et al. [445] is the only study to evaluate recent LLMs on pun detection, explanation, and generation. They refined the SemEval dataset by removing duplicates and adding human-annotated explanations. While the study included a manual evaluation of a small set of generated explanations, it lacks a comprehensive error analysis and detailed evaluation guidelines. Their results indicate that, although LLMs perform reasonably well on binary pun detection, they struggle with pun generation and explanation, often relying on memorization or biased shortcuts. Building on this work, our study conducts a robustness analysis targeting specific biases and examines common errors in model-generated rationales.

8.3 Experimental Methodology

The primary pun-related tasks previously discussed in the NLP literature relate to detection, location, and interpretation. *Pun detection* is a binary classification task, categorizing texts as either containing a pun or not. *Pun location* involves identifying the specific pair of words responsible for creating the double entendre, while *pun interpretation* requires associating each identified pun word with its correct meaning or sense [413, 122, 220].

Preliminaries. Structurally, a pun is composed of a *pun word* (w_p), an *alternative word* (w_a) and their respective *senses* s_p and s_a [376]. For brevity, we will refer to (w_p,w_a) as the *pun pair*. Following previous literature [122, 445], we focus on *heterographic* and *homographic* puns.

In *heterographic* puns (or het-puns), only w_p appears in the sentence, while w_a and its sense are evoked through the context. For example, the het-pun "*I bought a boat because it was for sail (sale)*" creates a double meaning by playing on the homophones *sail* (to navigate on a boat) and *sale* (an occasion when items are sold).

In contrast, *homographic* puns (hom-puns) feature a polysemous w_p, resulting in

$w_p = w_a$. The hom-pun "*I was wondering why the ball was getting bigger; then it hit (hit) me*" plays on the two senses of *hit* (to be physically struck) and *hit* (to suddenly realize).

The term *rationale* has been used in the literature to denote various types of evidence justifying a decision, particularly in the form of natural language explanations [426]. This is consistent with its usage in prior research on puns [445, 377], although we do not limit it to unstructured or free-text explanations.

8.3.1 Datasets

Existing collections of puns annotated with the structure just described are limited. The SemEval 2017 dataset [122] provides 4,030 samples encompassing heterographic and homographic puns. Later research [377, 445] refined this dataset by adding sentiment annotations and removing incorrect examples. The English part of the JOKER corpus [412, 413] extends SemEval and adds 632 new English puns collected from PunOfTheDay.com. By removing or replacing a single contextual word, these new puns have been altered to create an equal number of non-puns. Both datasets contain pun words and senses annotations.

We utilize the dataset proposed by Xu et al. [445], which is the most refined iteration of the SemEval 2017 dataset, along with a subset of the JOKER dataset. In the former, we corrected several typographical errors and 13 incorrect annotations, resulting in 2,589 samples. We split the dataset into training (1,071 samples), testing (1,341), and validation (177) sets, ensuring that no w_p and w_a in the training set appear in the test and validation sets. We refer to this as the *PunEval* dataset. Unlike the SemEval dataset, the full JOKER corpus is not publicly available and was released only as part of the JOKER CLEF workshop [413]. However, the authors kindly agreed to share the complete dataset with us upon request. Since most examples in this dataset originate from the SemEval dataset, we retained the subset of 632 puns and 632 non-puns created by replacing a single word in the original pun [412]. Dataset statistics are reported in Table 8.1.

8.3.2 Comparison Models

We benchmark five recent instruction-tuned LLMs on pun understanding, including both open- and closed-weights models: GPT-4o-2024-08-06 (OpenAI) [448], Qwen2.5-72B (Alibaba) [451], Llama3.3-70B (Meta) [447], Gemini2.0-Flash (Google DeepMind) [475], and Mistral3-24B (Mistral AI) [404]. We will refer to these models as follows: GPT-4o, Qwen2.5, Llama3.3, Gemini2.0, and Mistral3. Additionally, we employed two reasoning models from the DeepSeek family: DeepSeek-R1 (R1) and DeepSeek-R1-Distill-Llama-70B (R1-D) [476], the latter being a distilled version of the full R1, based on Llama-3.3-70B-Instruct.

Table 8.1: Puns dataset statistics. Datasets in bold represent the new ones introduced in this work. *N. chars* and *N. words* denote the average number of characters and words per sample, respectively.

Dataset	Avg. Statistics		Labels Count			
	N. chars	N. words	Het-pun	Hom-pun	Non-pun	Total
PunEval (train)	58	13	279	304	488	1071
PunEval (test)	58	13	312	461	568	1341
PunEval (val)	59	13	56	46	75	177
NAP	67	16	64	64	128	256
JOKER	64	14	381	251	632	1264
PunnyPattern	64	15	212	388	600	1200
PunBreak	66	15	100	100	900	1100

8.3.3 Prompts

Unlike in Xu et al. [445], in this work, we leverage the pun structure described in § 8.3 and instruct the model to respond in a semi-structured format, facilitating parsing and the evaluation of the rationales.

We adopt the following response format that is compatible with the rationale: `(yes|no) [<`w_p`> <`w_a`> [<`s_p`> <`s_a`>]]`. In the configurations without rationales, the answer would simply be *yes* or *no* for the pun detection task. The content within the angular brackets represents the rationale (or justification) for the answer. Inspired by the success of CoT prompting [385, 389], we explore an alternative prompting strategy to elicit reasoning from non-reasoning LLMs by asking them to *think before answering*. Unlike previous cases where responses were restricted to a structured format, the reasoning prompts allow the model to provide free-text reasoning before delivering the final answer in the same structured format. In what follows, we describe the prompt configurations used.

Zero-Shot. (0S) This prompt asks to provide a "Yes" or "No" answer on whether the text contains a pun, with no formal definition of a pun.
Few-Shot. (FS) A definition of pun is provided, including the concepts of w_p, w_a, s_p, and s_a, followed by six examples (three puns, three non-puns) drawn from the same set used by Xu et al. [445]. Examples are held constant across experiments.
Words. (W) Same as *Few-Shot* prompt, but requires reporting the pun pair as justification.
Words+Senses. (W+S) This prompt mirrors the *Words* prompt, but additionally requires reporting s_p and s_a after the pun pair.
Reasoning. (R+) This configuration uses the four prompts just described, but allows the generation of arbitrary text before the final answer. This prompt is applied only to the five non-reasoning LLMs.

8.4 RQ1: How Well Can LLMs Detect Puns?

The first research question investigates LLMs' overall performance in binary pun detection. We thus evaluate all comparison LLMs described in § 8.3.2 using the prompting strategies outlined in § 8.3.3.

8.4.1 Data

In addition to the PunEval and JOKER datasets described in § 8.3.1, we annotate a new collection of 128 puns from various sources, including websites[3], personal recollections, and original creations. The main reason for creating this new dataset was to avoid any potential contamination from existing datasets. Each pun is rephrased into a non-pun, similar to the JOKER dataset, but without the requirement of replacing a single word, resulting in a total of 256 instances. We use Mistral3 to aid us in this rephrasing process, and all generated samples are manually inspected and modified as needed. We refer to this as the *Newly Annotated Puns* (NAP) dataset. A sample pun and its corresponding non-pun are shown in Example 1. Finally, we also annotate each pun with its pun pair (w_p, w_a) and respective senses (s_p, s_a), using short definitions from online dictionaries[4]. This process is consistent with the methods employed in the SemEval and PunEval datasets, and these annotations will be used to address RQ3.

Original Pun
What did the buffalo say to his son? Bison (*Bye son*).
Rephrased pun
What did the buffalo ask his son? I do not know.

Example 1: A pun and non-pun from the NAP dataset.

Table 8.2: Pun detection F_1 score ± SD over 3 runs. Top results are in bold.

Model	**Prompt**	**F_1 score (%)**		
		NAP	**JOKER**	**PunEval**
Gemini2.0	0S	73.3 ± 0.0	71.2 ± 0.0	85.7 ± 0.2
	FS	74.9 ± 0.0	74.0 ± 0.1	89.3 ± 0.1
	W	76.2 ± 0.4	74.4 ± 0.1	88.3 ± 0.1
	W+S	**77.9** ± 0.2	74.4 ± 0.0	89.1 ± 0.1
	R+FS	70.9 ± 0.6	71.1 ± 0.2	82.7 ± 0.3

Continued on next page

[3] https://parade.com/1024249/marynliles/funny-puns
[4] https://dictionary.cambridge.org/dictionary, https://www.merriam-webster.com

Table 8.2: – *Continued from previous page*

Model	Prompt	F_1 score (%)		
		NAP	JOKER	PunEval
	R+W	76.6 ± 0.0	74.4 ± 0.0	**90.6** ± 0.1
	R+W+S	75.4 ± 0.2	**74.6** ± 0.1	89.5 ± 0.1
GPT-4o	0S	78.4 ± 0.1	77.2 ± 0.0	89.7 ± 0.1
	FS	81.0 ± 0.7	79.7 ± 0.1	92.8 ± 0.2
	W	**86.9** ± 0.8	**83.3** ± 0.4	87.3 ± 0.9
	W+S	85.0 ± 0.8	82.6 ± 0.1	88.9 ± 0.9
	R+FS	82.9 ± 0.0	80.5 ± 0.3	92.3 ± 0.1
	R+W	84.0 ± 1.1	81.1 ± 0.2	92.6 ± 0.4
	R+W+S	82.9 ± 0.8	81.3 ± 0.3	**93.0** ± 0.3
Llama3.3 (70B)	0S	79.6 ± 1.5	77.1 ± 0.8	84.0 ± 0.4
	FS	81.0 ± 0.7	77.6 ± 0.1	84.4 ± 0.4
	W	83.4 ± 0.9	**79.2** ± 0.2	**87.2** ± 0.2
	W+S	**83.6** ± 0.9	77.9 ± 0.6	86.4 ± 0.3
	R+FS	79.2 ± 0.2	75.2 ± 0.1	83.8 ± 0.2
	R+W	78.1 ± 0.8	76.4 ± 0.1	85.1 ± 0.5
	R+W+S	78.3 ± 1.0	76.8 ± 0.2	84.5 ± 0.3
Mistral3 (24B)	0S	**75.0** ± 0.5	**75.3** ± 0.2	80.8 ± 0.2
	FS	69.5 ± 2.2	66.2 ± 1.2	74.6 ± 1.9
	W	69.0 ± 1.7	69.3 ± 0.4	78.4 ± 0.5
	W+S	68.5 ± 0.9	68.5 ± 0.4	74.9 ± 0.9
	R+FS	73.6 ± 1.1	70.9 ± 0.1	82.4 ± 1.3
	R+W	70.7 ± 0.3	69.0 ± 0.4	84.2 ± 0.4
	R+W+S	70.9 ± 0.2	70.1 ± 0.3	**84.5** ± 0.2
Qwen2.5 (72B)	0S	71.3 ± 0.4	73.6 ± 0.2	83.6 ± 0.1
	FS	78.6 ± 0.4	76.4 ± 0.7	87.2 ± 0.2
	W	**81.1** ± 0.0	77.1 ± 0.4	86.8 ± 0.4
	W+S	80.7 ± 0.6	76.8 ± 0.8	87.3 ± 0.1
	R+FS	80.2 ± 0.7	**77.5** ± 0.6	88.5 ± 0.6
	R+W	77.2 ± 0.2	75.0 ± 0.1	**89.4** ± 0.5
	R+W+S	77.0 ± 1.1	75.8 ± 0.6	88.7 ± 0.1
R1-D (70B)	0S	74.8 ± 0.5	75.3 ± 0.6	86.7 ± 0.3
	FS	76.3 ± 2.7	76.2 ± 0.5	87.0 ± 0.3
	W	**79.8** ± 1.4	78.1 ± 0.9	86.8 ± 0.4
	W+S	78.7 ± 1.2	**78.7** ± 0.6	**88.3** ± 0.6
R1 (671B)	0S	74.2 ± 0.4	75.5 ± 0.5	88.5 ± 0.2
	FS	76.6 ± 0.5	78.4 ± 0.4	90.8 ± 0.2
	W	77.7 ± 0.4	77.7 ± 0.1	90.9 ± 0.1

Continued on next page

Table 8.2: – *Continued from previous page*

Model	Prompt	F_1 score (%)		
		NAP	**JOKER**	**PunEval**
	W+S	**79.1** ± 0.8	**79.2** ± 0.0	**91.3** ± 0.4
RoBERTa-L		64.0 ± 0.1	65.7 ± 0.1	92.5 ± 0.3

8.4.2 Results

The average F_1 score for these datasets across 3 runs is shown in Table 8.2. All LLMs achieved F_1 scores around 0.8 on the NAP dataset with the best prompting strategy. GPT-4o performed the best, while Mistral had the lowest scores. Notably, Mistral only performed well in the zero-shot setting and struggled to learn from additional context in the prompt.

Results on the JOKER dataset show a similar trend but are slightly more challenging for all models. This is likely due to its method of creating non-puns by replacing a single word, resulting in more "adversarial" examples. In contrast, in the NAP dataset, we rephrased puns into non-puns, potentially replacing or removing multiple words.

Aside from Mistral, which consistently performs poorly on our new datasets, the W/W+S prompts — which require LLMs to justify each decision with a rationale — achieve the highest performance, with an average improvement of +3% in F_1 score compared to the few-shot configuration. This finding aligns with previous research [377] that shows explanations can enhance pun detection performance. However, the reasoning prompt did not yield significant improvements over the rationale-augmented prompts (W and W+S) on the NAP and JOKER datasets. Overall, the strong performance suggests that LLMs can understand puns to some extent and benefit from jointly detecting and explaining.

Embedding space analysis In addition to the comparison LLMs, we also include a RoBERTa-large encoder model [194] that has been fine-tuned on the PunEval training split. The significant performance difference observed between PunEval (92.5) and the other datasets (64.0 in NAP and 65.7 in JOKER) prompted an investigation to understand the reasons behind this disparity. We analyzed the embedding distribution of the RoBERTa model using t-SNE [47] to visualize the embeddings in a 2D space (Fig 8.2). The visualizations reveal poor separation between subtly altered positive and negative examples in the NAP and JOKER datasets (Fig 8.2, right). In contrast, PunEval embeddings show better separation even before fine-tuning (Fig 8.2, center), with clusters often corresponding to recurring templates. For instance, the top-left cluster represents puns structured as "Old [...] never die, they just", a common pattern for creating puns (e.g., "*Old bankers never die, they*

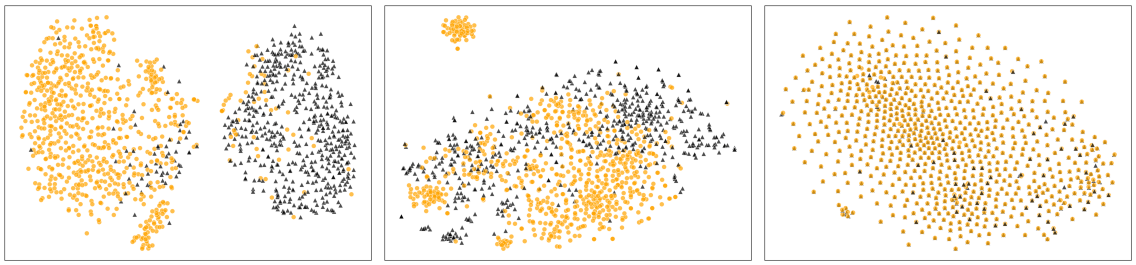


Figure 8.2: Separation of PunEval embeddings for the RoBERTa model: after fine-tuning (left), before fine-tuning (middle), and NAP+ JOKER separation before fine-tuning (right). Orange dots represent puns, while black triangles represent non-puns. Notably, PunEval puns exhibit greater separability in the embedding space compared to NAP/JOKER, even before fine-tuning.

just lose interest"). This suggests that dataset artifacts or structural similarities in PunEval may facilitate shortcut learning.

8.5 RQ2: How Robust Are LLMs at Detecting Puns?

Given the analysis in the previous section, one may wonder whether the performance gap between the PunEval dataset and the other two datasets is partly due to LLMs recognizing superficial cues or overfitting to specific pun structures without fully understanding their meanings. This observation prompts us to investigate the LLMs' robustness to common pun language patterns and simple pun alterations.

8.5.1 Pun Language Patterns

The PunEval dataset contains language patterns frequently used to create puns. Examples include "*Old [...] never die, they just [...]*" and "*Tom*" (a common name in English jokes), which may serve as shortcuts for recognizing puns.

To analyze the impact of these recurrent patterns on model performance, we extracted bag-of-n-gram features and trained a logistic regressor for pun detection on the PunEval training set. By examining the top-20 expressions based on learned coefficients and frequency, we manually identified six patterns that significantly correlate with the presence of a pun. These patterns, along with examples and their frequencies, are listed in Tables 8.3 and 8.4. While their presence has been noted previously [412], no detailed analysis of their effects has been provided.

We suspect that these language patterns simplify the task of detecting puns, as texts containing these expressions are often intended to be humorous. In the PunEval test split, 171 out of 173 samples that contain these patterns are puns. Excluding them results in a 2–3% drop in F_1 score and a 2–6% drop in precision across all models compared to the full test split, confirming that the presence of

Table 8.3: Pun language patterns and their occurrences in the PunEval dataset.

Structure	Short name	Example
Old [...] never die [...] they	`never_die`	OLD BANKERS never die they just lose interest
Tom	`tom`	"I've been to a film festival in Southern France" said Tom cannily
When the	`when`	When the TV repairman got married the reception was excellent
She was only [...] daughter but	`daughter`	She was only a horseman's daughter, but she didn't know how to say neigh
Doctor, Doctor	`doctor`	Doctor, Doctor, I keep thinking I'm a spoon. - Sit there and don't stir
used to [...] but	`used`	I used to be addicted to soap, but I'm clean now

Table 8.4: Occurrences of pun language patterns in all the datasets.

Pattern	Occurrences					
	PunEval (train)	PunEval (test)	PunEval (val)	NAP	JOKER	PunBreak
`never_die`	65	62	1	2	12	50
`tom`	61	61	1	0	0	15
`when`	13	20	0	0	22	5
`daughter`	11	19	1	0	0	15
`doctor`	3	7	0	0	0	0
`used`	2	4	1	2	10	10

these recognizable patterns can inflate performance. To further investigate this issue, we created a new benchmark to test robustness against this bias.

8.5.1.1 Data

For each of the six extracted patterns, we sampled 100 puns from the PunEval test split that contain the same expression, supplementing with new examples collected from the Internet or generated using GPT-4o (verified manually) as needed to complete the set. We also generated an equal number of verified non-pun sentences using GPT-4o (with human oversight), ensuring they contain the same expression. The resulting dataset, referred to as the *PunnyPattern* dataset, contains 1,200 samples, two of which are presented in Example 2.

Original pun

I used to be a comedian, but my life became a joke (*joke*).

Rephrased pun

I used to be a comedian, but my life became chaotic.

Example 2: A pun with a common pun language pattern and a derived non-pun.

Table 8.5: F_1 score, precision, and recall (%) on the best prompt on PunnyPattern and the absolute performance difference with the PunEval dataset.

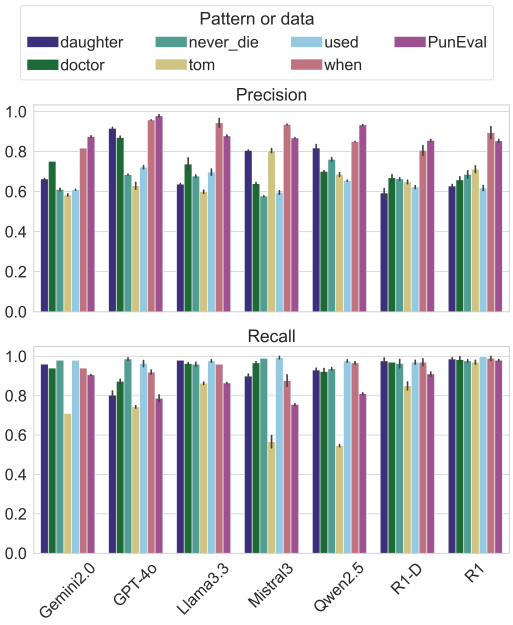
Model	Best prompt	**F_1**	**Precision**	**Recall**
Gemini2.0	W+S	76.9 (-12.2)	66.7 (-20.9)	91.6 (+0.9)
GPT-4o	W	**83.1** (-4.2)	**79.7** (-18.2)	88.1 (+9.4)
Llama3.3	W	81.0 (-6.2)	71.5 (-16.4)	94.2 (+7.7)
Mistral3	0S	77.4 (-3.4)	72.1 (-14.8)	87.5 (+11.9)
Qwen2.5	W	79.7 (-7.1)	74.1 (-19.3)	87.5 (+6.5)
R1-D	W+S	78.3 (-10.0)	66.9 (-18.7)	94.8 (+3.7)
R1	W+S	81.1 (-10.2)	69.5 (-15.9)	**98.3** (+0.3)

8.5.1.2 Results

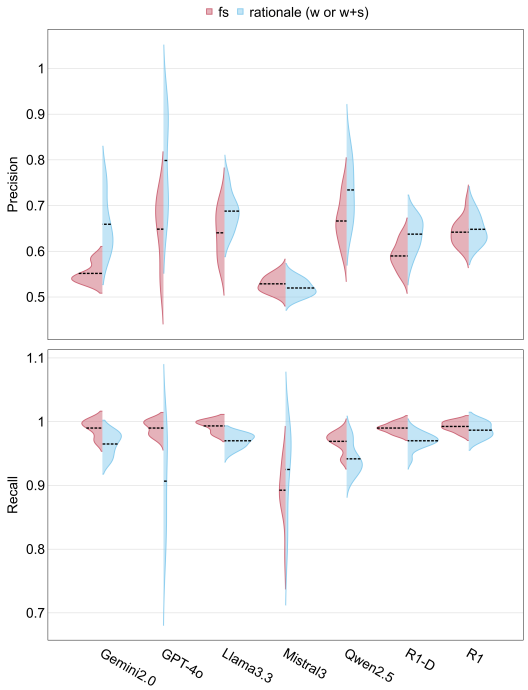
Table 8.5 compares the average performance metrics over 3 runs between the entire PunnyPattern dataset and the PunEval dataset. A visual comparison of the results over the single patterns is shown in Fig 8.3a.

Overall, there is an average drop of 16–23% in precision and 4–13% in F_1 on the PunnyPattern dataset across all models. Additionally, we observe a sharp imbalance of low precision and high recall for the `never_die`, `used`, and `doctor` patterns (Fig 8.3a), indicating that these patterns tend to mislead LLMs into identifying puns. In contrast, the `tom` and `when` patterns do not appear to consistently affect them. These results suggest that most LLMs tend to process certain patterns somewhat superficially, lacking understanding of the underlying principles of well-crafted puns.

To showcase the effect of rationale on detection, Fig 8.3b compares the impact of the best rationale-augmented prompt (W or W+S) on pun-detection performance over the PunnyPattern dataset, excluding the `tom` and `when` patterns that do not consistently bias the tested LLMs. The best prompt used is the one highlighted in Table 8.5. In the violin plots, the right side (light blue) represents the distributions of results over the three runs with the rationale-augmented prompt, while the left side (red) represents performance using the FS prompt — typically the best prompt without rationales. Note that in all cases except Mistral, incorporating rationales reduces the number of false positives, thereby lowering recall and improving precision, countering the bias effect.



(a) Precision and recall using the best prompt (all patterns).



(b) Precision and recall comparison between `fs` prompt and the best prompt between `w` and `w+s`.

Figure 8.3: Performance on the PunnyPattern dataset and comparison showing the rationale effect on detection.

8.5.2 Pun Alterations

Motivated by our previous results, we investigate whether LLMs exhibit robustness to simple alterations of puns designed to ruin them. For humans, the pun effect relies on the perception of two well-supported senses that create ambiguity and contrast. Therefore, replacing the pun word (w_p) with a lexical unit that fails to convey this duplicity would result in the loss of such effect, and the failure of the intended pun.

8.5.2.1 Data

To systematically investigate LLMs' understanding of this phenomenon, we replace the pun word w_p with multiple alternative expressions and measure their ability to classify "ruined" puns. To this end, we randomly select 100 heterographic and 100 homographic puns from the NAP and PunEval datasets and modify each pun with four different substitutions:

- `Pun syn`: a synonym or hypernym of the pun word w_p, different in phonetics and orthography.

Original pun
Long fairy tales have a tendency to dragon (*drag on*).
`Pun syn` **substitution**
Long fairy tales have a tendency to wyvern.
`Alt syn` **substitution**
Long fairy tales have a tendency to prolong.
`Homophone` **substitution**
Long fairy tales have a tendency to brag gone.
`Random` **substitution**
Long fairy tales have a tendency to tick.

Example 3: Example substitutions from the PunBreak.

- `Alt syn`: as above, but for the alternative word w_a.
- `Homophone`: a nonsensical expression phonetically similar to w_p.
- `Random`: a nonsensical random word.

Additionally, we include 100 randomly generated sentences (non-puns) as a control group, resulting in a total of 1,100 examples, which we refer to as the *PunBreak* dataset. A complete set of substitutions is shown in Example 3. Note that while some nonsensical replacements may be humorous due to the absurdity of the resulting sentences, they do not represent valid puns.

8.5.2.2 Results

The binary accuracy of each LLM for every substitution subset, using each model's best prompt from the RQ1 experiments, is shown in Fig 8.4. The "pun" subset contains exactly 200 puns (100 het-puns and 100 hom-puns); every other subset contains exactly 200 non-puns representing the various substitutions. Results from single prompts are reported in Table 8.6. The "rand sent" results are obtained from a control of 100 randomly generated non-puns to detect any unseen bias toward the pun class; scores exceed 0.8, indicating no such bias in the random sentences.

All models exhibit a bias toward labeling altered examples as puns, reflected in low accuracy across all substitution categories, which contain only negative examples (non-puns). In particular, GPT-4o, the best performer overall, reaches a low of 0.33 on the `homophone` subset and a high of 0.59 on `alt syn`. Conversely, all models achieve accuracy of around 0.8 or higher on the negative examples from PunEval and the control group (`rand sent`) (see Table 8.6 for details). The `homophone` subset is the most challenging overall, indicating that phonetic similarity between w_p and its replacement degrades the LLMs' ability to distinguish them in context. This

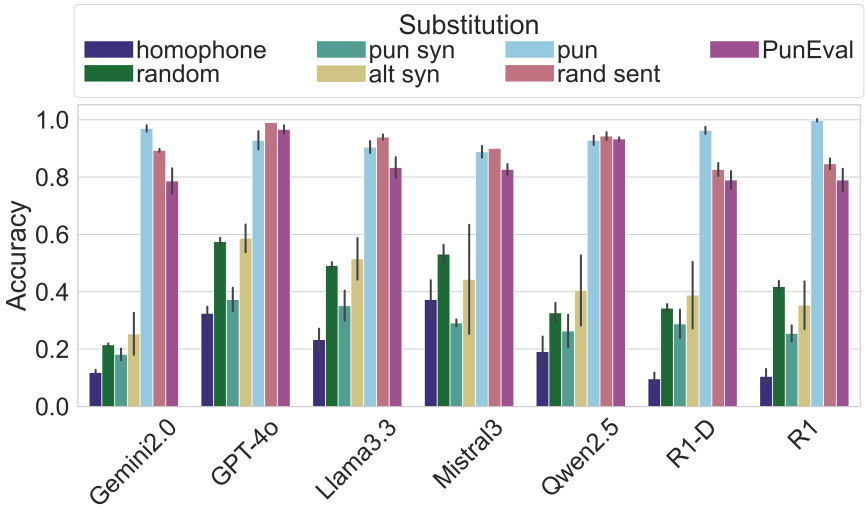


Figure 8.4: Binary accuracy of comparison LLMs on the PunBreak dataset, with their best prompt, with error bars representing the SD over 3 runs.

may also be influenced by orthographic similarity, as similar phonetics often imply similar orthography (e.g., *stock*/*stork* or *dragon*/*brag gone* in Example 3). Pun-word synonym substitutions (`pun syn`) also generally challenge the models, suggesting they may overlook the effect of the replaced word when it is semantically similar to a more fitting alternative. This indicates that LLMs might be actively "fixing" the pun in their reasoning process, thereby rectifying the alteration. The `random` and `alt syn` sets show slightly higher accuracy overall, further indicating that semantic or phonetic similarity to valid puns misleads predictions more than loosely related substitutions. Overall, these results suggest that LLMs fail to recognize contexts that should not trigger the *pun effect* in cases where the target sentence has a typical pun structure.

8.6 RQ3: To What Extent Can LLMs Explain Puns?

Our rationale-augmented prompts achieve the best performance in our previous experiments, notably reducing bias on the PunnyPattern dataset with a 9.8% increase in precision. In this section, we systematically examine the quality of the generated rationales to gain insights into LLMs' understanding of puns.

8.6.1 Keyword Evaluation

Our goal is to measure the overlap between human-provided semi-structured rationales and LLM-generated rationales, while also accounting for prediction accuracy. To this end, we quantify the alignment between the *LLM-predicted pun pair* (w'_p, w'_a) and the true pun/alternative word pair (w_p, w_a) using the *pun pair agreement*

Table 8.6: Percentage recall (true class) ± SD for each substitution category on the PunBreak dataset. Note that for all columns except JOKER, recall equals accuracy because each evaluation set contains only puns or only non-puns. JOKER reports recall for the negative class on the full dataset. The best overall prompt per model is marked in bold.

Mod.	Prm.	Substitution Category					Control groups	
		Homophone	Random	Pun syn	Alt syn	**Pun**	**Rand sent**	**JOKER**
Gemini2.0	0s	**23.7** ± 5.5	**37.0** ± 1.5	11.5 ± 0.5	18.0 ± 5.5	**99.0** ± 1.1	70.3 ± 0.5	26.3 ± 0.1
	fs	9.3 ± 0.8	16.2 ± 1.0	14.7 ± 0.5	20.3 ± 5.9	98.0 ± 1.1	85.7 ± 0.5	41.3 ± 0.2
	w	10.0 ± 1.1	14.5 ± 4.9	13.2 ± 1.0	20.3± 7.7	97.5 ± 1.4	87.0 ± 0.0	43.8 ± 0.3
	w+s	11.8 ± 1.0	21.5 ± 0.5	**18.2** ± 2.0	**25.3**± 7.3	97.0 ± 1.1	**89.3** ± 0.5	**47.9** ± 0.1
GPT-4o	0s	9.2 ± 1.3	23.8 ± 1.6	10.7 ± 0.5	23.2 ± 4.9	**99.2** ± 0.4	87.7 ± 1.4	47.4 ± 0.1
	fs	12.5 ± 3.1	26.2 ± 3.5	20.5 ± 0.5	32.5 ± 9.1	97.2 ± 1.2	91.3 ± 0.5	57.8 ± 0.6
	w	**32.5** ± 2.3	**57.5** ± 1.4	**37.3** ± 4.1	**58.7**± 4.8	92.8 ± 3.2	**99.0** ± 0.0	**77.4** ± 0.9
	w+s	24.0 ± 0.9	39.5 ± 5.5	30.0 ± 1.5	49.0 ± 5.9	93.5 ± 3.6	97.7 ± 0.5	71.8 ± 0.6
Llama3.3	0s	22.0 ± 1.4	45.5 ± 3.4	16.5 ± 2.7	35.2 ± 11.1	**96.8** ± 1.5	79.7 ± 2.1	50.6 ± 3.0
	fs	16.3 ± 6.0	37.0 ± 4.3	21.8 ± 1.5	40.3 ± 13.0	96.7 ± 1.6	88.7 ± 1.0	53.2 ± 0.1
	w	**23.3** ± 3.8	**49.2** ± 1.2	**35.2** ± 5.2	**51.5**± 7.2	90.5 ± 2.1	**94.0** ± 0.9	**63.9** ± 0.7
	w+s	13.2 ± 3.8	32.8 ± 5.9	26.5 ± 2.7	42.5 ± 7.6	92.3 ± 1.2	93.0 ± 0.0	56.0 ± 1.6
Mistral3	**0s**	**37.3** ± 6.7	**53.2** ± 3.3	**29.2** ± 1.2	**44.3**± 19.0	88.8 ± 2.0	**90.0** ± 0.0	**61.9** ± 0.2
	fs	14.5 ± 4.6	17.0 ± 5.8	15.8 ± 2.0	21.7 ± 5.0	86.7 ± 2.4	37.7 ± 1.4	26.4 ± 0.2
	w	5.2 ± 1.9	9.2 ± 1.5	12.3 ± 2.0	19.2 ± 5.1	93.3 ± 3.4	48.7 ± 3.1	24.1 ± 1.6
	w+s	2.7 ± 1.4	3.5 ± 2.5	6.5 ± 1.8	9.5 ± 3.4	**96.7** ± 1.5	25.7 ± 4.2	14.0 ± 1.7
Qwen2.5	0s	5.0 ± 0.6	11.2 ± 3.7	5.7 ± 0.8	11.3 ± 3.8	**98.2** ± 0.4	59.3 ± 2.1	36.6 ± 0.3
	fs	11.3 ± 3.4	14.7 ± 2.8	19.0 ± 4.6	33.7 ± 11.7	94.7 ± 1.0	90.0 ± 0.9	56.5 ± 0.9
	w	**19.2** ± 5.2	**32.7** ± 3.4	26.3 ± 5.6	**40.5**± 12.2	92.8 ± 1.6	**94.3** ± 1.4	**65.7** ± 0.5
	w+s	15.5 ± 4.4	29.8 ± 5.8	**27.0** ± 3.6	37.2 ± 11.9	95.0 ± 1.4	92.7 ± 1.4	62.7 ± 1.1
R1-D	0s	6.2 ± 1.2	24.9 ± 6.6	14.2 ± 1.2	21.2± 8.1	**99.3** ± 0.5	59.0 ± 2.4	37.5 ± 2.2
	fs	7.1 ± 2.3	27.3 ± 3.8	18.8 ± 4.5	26.9 ± 9.8	98.5 ± 0.5	69.7 ± 5.5	42.9 ± 1.3
	w	8.5 ± 3.7	**36.9** ± 3.1	26.7 ± 2.7	34.6± 8.3	96.5 ± 2.0	79.3 ± 5.2	52.1 ± 1.4
	w+s	**9.7** ± 2.2	34.3 ± 1.4	**28.8** ± 5.0	**38.8**± 11.6	96.3 ± 1.2	**82.7** ± 2.3	**55.5** ± 3.3
R1	0s	6.5 ± 2.7	27.3 ± 4.1	12.2 ± 2.0	18.8 ± 7.3	99.8 ± 0.4	71.3 ± 1.0	37.0 ± 1.7
	fs	8.5 ± 2.3	38.8 ± 3.7	22.3 ± 2.9	31.8 ± 4.1	99.3 ± 0.8	83.0 ± 0.9	48.2 ± 1.5
	w	9.5 ± 1.8	40.3 ± 2.7	22.3 ± 3.1	34.0 ± 6.0	99.8 ± 0.4	84.7 ± 2.3	44.9 ± 0.5
	w+s	**10.5** ± 2.6	**41.8**± 2.0	**25.5** ± 2.8	**35.3**± 8.3	**99.8** ± 0.4	**84.7** ± 1.9	**51.2** ± 0.1

(PPA) metric. Each prediction is scored: 2 if both w_p and w_a are correctly generated, 1 if only one is correct, and 0 if neither is correct. Predictions with incorrect labels receive a score of 0, since no rationales are generated for non-pun predictions.

Table 8.7: Average PPA, in [0 – 2], for each prompt setting (W, W+S) across datasets. SD is 0.0 for all measurements, and the best results are marked in bold.

Model	NAP		JOKER		PunEval	
	W	W+S	W	W+S	W	W+S
Gemini2.0	1.2	1.2	1.1	1.1	1.5	1.5
GPT-4o	**1.5**	**1.5**	**1.5**	**1.4**	**1.6**	1.6
Llama3.3	1.4	1.3	1.3	1.2	1.5	1.5
Mistral3	0.8	0.7	0.8	0.7	1.2	0.9
Qwen2.5	1.2	1.2	1.2	1.2	1.5	1.5
R1-D	1.3	1.2	1.2	1.2	1.5	1.6
R1	1.3	1.2	1.2	1.3	**1.6**	**1.7**

Furthermore, we also conduct a PPA evaluation restricted to the true positives (i.e., puns correctly identified by the LLMs), which allows us to isolate the quality of the rationales from detection performance.

Results Table 8.7 reports the average PPA over three runs, representing the mean number of words correctly identified in generated rationales for pun pairs on a 0–2 scale. GPT-4o is the top performer, with ~1.5 out of 2 correctly predicted words per pun across all three datasets. It is followed by Llama3.3 and DeepSeek-R1, whose performance is noticeably lower. Mistral has the lowest scores, trailing the next-lowest model, Gemini2, by ~33%. Among the 70B models, Llama3.3 performs best on average. We also observed no significant difference between the two prompts tested. Consistent with our earlier results, scores on the PunEval datasets are generally higher than on JOKER and NAP, where performance drops by approximately 20% and peaks at 1.5 out of 2.

The PPA evaluation restricted to true positives is reported in Table 8.8 and shows that most models, except Qwen and Mistral, can effectively justify their correct answers. In this evaluation, GPT-4o and DeepSeek-R1 are the top performers by a clear margin. The results also indicate that even the best LLMs (GPT-4o, Gemini, and R1) produce correct predictions for the wrong reasons at least 20% of the time, while other models have up to 40% incorrect rationales.

These findings motivate our investigation of the main error types in model reasoning, which is addressed in the next subsection.

8.6.2 Error Analysis: Manual Evaluation

The PPA metric measures the ability to match the original pun pair but does not indicate *why* predictions were wrong, nor does it evaluate the complete rationale obtained with the more expressive W+S prompt. Upon inspection of the produced rationales, we identify four main categories of mistakes frequently made by the

Table 8.8: Average PPA measured on the true positive examples, in [0 – 2], for each prompt setting (W, W+S) across datasets. SD is 0.0 for all measurements, and the best results are marked in bold.

Model	NAP		JOKER		PunEval	
	W	W+S	W	W+S	W	W+S
Gemini2.0	**1.6**	**1.5**	1.4	1.4	1.6	1.6
GPT-4o	1.5	**1.5**	**1.6**	1.5	**1.8**	**1.8**
Llama3.3	1.4	1.4	1.4	1.4	1.6	1.6
Mistral3	1.2	1.2	1.3	1.2	1.4	1.3
Qwen2.5	1.3	1.3	1.3	1.4	1.5	1.6
R1-D	1.5	**1.5**	1.5	1.5	1.7	1.7
R1	1.5	**1.5**	**1.6**	**1.6**	**1.8**	**1.8**

seven LLMs regarding the predicted pun pair (w'_p, w'_a) and the associated pair of senses (s'_p, s'_a):

- **Word-sense association** (`word-sense` error): incorrect association between word and sense (i.e., w'_p–s'_p or w'_a–s'_a);
- **Missing context** (`context`): the interpretation of w'_p as s'_p or w'_a as s'_a is not supported by the context;
- **Pun pair** (`pun pair`): w'_p and w'_a are not sufficiently similar in phonetic or orthographic terms to create wordplay (this only applies to het-puns);
- **Senses similarity** (`sense sim`): senses s'_p and s'_a are too similar. Puns work when polysemous words carry well-separated senses that create contrast.

These types of mistakes also highlight different weaknesses. A word-sense mistake can be considered a hallucination (e.g., "crustacean" interpreted as "a species of bird") while a pun pair mistake (as the initial example in Fig 8.1) indicates difficulty in grasping the phonological properties of words.

To precisely characterize the model's mistakes in the categories above, we conducted an error analysis with five native English speakers specifically hired for this task. We randomly sampled 80 incorrect answers from each of the three top-performing LLMs, according to the PPA evaluation in Table 8.7: GPT-4o, Llama3.3, and DeepSeek-R1. For each of the sampled answers, we designed a set of questions to determine the presence of each category of mistake. For example, the first two questions assess whether s'_p (the LLM-predicted pun sense) and s'_a (the predicted alternative sense) are legitimate interpretations of w'_p (predicted pun word) and w'_a (predicted alternative word), respectively. Annotators could answer with "Yes", "No", or "Maybe" if uncertain. If any of these questions is answered with "No" for a given sample, that sample is counted as a `word-sense` association error. Note that if both

senses are incorrect, it still counts as a single error, ensuring a more comparable error count across all mistake categories.

We report in Table 8.9 various examples of the error analysis on real data, along with human annotators' questionnaire responses. Note that, for brevity, the questionnaire items shown in the table have been slightly shortened.

Results Table 8.10 reports the results of the manual evaluation for the three analyzed LLMs. Our analysis reveals that, in general, the most frequent errors involve: (i) missing the required context to support both s_p and s_a (`context`) and (ii) selecting unsuitable word pairs that do not fit into a pun (`pun pair`). The latter category only applies when $w'_a != w'_p$, and typically results from forcing words that are not phonetically compatible to create wordplay. A third fairly common mistake arises from incorrectly pairing words with their meanings (`word-sense`). Llama3.3, the smallest among the three LLMs, tends to struggle the most in this area, producing more sense hallucinations. In contrast, GPT-4o and R1 tend to assess the senses correctly, even when some words in the pun pair are incorrectly identified.

These error patterns suggest a lack of understanding of the mechanisms of puns, particularly the inability to distinguish inappropriate contexts and difficulties in handling wordplay based on phonetic or orthographic similarity. Among the three evaluated models, DeepSeek-R1 proved to be the most precise, exhibiting the lowest number of errors in its responses.

8.6.3 Discussion

Our findings align with prior work showing that LLMs struggle with humor, particularly when it depends on nuanced context or roleplay [469, 428]. Interestingly, these studies suggest that safety constraints (e.g., the "Harmless, Helpful, Honest" criteria) can cause LLMs to misinterpret relational context in potentially offensive situations. Specifically, these safety mechanisms bias models toward a safer, "washed-out" form of humor, stripping away the surprise and edge on which many jokes depend. Puns, in contrast, are rich in surprise, subtlety, and timing; they are also deeply rooted in human emotions and experiences, qualities that current LLMs cannot fully tap into.

LLMs also exhibit a form of "regressive sycophancy" — a tendency to agree with a user prompt even when incorrect — particularly in ambiguous or sensitive contexts [441, 466]. We suspect that this tendency causes the models to produce false positives by forcing inputs into a "pun" template. Such behavior is a known artifact of human-preference alignment (post-training), which can bias models toward agreeable responses at the expense of contextual and factual accuracy [464].

Finally, the frequent `pun pair` errors point to weaknesses in assessing phonetic and orthographic similarity. These issues are likely tied to tokenization, which can mask morphological elements critical to wordplay, and to limited phonological modeling [478]. The composition of pre-training corpora and data cleaning practices

Table 8.9: Examples of error-analysis results for the three models analyzed in RQ3. The final column lists the specific error types used in our statistics, with the logical expression in brackets explaining how each error was computed from the annotators' answers. For example, 'X&Y' specifies that an error is counted if a 'no' answer was given to either question X or question Y. Answers marked in red indicate the model's errors.

Input	Output	Answers by annotators	Errors
		GPT-4o	
German sausage jokes are just the sausage	yes <the sausage> <the wurst> <the main part or essence of something> <a type of German sausage>	• A. Can "the sausage" be interpreted as "the main part or essence of something"? **No** • B. Can "the wurst" be interpreted as "a type of German sausage"? Yes • E. Does the context support the interpretation of "the wurst" as "a type of German sausage"? Yes • FG. Are "the sausage" and "the wurst" orthographically or phonetically close? **No** • H. Rate pun quality in 0 – 5: 1	`word-sense` ($A\&B$), `pun pair` ($F\|G$)
		Llama3.3	
The sailor called his girlfriend's belly ring an oven destroyer	yes <oven destroyer> <over destroyer> <a device that ruins an oven> <one who destroys from above>	• A. Can "oven destroyer" be interpreted as "a device that ruins an oven"? Yes • B. Can "over destroyer" be interpreted as "one who destroys from above"? **No** • D. Does the context support the interpretation of "oven destroyer" as "a device that ruins an oven"? **No** • FG. Are "oven destroyer" and "over destroyer" orthographically or phonetically close? Yes • H. Rate pun quality in 0 – 5: 0	`word-sense` ($A\&B$), `context` ($D\&E$)
		DeepSeek-R1	
Why are frogs so happy? They eat whatever feather them	yes <feather> <fed them> <to cover with feathers> <to provide food for them>	• A. Can "feather" be interpreted as "to cover with feathers"? **No** • B. Can "fed them" be interpreted as "to provide food for them"? Yes • E. Does the context support the interpretation of "fed them" as "to provide food for them"? Yes • FG. Are "feather" and "fed them" orthographically or phonetically close? Yes • H. Rate pun quality in 0 – 5: 2	`word-sense` ($A\&B$)

Table 8.10: Error statistics by category and overall on 80 misclassified samples from the PunBreak dataset. Note that there may be more than one error per sample.

Error Breakdown by Category			
Error Category	**GPT-4o**	**Llama3.3**	**R1**
`context`	66	**49**	50
`pun pair`	50	39	**27**
`word-sense`	9	17	**8**
`sense sim`	**2**	3	**2**
Aggregated statistics			
Total errors	128	111	**87**
Avg. errors	1.6	1.4	**1.2**

may also deprive models of the informal, creative language where puns are common. Because developers rarely disclose detailed preprocessing and corpus information, it is difficult to determine how these practices affect fluency with such language.

Improving an AI's grasp of puns and humor in general will likely require deeper human-machine collaboration and greater social awareness. Prior work suggests that human alignment of LLMs should shift from a single global standard to a more granular, audience-aware approach [469, 428]. This would help models to adapt to specific audiences and contexts, adjusting appropriateness and fairness criteria accordingly.

8.7 Limitations and Opportunities

A limitation of our work is that, despite our efforts to create high-quality data, the newly contributed datasets have not been comprehensively validated by language experts. Our manual error analysis indicates the possibility of incorrect annotations or "accidental" puns, stemming from the inherently subjective nature of pun perception. Discussions with annotators also revealed that the definition of a pun should be more formally characterized; we therefore provided many examples before annotation and corrected some samples based on their feedback. Despite these efforts, our manual error analysis remains limited by the subjectivity of several questions.

Additionally, there is the possibility that most LLMs have previously encountered many examples from our datasets, as puns often exist in various versions and can be easily found online. Consequently, we may not have fully disentangled this effect from our bias measurements on the PunBreak and, in particular, the PunnyPattern dataset, the latter of which includes many examples sourced from PunEval. We believe that further tests — such as injecting the identified language patterns into arbitrary sentences — could help clarify the impact of model bias on performance.

Future work should also investigate the severity and correctability of these biases. For example, it remains to be tested whether leveraging ICL through prompting with targeted negative examples, such as those from our new datasets, can mitigate the models' tendencies to classify most examples as puns. Moreover, the fact that our evaluation was conducted solely on English puns further limits the scope of the conclusions that can be drawn from these results.

Our results show a positive effect of asking for explanations on detection performance, but we do not analyze how well those explanations align with the model. Evaluating these generated rationales may be flawed because LLMs could be merely producing agreeable answers rather than genuinely describing their reasoning. Consequently, assessing the faithfulness of such rationales is an important research direction and has been questioned in recent work [419, 319].

Finally, our prompt design was tailored for Llama 3.1, and we utilized the same prompt for all LLMs. We acknowledge that custom-tailored prompts for each model are likely to impact performance. During this work, several new LLMs were published, and while we tested some of them, we did not have the opportunity to extensively optimize prompts for each one.

8.8 Conclusion

Benchmarks in pun detection often suggest high performance from current LLMs. However, it remains unclear whether this performance stems from a true understanding of linguistic mechanisms or from exploiting superficial cues in the data. To assess robustness, we challenged state-of-the-art LLMs with two new annotated datasets that we publicly released for future research: PunnyPattern, which collects examples with language patterns common in English puns, and PunBreak, containing sentences subtly modified to mimic puns. Although LLMs can detect puns on existing benchmarks (> 0.83 average accuracy), accuracy drops substantially on our more challenging datasets: -15% on PunnyPattern, and -50% on PunBreak, indicating limited understanding of pun mechanisms and over-reliance on similarity with known puns. We used LLMs to explain puns by generating semi-structured rationales and evaluated those explanations automatically and manually. Only three models correctly identified the pun-related words in more than 70% of cases. Moreover, all models struggled to recognize when context and word choice legitimately supported wordplay. Finally, we found that combining rationale generation with the detection task improved performance in all but one case, which invites further study into prompting techniques to counter negative biases. This research underscores the need for more rigorous evaluations of LLMs' reasoning capabilities, especially for tasks requiring subtle linguistic insight and emotional intelligence, where current models may not yet be trustworthy.

This analysis concludes the exploration of XAI and rationale generation, showing that a LLM's impressive performance can mask a shallow, non-human-like grasp of

a specific form of nuanced language. The fundamental misalignment between the internal logic of LLMs and human reasoning poses a significant challenge to their use in sensitive contexts. Further research into their rationalization and reasoning capabilities, however, may provide a pathway toward improved explainability.

9

Conclusion

My doctoral research has explored the intersection of two pivotal areas in modern Artificial Intelligence (AI): the development of Text Classification (TC) techniques using deep-learning-based Language Models (LMs), and the design of Explainable AI (XAI) strategies to render these models more transparent.

Part I established the foundational concepts for the two main themes of my research by providing a comprehensive literature review. First, Chapter 2 detailed the complete TC pipeline, from preprocessing and tokenization to classification, with specific considerations for handling multilingual content. This review provided the necessary knowledge of foundational neural architectures and paradigms used in Natural Language Processing (NLP) that supported the subsequent research. Chapter 3 then introduced the field of XAI, providing an overview of methods, taxonomies, and evaluation criteria related to XAI. This included a discussion of key terminology and an overview of the main types of methods.

Part II detailed my contributions to the field of NLP, beginning in Chapter 4 with a benchmark of LMs in a multilingual scenario. This study evaluated several state-of-the-art Transformer-based LMs and traditional methods on two multilabel classification tasks: topic labeling on a novel corpus derived from Wikipedia, and news categorization using the Reuters RCV1 dataset, both across three languages. Our results demonstrated that LMs pre-trained with a language modeling objective, including multilingual variants, excelled at these tasks, while methods based on static word embeddings performed more poorly, particularly for non-English languages.

Building on this, Chapter 5 addressed the hierarchical nature of labels, which had been simplified to a flat set of labels in the previous study. This chapter introduced a novel approach using pre-trained Transformers and applied it to two noisy, domain-specific datasets containing customer support tickets and software bug reports, respectively. This research validated the use of LMs for an automated ticket routing mechanism to expedite support pipelines. We compared BERT LMs with various document summarization strategies and three novel global hierarchical classification methods that leveraged the label hierarchy. Our final approach achieved improved results over the existing baselines.

Part III focused on my contributions in XAI. This exploration began in Chapter 6

with an investigation into the value of simpler, interpretable models. We presented a case study on image-based fruit ripeness detection, demonstrating how a transparent system could be built by relying on expert knowledge and a transparent classification algorithm. A user study comparing our system with an end-to-end deep learning classifier using SHAP for post-hoc explanations revealed a tendency for users to favor simpler, less informative explanations, highlighting the importance of designing effective user interfaces for XAI. In addition to this, we outlined general guidelines for designing a classification system with explainability as a priority.

Acknowledging that opaque LMs are often indispensable for complex NLP tasks where domain knowledge is limited, Chapter 7 introduced a novel explainable-by-design framework. We observed that many post-hoc methods suffer from an "explanation divide", as they explain a system's decisions in terms of low-level features that hold little meaning for some end-users. Our design addressed this by using deep learning models to extract high-level, semantically meaningful features, which were then fed into a more transparent final classifier. This allowed our system to produce more faithful explanations grounded in unambiguously interpretable concepts. While this research is promising, we recognized that identifying intuitive features might not be doable for every problem. Nonetheless, we argue that this emphasis on problem decomposition into semantic attributes of the data is a necessary compromise for designers aiming to build transparent systems.

Motivated by the impressive performance of modern generative LLMs on mathematical and reasoning tasks, and their potential as semantic feature extractors, Chapter 8 addressed the limits of these models in understanding nuanced language, specifically on the task of paronomasia (pun) recognition. We explored this by measuring their abilities to classify and explain their decisions, using self-generated rationales to probe their reasoning in several challenging benchmarks. Our findings revealed that even state-of-the-art LLMs have a fragile grasp of the underlying linguistic mechanisms of puns. Additionally, their explanations often fail to correctly identify the supporting context and phonetic similarities. However, we found that prompting for a rationale usually improved performance, suggesting that this encourages a more structured reasoning process.

The next subsections will present some considerations on the use of LLMs in research and conclude with open issues that deserve further attention.

9.1 On the Challenges of LLMs in Research

Except for the final chapter, the majority of my research has focused on discriminative models applied to TC tasks. This specific focus explains why generative LLMs were not the primary subject of my work, despite their recent emergence as "general problem solvers", as discussed in Chapter 2. As this paradigm is increasingly adopted for a wide variety of discriminative tasks, this section examines some of the challenges of applying generative Transformer architectures in research

settings, challenges that were less prominent in earlier paradigms, such as transfer learning via fine-tuning. As previously noted, the fine-tuning approach adapts general linguistic knowledge acquired during pre-training to specific domains through parameter updates. In contrast, the generalized solver approach, which leverages *in-context learning*, can circumvent parameter updates entirely, relying instead on the model's ability to leverage its massive internal knowledge and symbolic reasoning via prompting.

Tackling discriminative tasks with this generative paradigm introduces specific practical challenges. First, as performance scales with model size, the sheer parameter count of these models creates a substantial barrier to entry, particularly for institutions lacking the high-performance computing infrastructure required for deploying them. While commercial providers offer API-based solutions, this introduces operational costs and necessitates data transmission to third parties. In research scenarios involving sensitive, proprietary, or personally identifiable information, these privacy concerns often render API-based solutions unviable; indeed, in some of our collaborations, this factor alone motivated the use of locally deployable, smaller-scale models. A related concern is the high environmental cost associated with training and querying large-scale models, which might conflict with the ethical goal of developing environmentally sustainable AI technologies.

Beyond these practical constraints, the reliance on closed-source or massive-scale LLMs raises methodological questions regarding the validity of scientific results. A primary challenge is *data contamination* (or test-set leakage). Since these models are trained on vast, undocumented snapshots of the Web, ensuring that a model has not previously encountered specific test samples is increasingly difficult. This ambiguity casts doubt on whether high performance stems from genuine reasoning and generalization or merely from approximate retrieval and memorization. We touched upon this issue in Chapter 8, where the model's performance on established public datasets was notably higher than on our newly collected examples, suggesting that the model's internal knowledge base may have implicitly leaked the test data. Given that commercial models are regularly updated with new content, maintaining a clean, controlled evaluation environment is hard and further complicates rigorous analysis.

Moreover, unlike the fine-tuning paradigm, where dataset quality is the primary driver of performance, generative AI via prompting is notably sensitive to the framing and formatting of the problem, as well as to inference hyperparameters such as temperature and decoding strategies [460]. While some researchers propose prompt engineering frameworks and meta-optimization as solutions, current practices often rely on trial-and-error methods to select the best-performing prompts. However, the combinatorial explosion of possible prompt variations renders exhaustive searches impractical, suggesting a need for more disciplined procedures to create effective prompts for robust results. A related issue with API-based access and closed-source LLMs is the difficulty of *estimating* a model's *confidence* in its answers, since most APIs return only the generated text and not the underlying raw predictions (logits)

[460]. Consequently, confidence is often inferred either by explicitly asking the model to provide a probability score (which can result in hallucinated values) or by measuring variability across multiple runs of the same query. Although some providers now allow retrieval of token logits, this capability is not universal, typically incurs additional costs, and can significantly slow down inference.

A further complication is *sycophancy*, a tendency for models to align with the user's implied view and provide agreeable answers rather than truthfully evaluating content [472, 464]. This behavior is reportedly induced by extensive post-training routines for user alignment, which, while more reasonable in conversational contexts, are counterproductive for formal discriminative tasks like classification. Unsurprisingly, when paired with the high prompt sensitivity, carefully constructed adversarial prompts or slight variations in instructions can condition the model's operative setting, leading to contradictory results. This effect complicates the reproducibility and generalization of scientific claims.

9.2 Future Research Directions

While each chapter concluded with a discussion of its specific limitations and opportunities, this final section outlines some broader directions we believe are important for future research.

First, a challenge lies in the sustainability of continued development and evaluation of LLMs. As the current trend in scale is unsustainable [288], future work should prioritize strategies for creating smaller but smarter models. Besides the obvious environmental benefits, this would democratize research, enabling more institutions to contribute to the development of fairer, more transparent open-source alternatives. Alongside sustainability, a deeper focus on evaluation is also important [460]. As our research on puns demonstrated, even state-of-the-art models that show impressive performance on some tasks, like coding or math, can rely on superficial heuristics, proving they are still far from being well-rounded reasoners.

Despite their limitations, the capabilities of these models are remarkable, and this also presents significant opportunities for XAI. The layered framework from Chapter 7, for instance, depends on high-quality semantic features, and a promising direction is to explore the use of modern LLMs to extract them. Relatedly, we deem the possibilities offered by Retrieval-Augmented Generation (RAG) systems to be particularly interesting. By connecting LLMs to external data sources like knowledge graphs, RAG is being widely studied for its ability to decouple a model's knowledge from its internal parameters. An interesting property is that, since knowledge graphs are formalized and relatively intuitive data structures, providing explanations in terms of paths on these graphs can yield faithful and expressive results, as the "vocabulary" of the explanation is a human-understandable data scheme. This could provide a clear, verifiable reasoning path for complex decisions, as recent work has started to explore [431, 465, 427].

We also support the argument of some authors [479, 414] that there is a pressing need for more rigorous, comparative evaluations of XAI methods, particularly concerning their faithfulness. As emphasized throughout my research, XAI techniques are not one-size-fits-all solutions; each has its strengths and weaknesses, and measures slightly different things. Clarifying how each method can help diagnose problems is crucial for making good use of this variety of choices to answer the right questions. Additionally, since the ultimate goal of XAI is to be useful to a system's end-users, we stress that a significant part of the field should be concerned with human-centered design and evaluation, intersecting with fields like Human-Computer Interaction (HCI) and the social sciences [129]. We do not argue that all research on interpretability must involve end-users. However, purely computational methods like LIME, SHAP, or Grad-CAM, while providing valuable theoretical approaches, often produce explanations that are only truly useful to technical experts. They may prove inadequate for users with different backgrounds, whose specific, high-level questions and mental models must be explicitly integrated into the design process. We note that here, too, LLMs may offer an additional opportunity in designing the user interface itself. Their powerful conversational technologies could enable interfaces where a user can engage in a dialogue with a system, ask follow-up questions, and receive adaptive explanations, a goal long advocated in both XAI and HCI literature [366, 393, 190]. However, for such a dialogue to be trustworthy, these models must be grounded so that they answer based on factual information — a capability that is still an active and challenging area of research [416, 276].

We believe that developing more inherently interpretable deep learning architectures can also contribute to the goals of XAI. While no neural network is likely to be completely transparent, some architectures are more aligned with human reasoning. Prototypical Networks, for example, which learn by comparing inputs to learned archetypes, offer an intuitive mechanism that aligns well with concepts like similarity and counterfactual explanations [208, 332]. Exploring how such human-centric constraints can be built into a model's design can help to quantify its limits and to provide more intuitive explanations. Likely, combining these complementary directions (more transparent models, expressive features, and more human-centered interfaces) will truly benefit the field.

Finally, we mention a challenge that will likely impact LLMs' abilities in the not-too-distant future. Their generative abilities have popularized the creation of synthetic datasets as a cost-effective alternative to manual annotation. While this opens new possibilities for data-scarce domains, it carries the risk of reinforcing existing biases when models are trained on increasing amounts of synthetically generated data. Synthetic data, regardless of quality, may not perfectly mirror the distribution of real data [480]. This mismatch, known as distribution shift, can result in models that overfit to synthetic patterns and progressively drift away from reality, leading to biased and discriminatory outcomes — a scenario termed "model collapse" [433, 424]. As a possible mitigation strategy, Shumailov et al. [433] suggests

ensuring original datasets containing human-created content remain available and separate from synthetic ones.

We offer these reflections to the community, hoping they may benefit both practitioners and researchers in the challenges that lie ahead.

Bibliography

[1] Andrea Gasparetto, Matteo Marcuzzo, Alessandro Zangari and Andrea Albarelli. 'A Survey on Text Classification Algorithms: From Text to Predictions'. In: *Information* 13.2 (Feb. 2022). Publisher: Multidisciplinary Digital Publishing Institute, p. 83. ISSN: 2078-2489. DOI: 10.3390/info13020083.

[2] Andrea Gasparetto, Alessandro Zangari, Matteo Marcuzzo and Andrea Albarelli. 'A survey on text classification: Practical perspectives on the Italian language'. In: *PLOS ONE* 17.7 (July 2022). Publisher: Public Library of Science, pp. 1–46. DOI: 10.1371/journal.pone.0270904.

[3] Matteo Marcuzzo, Alessandro Zangari, Andrea Albarelli and Andrea Gasparetto. 'Recommendation Systems: An Insight Into Current Development and Future Research Challenges'. In: *IEEE Access* 10 (July 2022), pp. 86578–86623. DOI: 10.1109/ACCESS.2022.3194536.

[4] Matteo Marcuzzo, Alessandro Zangari, Michele Schiavinato, Lorenzo Giudice, Andrea Gasparetto and Andrea Albarelli. 'A multi-level approach for hierarchical Ticket Classification'. In: *Proceedings of the Eighth Workshop on Noisy User-generated Text (W-NUT 2022)*. Gyeongju, Republic of Korea: Association for Computational Linguistics, Oct. 2022, pp. 201–214. URL: https://aclanthology.org/2022.wnut-1.22.

[5] Matteo Rizzo, Matteo Marcuzzo, Alessandro Zangari, Andrea Gasparetto and Andrea Albarelli. 'Fruit ripeness classification: A survey'. In: *Artificial Intelligence in Agriculture* 7 (Mar. 2023), pp. 44–57. ISSN: 2589-7217. DOI: 10.1016/j.aiia.2023.02.004.

[6] Alessandro Zangari, Matteo Marcuzzo, Michele Schiavinato, Andrea Gasparetto and Andrea Albarelli. 'Ticket automation: An insight into current research with applications to multi-level classification scenarios'. In: *Expert Systems with Applications* 225 (Sept. 2023), p. 119984. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2023.119984.

[7] Alessandro Zangari, Matteo Marcuzzo, Matteo Rizzo, Lorenzo Giudice, Andrea Albarelli and Andrea Gasparetto. 'Hierarchical Text Classification and Its Foundations: A Review of Current Research'. In: *Electronics* 13.7 (Jan. 2024). Publisher: Multidisciplinary Digital Publishing Institute, p. 1199. ISSN: 2079-9292. DOI: 10.3390/electronics13071199.

[8] Matteo Rizzo, Matteo Marcuzzo, Alessandro Zangari, Michele Schiavinato, Andrea Albarelli and Andrea Gasparetto. 'Stop Overkilling Simple Tasks with Black-Box Models, Use More Transparent Models Instead'. In: *Pattern Recognition and Artificial Intelligence*. Ed. by Christian Wallraven, Cheng-Lin Liu and Arun Ross. Singapore: Springer Nature, Feb. 2025, pp. 279–293. ISBN: 978-981-97-8702-9. DOI: 10.1007/978-981-97-8702-9_19.

[9] Alessandro Zangari, Matteo Marcuzzo, Matteo Rizzo, Andrea Albarelli and Andrea Gasparetto. 'Crossing the Divide: Designing Layers of Explainability'. In: *Artificial Intelligence and Soft Computing*. Ed. by Leszek Rutkowski, Rafał Scherer, Marcin Korytkowski, Witold Pedrycz, Ryszard Tadeusiewicz and Jacek M. Zurada. Cham: Springer Nature Switzerland, Feb. 2025, pp. 253–265. ISBN: 978-3-031-84353-2. DOI: 10.1007/978-3-031-84353-2_22.

[10] Matteo Marcuzzo, Alessandro Zangari, Andrea Albarelli, Jose Camacho-Collados and Mohammad Taher Pilehvar. 'Morables: A Benchmark for Assessing Abstract Moral Reasoning in LLMs with Fables'. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2025. Ed. by Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose and Violet Peng. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 27715–27739. ISBN: 979-8-89176-332-6. DOI: 10.18653/v1/2025.emnlp-main.1411.

[11] Alessandro Zangari, Matteo Marcuzzo, Andrea Albarelli, Mohammad Taher Pilehvar and Jose Camacho-Collados. 'Pun Unintended: LLMs and the Illusion of Humor Understanding'. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2025. Ed. by Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose and Violet Peng. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 27924–27959. ISBN: 979-8-89176-332-6. DOI: 10.18653/v1/2025.emnlp-main.1419.

[12] James Brown. 'Eight Types of Puns'. In: *PMLA* 71.1 (1956). Publisher: Modern Language Association, pp. 14–26. ISSN: 00308129. URL: http://www.jstor.org/stable/460188 (visited on 28/03/2025).

[13] Karen Spärck Jones. 'A statistical interpretation of term specificity and its application in retrieval'. In: *Journal of Documentation* 28.1 (1st Jan. 1972). Publisher: MCB UP Ltd, pp. 11–21. ISSN: 0022-0418. DOI: 10.1108/eb026526.

[14] A. V. Aho, J. E. Hopcroft and J. D. Ullman. 'On Finding Lowest Common Ancestors in Trees'. In: *SIAM Journal on Computing* 5.1 (1976), pp. 115–132. DOI: 10.1137/0205011.

[15] Ranulph Glanville. 'Inside every white box there are two black boxes trying to get out'. In: *Behavioral Science* 27.1 (1982), pp. 1–11. DOI: 10.1002/bs.3830270102.

[16] Leo Breiman. *Classification and Regression Trees*. New York: Routledge, 1984. ISBN: 978-1-315-13947-0. DOI: 10.1201/9781315139470.

[17] D. E. Rumelhart, G. E. Hinton and R. J. Williams. 'Learning Internal Representations by Error Propagation'. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*. Section: 11. Cambridge, MA, USA: MIT Press, Jan. 1986, pp. 318–362. ISBN: 0-262-68053-X.

[18] David Lewis. *Reuters-21578 Text Categorization Collection*. 1987. DOI: 10.24432/C52G6M.

[19] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel. 'Backpropagation Applied to Handwritten Zip Code Recognition'. In: *Neural Computation* 1.4 (1989), pp. 541–551. DOI: 10.1162/neco.1989.1.4.541.

[20] Mark A. Kramer. 'Nonlinear principal component analysis using autoassociative neural networks'. In: *AIChE Journal* 37.2 (1991), pp. 233–243. DOI: 10.1002/aic.690370209.

[21] Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik. 'A Training Algorithm for Optimal Margin Classifiers'. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. COLT '92. event-place: Pittsburgh, Pennsylvania, USA. New York, NY, USA: Association for Computing Machinery, 27th July 1992, pp. 144–152. ISBN: 0-89791-497-X. DOI: 10.1145/130385.130401.

[22] Jonathan J. Webster and Chunyu Kit. 'Tokenization as the Initial Phase in NLP'. In: *Proceedings of the 14th Conference on Computational Linguistics - Volume 4*. COLING '92. Nantes, France: Association for Computational Linguistics, 23rd Aug. 1992, pp. 1106–1110. DOI: 10.3115/992424.992434.

[23] Ken Lang. 'NewsWeeder: Learning to Filter Netnews'. In: *Machine Learning Proceedings 1995*. Ed. by Armand Prieditis and Stuart Russell. San Francisco (CA): Morgan Kaufmann, 1st Jan. 1995, pp. 331–339. ISBN: 978-1-55860-377-6. DOI: 10.1016/B978-1-55860-377-6.50048-7.

[24] Tin Kam Ho. 'Random decision forests'. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. Vol. 1. Montreal, QC, Canada, 14th Aug. 1995, pp. 278–282. DOI: 10.1109/ICDAR.1995.598994.

[25] Sepp Hochreiter and Jürgen Schmidhuber. 'Long Short-Term Memory'. In: *Neural Computation* 9.8 (15th Nov. 1997), pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

[26] Daphne Koller and Mehran Sahami. 'Hierarchically Classifying Documents Using Very Few Words'. In: *Proceedings of the Fourteenth International Conference on Machine Learning*. ICML '97. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 170–178. ISBN: 1-55860-486-3.

[27] Mitchell P Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz and Ann Taylor. 'Treebank-3'. In: *Linguistic Data Consortium, Philadelphia* 14 (1999).

[28] Cristina Bosco, Vincenzo Lombardo, Daniela Vassallo and Leonardo Lesmo. 'Building a Treebank for Italian: a Data-driven Annotation Schema'. In: *Proceedings of the Second International Conference on Language Resources and Evaluation (LREC'00)*. Athens, Greece: European Language Resources Association (ELRA), May 2000.

[29] Aixin Sun and Ee-Peng Lim. 'Hierarchical Text Classification and Evaluation'. In: *Proceedings of the 2001 IEEE International Conference on Data Mining*. ICDM '01. USA: IEEE Computer Society, 2001, pp. 521–528. ISBN: 0-7695-1119-8.

[30] Fabrizio Sebastiani. 'Machine Learning in Automated Text Categorization'. In: *ACM Comput. Surv.* 34.1 (Mar. 2002). Place: New York, NY, USA Publisher: Association for Computing Machinery, pp. 1–47. ISSN: 0360-0300. DOI: 10.1145/505282.505283.

[31] Aixin Sun, Ee-Peng Lim and Wee-Keong Ng. 'Hierarchical Text Classification Methods and Their Specification'. In: *Cooperative Internet Computing*. Section: 14. Boston, MA: Springer US, 2003, pp. 236–256. ISBN: 978-1-4615-0435-1. DOI: 10.1007/978-1-4615-0435-1_14.

[32] Erik F. Tjong Kim Sang and Fien De Meulder. 'Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition'. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*. 2003, pp. 142–147.

[33] David D. Lewis, Yiming Yang, Tony G. Rose and Fan Li. 'RCV1: A New Benchmark Collection for Text Categorization Research'. In: *J. Mach. Learn. Res.* 5 (Dec. 2004). Publisher: JMLR.org, pp. 361–397. ISSN: 1532-4435.

[34] Halvor Bøyesen Eifring and Rolf Theil. *Linguistics for Students of Asian and African Languages*. Section: 4. Institutt for østeuropeiske og orientalske studier, 2004. 187 pp.

[35] Bryan Klimt and Yiming Yang. 'The Enron Corpus: A New Dataset for Email Classification Research'. In: *Proceedings of the 15th European Conference on Machine Learning*. ECML'04. event-place: Pisa, Italy. Berlin, Heidelberg: Springer-Verlag, 2004, pp. 217–226. ISBN: 3-540-23105-6. DOI: 10.1007/978-3-540-30115-8_22.

[36] A. Sun, E.-P. Lim, W.-K. Ng and J. Srivastava. 'Blocking reduction strategies in hierarchical text classification'. In: *IEEE Transactions on Knowledge and Data Engineering* 16.10 (2004), pp. 1305–1308. DOI: 10.1109/TKDE.2004.50.

[37] Lionel Clément and Éric Villemonte de La Clergerie. 'MAF: a Morphosyntactic Annotation Framework'. In: *2nd Language & Technology Conference (LTC'05)*. Ed. by Zygmunt Vetulani. 2nd Language & Technology Conference (LTC'05). Poznan, Poland, 2005, pp. 90–94. URL: https://hal.science/hal-01104466 (visited on 01/07/2025).

[38] Gianna M. Del Corso, Antonio Gullí and Francesco Romani. 'Ranking a stream of news'. In: *Proceedings of the 14th International Conference on World Wide Web*. WWW '05. event-place: Chiba, Japan. New York, NY, USA: Association for Computing Machinery, 2005, pp. 97–106. ISBN: 1-59593-046-9. DOI: 10.1145/1060745.1060764.

[39] Steven Bird. 'NLTK: The Natural Language Toolkit'. In: *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*. Sydney, Australia: Association for Computational Linguistics, July 2006, pp. 69–72. DOI: 10.3115/1225403.1225421.

[40] Svetlana Kiritchenko, Stan Matwin, Richard Nock and A. Fazel Famili. 'Learning and Evaluation in the Presence of Class Hierarchies: Application to Text Categorization'. In: *Advances in Artificial Intelligence*. Ed. by Luc Lamontagne and Mario Marchand. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 395–406. ISBN: 978-3-540-34630-2.

[41] F. Mendoza and J.M. Aguilera. 'Application of Image Analysis for Classification of Ripening Bananas'. In: *Journal of Food Science* 69.9 (2006), E471–E477. DOI: 10.1111/j.1365-2621.2004.tb09932.x.

[42] Alex Freitas and Andre de Carvalho. 'A Tutorial on Hierarchical Classification with Applications in Bioinformatics'. In: *Research and Trends in Data Mining Technologies and Applications* (Jan. 2007). ISBN: 9781599042732. DOI: 10.4018/978-1-59904-271-8.ch007.

[43] Michelangelo Ceci and Donato Malerba. 'Classifying web documents in a hierarchy of categories: a comprehensive study'. In: *Journal of Intelligent Information Systems* 28.1 (1st Feb. 2007), pp. 37–78. ISSN: 1573-7675. DOI: 10.1007/s10844-006-0003-2.

[44] Vikramjit Mitra, Chia-Jiu Wang and Satarupa Banerjee. 'Text classification: A least square support vector machine approach'. In: *Applied Soft Computing* 7.3 (June 2007), pp. 908–914. ISSN: 1568-4946. DOI: 10.1016/j.asoc.2006.04.002.

[45] Bernardo Magnini et al. 'Evaluation of Natural Language Tools for Italian: EVALITA 2007'. In: *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco: European Language Resources Association (ELRA), 28th May 2008. URL: http://www.lrec-conf.org/proceedings/lrec2008/pdf/630_paper.pdf.

[46] Eric Laporte, Takuya Nakamura and Stavroula Voyatzi. 'A French Corpus Annotated for Multiword Nouns'. In: *Language Resources and Evaluation Conference. Workshop Towards a Shared Task on Multiword Expressions*. Marrakech, Morocco, 1st June 2008, pp. 27–30.

[47] Laurens van der Maaten and Geoffrey Hinton. 'Visualizing Data using t-SNE'. In: *Journal of Machine Learning Research* 9.86 (2008), pp. 2579–2605. URL: http://jmlr.org/papers/v9/vandermaaten08a.html.

[48] Kunal Punera and Joydeep Ghosh. 'Enhanced Hierarchical Classification via Isotonic Smoothing'. In: *Proceedings of the 17th International Conference on World Wide Web*. WWW '08. event-place: Beijing, China. New York, NY, USA: Association for Computing Machinery, 2008, pp. 151–160. ISBN: 978-1-60558-085-2. DOI: 10.1145/1367497.1367518.

[49] Victor Raskin, ed. *The primer of humor research*. Berlin, New York: De Gruyter Mouton, 2008. ISBN: 978-3-11-019849-2. DOI: 10.1515/9783110198492.

[50] Qihong Shao, Yi Chen, Shu Tao, Xifeng Yan and Nikos Anerousis. 'Efficient Ticket Routing by Resolution Sequence Mining'. In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '08. event-place: Las Vegas, Nevada, USA. New York, NY, USA: Association for Computing Machinery, 2008, pp. 605–613. ISBN: 978-1-60558-193-4. DOI: 10.1145/1401890.1401964.

[51] Jingnian Chen, Houkuan Huang, Shengfeng Tian and Youli Qu. 'Feature selection for text classification with Naïve Bayes'. In: *Expert Systems with Applications* 36.3 (Apr. 2009), pp. 5432–5435. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2008.06.054.

[52] Kai Zhang, James T. Kwok and Bahram Parvin. 'Prototype vector machine for large scale semi-supervised learning'. In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09. New York, NY, USA: Association for Computing Machinery, 14th June 2009, pp. 1233–1240. ISBN: 978-1-60558-516-1. DOI: 10.1145/1553374.1553531.

[53] Thomas Beckers, Ingo Frommholz and Ralf Bönning. 'Multi-facet Classification of E-mails in a Helpdesk Scenario'. In: *University of Bedfordshire Repository* (Sept. 2009). URL: https://core.ac.uk/reader/29820850.

[54] Johan Bos, Fabio Massimo Zanzotto and Marco Pennacchiotti. 'Textual Entailment at EVALITA 2009'. In: (2009).

[55] Liviu P. Dinu and Andrei Rusu. 'Rank Distance Aggregation as a Fixed Classifier Combining Rule for Text Categorization'. In: *Proceedings of the 11th International Conference on Computational Linguistics and Intelligent Text Processing*. CICLing'10. Iaşi, Romania: Springer-Verlag, 21st Mar. 2010, pp. 638–647. ISBN: 3-642-12115-2. DOI: 10.1007/978-3-642-12116-6_54.

[56] Peter Prettenhofer and Benno Stein. *Webis Cross-Lingual Sentiment Dataset 2010 (Webis-CLS-10)*. 16th July 2010. URL: https://zenodo.org/records/3251672 (visited on 31/07/2025).

[57] Carlos N. Silla and Alex A. Freitas. 'A survey of hierarchical classification across different application domains'. In: *Data Mining and Knowledge Discovery* 22.1 (1st Jan. 2011), pp. 31–72. ISSN: 1573-756X. DOI: 10.1007/s10618-010-0175-9.

[58] Antonio Toral, Pavel Pecina, Marc Poch and Andy Way. 'Towards a User-Friendly Webservice Architecture for Statistical Machine Translation in the PANACEA project'. In: *Proceedings of the 15th Annual conference of the European Association for Machine Translation*. Leuven, Belgium: European Association for Machine Translation, 30th May 2011.

[59] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng and Christopher Potts. 'Learning Word Vectors for Sentiment Analysis'. In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150.

[60] Konstantinos Sechidis, Grigorios Tsoumakas and Ioannis Vlahavas. 'On the Stratification of Multi-label Data'. In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba and Michalis Vazirgiannis. Athens, Greece: Springer Berlin Heidelberg, 5th Sept. 2011, pp. 145–158. ISBN: 978-3-642-23808-6. DOI: 10.1007/978-3-642-23808-6_10.

[61] Emily M. Bender. 'On Achieving and Evaluating Language-Independence in NLP'. In: *Linguistic Issues in Language Technology* 6 (1st Oct. 2011). DOI: 10.33011/lilt.v6i.1239.

[62] Ricardo Cerri, Rodrigo C. Barros and André C. P. L. F. de Carvalho. 'Hierarchical multi-label classification for protein function prediction: A local approach based on neural networks'. In: *2011 11th International Conference on Intelligent Systems Design and Applications*. 2011, pp. 337–343. DOI: 10.1109/ISDA.2011.6121678.

[63] F. Pedregosa et al. 'Scikit-learn: Machine Learning in Python'. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[64] Mike Schuster and Kaisuke Nakajima. 'Japanese and Korean voice search'. In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Kyoto, Japan, 25th Mar. 2012, pp. 5149–5152. DOI: 10.1109/ICASSP.2012.6289079.

[65] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua and Sabine Süsstrunk. 'SLIC Superpixels Compared to State-of-the-Art Superpixel Methods'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.11 (2012), pp. 2274–2282. DOI: 10.1109/TPAMI.2012.120.

[66] Hari S. Gupta and Bikram Sengupta. 'Scheduling Service Tickets in Shared Delivery'. In: *Service-Oriented Computing*. Ed. by Chengfei Liu, Heiko Ludwig, Farouk Toumani and Qi Yu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 79–95. ISBN: 978-3-642-34321-6. DOI: 10.1007/978-3-642-34321-6_6.

[67] Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy and James R. Curran. 'Learning Multilingual Named Entity Recognition from Wikipedia'. In: *Artif. Intell.* 194 (Jan. 2013). Publisher: Elsevier Science Publishers Ltd., pp. 151–175. ISSN: 0004-3702. DOI: 10.1016/j.artint.2012.03.006.

[68] Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean. 'Efficient Estimation of Word Representations in Vector Space'. In: *1st International Conference on Learning Representations, ICLR 2013*. Vol. abs/1301.3781. eprint: 1301.3781. May 2013, pp. 1–12. DOI: 10.48550/arXiv.1301.3781.

[69] Razvan Pascanu, Tomas Mikolov and Yoshua Bengio. 'On the Difficulty of Training Recurrent Neural Networks'. In: *Proceedings of the 30th International Conference on Machine Learning - Volume 28*. ICML'13. Atlanta, Georgia, USA: JMLR.org, 16th June 2013, pp. III–1310–III–1318.

[70] Zobia Rehman, Waqas Anwar, Usama Ijaz Bajwa, Wang Xuan and Zhou Chaoying. 'Morpheme Matching Based Text Tokenization for a Scarce Resourced Language'. In: *PLOS ONE* 8.8 (21st Aug. 2013). Publisher: Public Library of Science, pp. 1–8. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0068178.

[71] Roberto Basili, Diego De Cao, Alessandro Lenci, Alessandro Moschitti and Giulia Venturi. 'EvalIta 2011: The Frame Labeling over Italian Texts Task'. In: *Evaluation of Natural Language and Speech Tools for Italian*. Ed. by Bernardo Magnini, Francesco Cutugno, Mauro Falcone and Emanuele Pianta. Springer Berlin Heidelberg, 2013, pp. 195–204. ISBN: 978-3-642-35828-9.

[72] Verena Lyding et al. *PAISÀ Corpus of Italian Web Text*. 2013. URL: http://hdl.handle.net/20.500.12124/3.

[73] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado and Jeffrey Dean. 'Distributed Representations of Words and Phrases and Their Compositionality'. In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'13. Curran Associates Inc., 2013, pp. 3111–3119.

[74] Saif M. Mohammad and Peter D. Turney. 'Crowdsourcing a word-emotion association lexicon'. In: *Computational Intelligence* 29.3 (2013), pp. 436–465.

[75] David I Shuman, Sunil K. Narang, Pascal Frossard, Antonio Ortega and Pierre Vandergheynst. 'The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains'. In: *IEEE Signal Processing Magazine* 30.3 (2013), pp. 83–98. DOI: 10.1109/MSP.2012.2235192.

[76] Verena Lyding et al. 'The PAISÀ Corpus of Italian Web Texts'. In: *Proceedings of the 9th Web as Corpus Workshop (WaC-9)*. Gothenburg, Sweden: Association for Computational Linguistics, Apr. 2014, pp. 36–43. DOI: 10.3115/v1/W14-0406.

[77] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk and Yoshua Bengio. 'Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation'. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 25th Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179.

[78] Yoon Kim. 'Convolutional Neural Networks for Sentence Classification'. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 25th Oct. 2014, pp. 1746–1751. DOI: 10.3115/v1/D14-1181.

[79] Jeffrey Pennington, Richard Socher and Christopher Manning. 'GloVe: Global Vectors for Word Representation'. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 25th Oct. 2014, pp. 1532–1543. DOI: 10.3115/v1/D14-1162.

[80] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau and Yoshua Bengio. 'On the Properties of Neural Machine Translation: Encoder–Decoder Approaches'. In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 103–111. DOI: 10.3115/v1/W14-4012.

[81] Vladimír Benko. 'Aranea: Yet Another Family of (Comparable) Web Corpora'. In: *Text, Speech and Dialogue*. Ed. by Petr Sojka, Aleš Horák, Ivan Kopeček and Karel Pala. Springer, Cham: Springer International Publishing, 2014, pp. 247–256. ISBN: 978-3-319-10816-2. DOI: 10.1007/978-3-319-10816-2_31.

[82] KB Lab. *KB Europeana Newspapers NER Dataset*. 2014. URL: https://lab.kb.nl/dataset/europeana-newspapers-ner (visited on 13/08/2025).

[83] Min-Ling Zhang and Zhi-Hua Zhou. 'A Review on Multi-Label Learning Algorithms'. In: *IEEE Transactions on Knowledge and Data Engineering* 26.8 (2014), pp. 1819–1837. DOI: 10.1109/TKDE.2013.39.

[84] Aris Kosmopoulos, Ioannis Partalas, Eric Gaussier, Georgios Paliouras and Ion Androutsopoulos. 'Evaluation measures for hierarchical classification: a unified view and novel approaches'. In: *Data Mining and Knowledge Discovery* 29.3 (1st May 2015), pp. 820–865. ISSN: 1573-756X. DOI: 10.1007/s10618-014-0382-x.

[85] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller and Wojciech Samek. 'On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation'. In: *PLOS ONE* 10.7 (10th July 2015). Publisher: Public Library of Science, e0130140. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0130140.

[86] Tristan Miller and Iryna Gurevych. 'Automatic disambiguation of English puns'. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. ACL-IJCNLP 2015. Ed. by Chengqing Zong and Michael Strube. Beijing, China: Association for Computational Linguistics, July 2015, pp. 719–729. DOI: 10.3115/v1/P15-1070.

[87] Kranti Vithal Ghag and Ketan Shah. 'Comparative analysis of effect of stopwords removal on sentiment classification'. In: *2015 International Conference on Computer, Communication and Control (IC4)*. Indore, India, 10th Sept. 2015, pp. 1–6. DOI: 10.1109/IC4.2015.7375527.

[88] James W. Pennebaker, Ryan L. Boyd, Kayla Jordan and Kate Blackburn. 'The Development and Psychometric Properties of LIWC2015'. In: (15th Sept. 2015). URL: http://hdl.handle.net/2152/31333 (visited on 20/09/2025).

[89] Thang Luong, Hieu Pham and Christopher D. Manning. 'Effective Approaches to Attention-based Neural Machine Translation'. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sept. 2015, pp. 1412–1421. DOI: 10.18653/v1/D15-1166.

[90] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba and S. Fidler. 'Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books'. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. ISSN: 2380-7504. Santiago, Chile: IEEE Computer Society, 11th Dec. 2015, pp. 19–27. DOI: 10.1109/ICCV.2015.11.

[91] Xiang Zhang, Junbo Zhao and Yann LeCun. 'Character-Level Convolutional Networks for Text Classification'. In: *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'15. Cambridge, MA, USA: MIT Press, Dec. 2015, pp. 649–657.

[92] Dzmitry Bahdanau, Kyunghyun Cho and Yoshua Bengio. 'Neural Machine Translation by Jointly Learning to Align and Translate'. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015.

[93] François Chollet et al. *Keras*. 2015. URL: https://keras.io.

[94] Tristan Miller and Mladen Turković. 'Towards the automatic detection and identification of English puns'. In: *The European Journal of Humour Research* 4.1 (Jan. 2016), pp. 59–75. DOI: 10.7592/EJHR2016.4.1.miller.

[95] Ethan Fast, Binbin Chen and Michael S. Bernstein. 'Empath: Understanding Topic Signals in Large-Scale Text'. In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. New York, NY, USA: Association for Computing Machinery, 7th May 2016, pp. 4647–4657. ISBN: 978-1-4503-3362-7. DOI: 10.1145/2858036.2858535.

[96] Anne-Lyse Minard, Manuela Speranza, Ruben Urizar, Begoña Altuna, Marieke van Erp, Anneleen Schoen and Chantal van Son. 'MEANTIME, the NewsReader Multilingual Event and Time Corpus'. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), May 2016, pp. 4417–4422.

[97] Joakim Nivre et al. 'Universal Dependencies v1: A Multilingual Treebank Collection'. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), May 2016, pp. 1659–1666.

[98] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola and Eduard Hovy. 'Hierarchical Attention Networks for Document Classification'. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California, USA: Association for Computational Linguistics, 12th June 2016, pp. 1480–1489. DOI: 10.18653/v1/N16-1174.

[99] Maria Pontiki et al. 'SemEval-2016 Task 5: Aspect Based Sentiment Analysis'. In: *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. San Diego, California: Association for Computational Linguistics, June 2016, pp. 19–30. DOI: 10.18653/v1/S16-1002.

[100] Marco Ribeiro, Sameer Singh and Carlos Guestrin. '"Why Should I Trust You?": Explaining the Predictions of Any Classifier'. In: *Proceedings of the 2016 Conference of the North American Chapter of the ACL: Demonstrations*. San Diego, California: Association for Computational Linguistics, June 2016, pp. 97–101. DOI: 10.18653/v1/N16-3020.

[101] Rico Sennrich, Barry Haddow and Alexandra Birch. 'Neural Machine Translation of Rare Words with Subword Units'. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, 7th Aug. 2016, pp. 1715–1725. DOI: 10.18653/v1/P16-1162.

[102] Himabindu Lakkaraju, Stephen H. Bach and Jure Leskovec. 'Interpretable Decision Sets: A Joint Framework for Description and Prediction'. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. New York, NY, USA: Association for Computing Machinery, 13th Aug. 2016, pp. 1675–1684. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939874.

[103] Wei-Hao Peng, Ming-Yang Lee, Tsung-Han Li, Chi-Hung Huang and Po-Chiang Lin. 'Performance comparison of image keypoint detection, description, and matching methods'. In: *2016 IEEE 5th Global Conference on Consumer Electronics*. 2016 IEEE 5th Global Conference on Consumer Electronics. Oct. 2016, pp. 1–2. DOI: 10.1109/GCCE.2016.7800416.

[104] Tao Lei, Regina Barzilay and Tommi Jaakkola. 'Rationalizing Neural Predictions'. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2016. Ed. by Jian Su, Kevin Duh and Xavier Carreras. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 107–117. DOI: 10.18653/v1/D16-1011.

[105] Francesco Barbieri, Valerio Basile, Danilo Croce, Malvina Nissim, Nicole Novielli and Viviana Patti. 'Overview of the Evalita 2016 SENTIment POLarity Classification Task'. In: *Proceedings of Third Italian Conference on Computational Linguistics (CLiC-it 2016) & Fifth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian (EVALITA 2016)*. Vol. 1749. CEUR Workshop Proceedings. Naples, Italy, 5th Dec. 2016, pp. 146–155.

[106] Anne-Lyse Minard, Manuela Sperenza and Tommaso Caselli. 'The EVALITA 2016 Event Factuality Annotation Task (FactA)'. In: *Proceedings of Third Italian Conference on Computational Linguistics (CLiC-it 2016) & Fifth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*. Naples, Italy, 5th Dec. 2016.

[107] Jenna Burrell. 'How the machine 'thinks': Understanding opacity in machine learning algorithms'. In: *Big Data & Society* 3.1 (2016). DOI: 10.1177/2053951715622512.

[108] Mengchen Liu, Jiaxin Shi, Zhen Li, Chongxuan Li, Jun Zhu and Shixia Liu. 'Towards Better Analysis of Deep Convolutional Neural Networks'. In: *IEEE Transactions on Visualization and Computer Graphics* 23.1 (Jan. 2017), pp. 91–100. ISSN: 1941-0506. DOI: 10.1109/TVCG.2016.2598831.

[109] Salvatore Attardo, ed. *The Routledge Handbook of Language and Humor*. New York: Routledge, Feb. 2017. ISBN: 978-1-315-73116-2. DOI: 10.4324/9781315731162.

[110] Armand Joulin, Edouard Grave, Piotr Bojanowski and Tomas Mikolov. 'Bag of Tricks for Efficient Text Classification'. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Valencia, Spain: Association for Computational Linguistics, 3rd Apr. 2017, pp. 427–431.

[111] Nemanja Spasojevic, Preeti Bhargava and Guoning Hu. 'DAWT: Densely Annotated Wikipedia Texts Across Multiple Languages'. In: *Proceedings of the 26th International Conference on World Wide Web Companion*. WWW '17 Companion. Perth, Australia: International World Wide Web Conferences Steering Committee, 3rd Apr. 2017, pp. 1655–1662. ISBN: 978-1-4503-4914-7. DOI: 10.1145/3041021.3053367.

[112] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek and Klaus-Robert Müller. 'Explaining nonlinear classification decisions with deep Taylor decomposition'. In: *Pattern Recognition* 65 (1st May 2017), pp. 211–222. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2016.11.008.

[113] Shuo Xu, Yan Li and Zheng Wang. 'Bayesian Multinomial Naïve Bayes Classifier to Text Classification'. In: *Advanced Multimedia and Ubiquitous Engineering. FutureTech MUE 2017. Lecture Notes in Electrical Engineering*. Vol. LNEE 448. Seoul, Korea: Springer Singapore, 22nd May 2017, pp. 347–352. ISBN: 978-981-10-5041-1. DOI: 10.1007/978-981-10-5041-1_57.

[114] Piotr Bojanowski, Edouard Grave, Armand Joulin and Tomas Mikolov. 'Enriching Word Vectors with Subword Information'. In: *Transactions of the Association for Computational Linguistics* 5 (1st June 2017), pp. 135–146. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00051.

[115] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas and Martin Wattenberg. *SmoothGrad: removing noise by adding noise*. 12th June 2017. DOI: 10.48550/arXiv.1706.03825.

[116] Mirko Polato. *Dataset belonging to the help desk log of an Italian Company*. July 2017. DOI: 10.4121/uuid:0c60edf1-6f83-4e75-9367-4c63b3e9d5bb.

[117] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals and George E. Dahl. 'Neural Message Passing for Quantum Chemistry'. In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 6th Aug. 2017, pp. 1263–1272. URL: https://proceedings.mlr.press/v70/gilmer17a.html.

[118] Mukund Sundararajan, Ankur Taly and Qiqi Yan. 'Axiomatic Attribution for Deep Networks'. In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, 6th Aug. 2017, pp. 3319–3328. URL: https://proceedings.mlr.press/v70/sundararajan17a.html.

[119] Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller and Wojciech Samek. '"What is relevant in a text document?": An interpretable machine learning approach'. In: *PLOS ONE* 12.8 (11th Aug. 2017). Publisher: Public Library of Science, e0181142. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0181142.

[120] Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight and Heng Ji. 'Cross-lingual Name Tagging and Linking for 282 Languages'. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, 30th Aug. 2017, pp. 1946–1958. DOI: 10.18653/v1/P17-1178.

[121] Jingzhou Liu, Wei-Cheng Chang, Yuexin Wu and Yiming Yang. 'Deep Learning for Extreme Multi-Label Text Classification'. In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '17. Shinjuku, Tokyo, Japan: Association for Computing Machinery, Aug. 2017, pp. 115–124. ISBN: 978-1-4503-5022-8. DOI: 10.1145/3077136.3080834.

[122] Tristan Miller, Christian Hempelmann and Iryna Gurevych. 'SemEval-2017 Task 7: Detection and Interpretation of English Puns'. In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Ed. by Steven Bethard, Marine Carpuat, Marianna Apidianaki, Saif M. Mohammad, Daniel Cer and David Jurgens. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 58–68. DOI: 10.18653/v1/S17-2005.

[123] Piotr Szymański and Tomasz Kajdanowicz. 'A Network Perspective on Stratification of Multi-Label Data'. In: *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications*. Ed. by Luís Torgo, Paula Branco and Nuno Moniz. Vol. 74. Proceedings of Machine Learning Research. Skopje, Macedonia: PMLR, 22nd Sept. 2017, pp. 22–35.

[124] Shuyi Lin, Wenxing Hong, Dingding Wang and Tao Li. 'A survey on expert finding techniques'. In: *Journal of Intelligent Information Systems* 49.2 (1st Oct. 2017), pp. 255–279. ISSN: 1573-7675. DOI: 10.1007/s10844-016-0440-5.

[125] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh and Dhruv Batra. 'Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization'. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. ICCV 2017. ISSN: 2380-7504. Oct. 2017, pp. 618–626. DOI: 10.1109/ICCV.2017.74.

[126] Ye Zhang and Byron Wallace. 'A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification'. In: *Proceedings of the 8th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, 17th Nov. 2017, pp. 253–263. URL: https://aclanthology.org/I17-1026.

[127] Derek Doran, Sarah Schulz and Tarek R. Besold. 'What Does Explainable AI Really Mean? A New Conceptualization of Perspectives'. In: *Proceedings of the First International Workshop on Comprehensibility and Explanation in AI and ML 2017*. Comprehensibility and Explanation in AI and ML 2017. Ed. by Tarek R. Besold and Oliver Kutz. Vol. 2071. CEUR Workshop Proceedings. ISSN: 1613-0073. Bari, Italy: CEUR, Nov. 2017. URL: https://ceur-ws.org/Vol-2071/#paper2 (visited on 12/09/2025).

[128] Ashish Vaswani et al. 'Attention is All You Need'. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc, 4th Dec. 2017, pp. 6000–6010. ISBN: 978-1-5108-6096-4.

[129] Tim Miller, Piers Howe and Liz Sonenberg. *Explainable AI: Beware of Inmates Running the Asylum Or: How I Learnt to Stop Worrying and Love the Social and Behavioural Sciences*. 5th Dec. 2017. DOI: 10.48550/arXiv.1712.00547.

[130] Daniel Beneker and Carsten Gips. 'Using Clustering for Categorization of Support Tickets'. In: *Lernen, Wissen, Daten, Analysen (LWDA) Conference Proceedings, Rostock, Germany, September 11-13, 2017*. Ed. by Michael Leyer. Vol. 1917. CEUR Workshop Proceedings. CEUR-WS.org, 2017, pp. 51–62. URL: http://ceur-ws.org/Vol-1917/paper10.pdf.

[131] Jianglei Han and Aixin Sun. 'Mean Average Distance to Resolver: An Evaluation Metric for Ticket Routing in Expert Network'. In: *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. 2017, pp. 594–602. DOI: 10.1109/ICSME.2017.18.

[132] Colin Lea, Michael D. Flynn, René Vidal, Austin Reiter and Gregory D. Hager. 'Temporal Convolutional Networks for Action Segmentation and Detection'. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1003–1012. DOI: 10.1109/CVPR.2017.113.

[133] Scott M Lundberg and Su-In Lee. 'A Unified Approach to Interpreting Model Predictions'. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. DOI: 10.48550/arXiv.1705.07874.

[134] Nishtha Madaan, Gautam Singh, Arun Kumar and Gargi B Dasgupta. 'Neev: A cognitive support agent for content improvement in hardware tickets'. In: *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. 2017, pp. 239–246. DOI: 10.23919/INM.2017.7987285.

[135] Wubai Zhou et al. 'STAR: A System for Ticket Analysis and Resolution'. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '17. event-place: Halifax, NS, Canada. New York, NY, USA: Association for Computing Machinery, 2017, pp. 2181–2190. ISBN: 978-1-4503-4887-4. DOI: 10.1145/3097983.3098190.

[136] Pierpaolo Basile, Danilo Croce, Valerio Basile and Marco Polignano. 'Overview of the EVALITA 2018 Aspect-based Sentiment Analysis task (ABSITA)'. In: Jan. 2018, pp. 10–16. ISBN: 978-88-31978-42-2. DOI: 10.4000/books.aaccademia.4451.

[137] Pierpaolo Basile and Nicole Novielli. 'Overview of the Evalita 2018 itaLIan Speech acT labEliNg (iLISTEN) Task'. In: *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*. Vol. 2263. CEUR Workshop Proceedings. Jan. 2018, pp. 44–50. ISBN: 978-88-31978-42-2. DOI: 10.4000/books.aaccademia.4482.

[138] Felice Dell'Orletta and Malvina Nissim. 'Overview of the EVALITA 2018 Cross-Genre Gender Prediction (GxG) Task'. In: *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian, 2018*. Vol. 2263. CEUR Workshop Proceedings. Jan. 2018, pp. 35–43. ISBN: 978-88-31978-42-2. DOI: 10.4000/books.aaccademia.4478.

[139] Jian Xu, Hang Zhang, Wubai Zhou, Rouying He and Tao Li. 'Signature based trouble ticket classification'. In: *Future Generation Computer Systems* 78 (Jan. 2018), pp. 41–58. ISSN: 0167-739X. DOI: 10.1016/j.future.2017.07.054.

[140] Grégoire Montavon, Wojciech Samek and Klaus-Robert Müller. 'Methods for interpreting and understanding deep neural networks'. In: *Digital Signal Processing* 73 (1st Feb. 2018), pp. 1–15. ISSN: 1051-2004. DOI: 10.1016/j.dsp.2017.10.011.

[141] Qimai Li, Zhichao Han and Xiao-Ming Wu. 'Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning'. In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI*

Symposium on Educational Advances in Artificial Intelligence. AAAI'18. Issue: 433. New Orleans, Louisiana, USA: AAAI Press, 2nd Feb. 2018, pp. 3538–3545. ISBN: 978-1-57735-800-8. DOI: 10.1609/aaai.v32i1.11604.

[142] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader and Vineeth N Balasubramanian. 'Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks'. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). Mar. 2018, pp. 839–847. DOI: 10.1109/WACV.2018.00097.

[143] Oscar Li, Hao Liu, Chaofan Chen and Cynthia Rudin. 'Deep Learning for Case-Based Reasoning Through Prototypes: A Neural Network That Explains Its Predictions'. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1 (29th Apr. 2018). ISSN: 2374-3468. DOI: 10.1609/aaai.v32i1.11771.

[144] Andrew Ross and Finale Doshi-Velez. 'Improving the Adversarial Robustness and Interpretability of Deep Neural Networks by Regularizing Their Input Gradients'. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1 (Apr. 2018). DOI: 10.1609/aaai.v32i1.11504.

[145] Manuela Sanguinetti, Fabio Poletto, Cristina Bosco, Viviana Patti and Marco Stranisci. 'An Italian Twitter Corpus of Hate Speech against Immigrants'. In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA), 7th May 2018.

[146] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee and Luke Zettlemoyer. 'Deep Contextualized Word Representations'. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 1st June 2018, pp. 2227–2237. DOI: 10.18653/v1/N18-1202.

[147] Adina Williams, Nikita Nangia and Samuel Bowman. 'A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference'. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 1st June 2018, pp. 1112–1122. DOI: 10.18653/v1/N18-1101.

[148] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell and Marcus Rohrbach. 'Multimodal Explanations: Justifying Decisions and Pointing to the Evidence'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. CVPR 2018. ISSN: 2575-7075. June 2018, pp. 8779–8788. DOI: 10.1109/CVPR.2018.00915.

[149] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov and Liang-Chieh Chen. 'MobileNetV2: Inverted Residuals and Linear Bottlenecks'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. CVPR 2018. ISSN: 2575-7075. June 2018, pp. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

[150] Wenjie Zhang, Junchi Yan, Xiangfeng Wang and Hongyuan Zha. 'Deep Extreme Multi-Label Learning'. In: *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*. ICMR '18. Yokohama, Japan: Association for Computing Machinery, June 2018, pp. 100–107. ISBN: 978-1-4503-5046-4. DOI: 10.1145/3206025.3206030.

[151] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas and Rory sayres. 'Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV)'. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, 10th July 2018, pp. 2668–2677. URL: https://proceedings.mlr.press/v80/kim18d.html.

[152] Jonatas Wehrmann, Ricardo Cerri and Rodrigo Barros. 'Hierarchical Multi-Label Classification Networks'. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, 10th July 2018, pp. 5075–5084. URL: https://proceedings.mlr.press/v80/wehrmann18a.html.

[153] Jeremy Howard and Sebastian Ruder. 'Universal Language Model Fine-tuning for Text Classification'. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, 15th July 2018, pp. 328–339. DOI: 10.18653/v1/P18-1031.

[154] Taku Kudo. 'Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates'. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, 15th July 2018, pp. 66–75. DOI: 10.18653/v1/P18-1007.

[155] Pranav Rajpurkar, Robin Jia and Percy Liang. 'Know What You Don't Know: Unanswerable Questions for SQuAD'. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 15th July 2018, pp. 784–789. DOI: 10.18653/v1/P18-2124.

[156] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti and Dino Pedreschi. 'A Survey of Methods for Explaining Black Box Models'. In: *ACM Comput. Surv.* 51.5 (22nd Aug. 2018), 93:1–93:42. ISSN: 0360-0300. DOI: 10.1145/3236009.

[157] Lloyd Montgomery, Daniela Damian, Tyson Bulmer and Shaikh Quader. 'Customer support ticket escalation prediction using feature engineering'. In: *Requirements Engineering* 23.3 (1st Sept. 2018), pp. 333–355. ISSN: 1432-010X. DOI: 10.1007/s00766-018-0292-3.

[158] P.S. Parmar, P.K. Biju, M. Shankar and N. Kadiresan. 'Multiclass Text Classification and Analytics for Improving Customer Support Response through different Classifiers'. In: *7th International Conference on Advances in Computing, Communications and Informatics (ICACCI 2018)*. Institute of Electrical and Electronics Engineers Inc., 19th Sept. 2018, pp. 538–542. ISBN: 978-1-5386-5314-2. DOI: 10.1109/ICACCI.2018.8554881.

[159] Zachary C. Lipton. 'The mythos of model interpretability'. In: *Commun. ACM* 61.10 (26th Sept. 2018), pp. 36–43. ISSN: 0001-0782. DOI: 10.1145/3233231.

[160] Daniel Cer et al. 'Universal Sentence Encoder for English'. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Nov. 2018, pp. 169–174. DOI: 10.18653/v1/D18-2029.

[161] Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk and Veselin Stoyanov. 'XNLI: Evaluating Cross-lingual Sentence Representations'. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 2475–2485. DOI: 10.18653/v1/D18-1269.

[162] Danilo Croce, Alexandra Zelenanska and Roberto Basili. 'Neural Learning for Question Answering in Italian: XVIIth International Conference of the Italian Association for Artificial Intelligence'. In: Nov. 2018, pp. 389–402. ISBN: 978-3-030-03839-7. DOI: 10.1007/978-3-030-03840-3_29.

[163] Taku Kudo and John Richardson. 'SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing'. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 66–71. DOI: 10.18653/v1/D18-2012.

[164] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy and Samuel Bowman. 'GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding'. In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for*

NLP. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 353–355. DOI: 10.18653/v1/W18-5446.

[165] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt and Been Kim. 'Sanity checks for saliency maps'. In: *Proceedings of 32nd International Conference on Neural Information Processing Systems*. Montréal, Canada, Dec. 2018, pp. 9525–9536.

[166] David Alvarez-Melis and Tommi S. Jaakkola. 'On the Robustness of Interpretability Methods'. In: *ICML Workshop on Human Interpretability in Machine Learning (WHI 2018)*. eprint: 1806.08049. Stockholm, Sweden, 2018. DOI: 10.48550/arXiv.1806.08049.

[167] Finale Doshi-Velez and Been Kim. 'Considerations for Evaluation and Generalization in Interpretable Machine Learning'. In: *Explainable and Interpretable Models in Computer Vision and Machine Learning*. Ed. by Hugo Jair Escalante, Sergio Escalera, Isabelle Guyon, Xavier Baró, Yağmur Güçlütürk, Umut Güçlü and Marcel van Gerven. Cham: Springer International Publishing, 2018, pp. 3–17. ISBN: 978-3-319-98131-4. DOI: 10.1007/978-3-319-98131-4_1.

[168] Sajad Sotudeh Gharebagh, Peyman Rostami and Mahmood Neshati. 'T-Shaped Mining: A Novel Approach to Talent Finding for Agile Software Teams'. In: *Advances in Information Retrieval*. Ed. by Gabriella Pasi, Benjamin Piwowarski, Leif Azzopardi and Allan Hanbury. Cham: Springer International Publishing, 2018, pp. 411–423. ISBN: 978-3-319-76941-7. DOI: 10.1007/978-3-319-76941-7_31.

[169] Jianglei Han, Ka Hian Goh, Aixin Sun and Mohammad Akbari. 'Towards Effective Extraction and Linking of Software Mentions from User-Generated Support Tickets'. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. CIKM '18. event-place: Torino, Italy. New York, NY, USA: Association for Computing Machinery, 2018, pp. 2263–2271. ISBN: 978-1-4503-6014-2. DOI: 10.1145/3269206.3272026.

[170] Patrick Kubiak and Stefan Rass. 'An Overview of Data-Driven Techniques for IT-Service-Management'. In: *IEEE Access* 6 (2018), pp. 63664–63688. DOI: 10.1109/ACCESS.2018.2875975.

[171] Volodymyr Lyubinets, Taras Boiko and Deon Nicholas. 'Automated Labeling of Bugs and Tickets Using Attention-Based Mechanisms in Recurrent Neural Networks'. In: *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*. 2018, pp. 271–275. DOI: 10.1109/DSMP.2018.8478511.

[172] S.P. Paramesh, C. Ramya and K.S. Shreedhara. 'Classifying the Unstructured IT Service Desk Tickets Using Ensemble of Classifiers'. In: *2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS)*. 2018, pp. 221–227. DOI: 10.1109/CSITSS.2018.8768734.

[173] Hao Peng et al. 'Large-Scale Hierarchical Text Classification with Recursively Regularized Deep Graph-CNN'. In: *Proceedings of the 2018 World Wide Web Conference*. WWW '18. event-place: Lyon, France. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2018, pp. 1063–1072. ISBN: 978-1-4503-5639-8. DOI: 10.1145/3178876.3186005.

[174] Alec Radford, Karthik Narasimhan, Tim Salimans and Ilya Sutskever. *Improving Language Understanding by Generative Pre-Training*. 2018. URL: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

[175] Marko Robnik-Šikonja and Marko Bohanec. 'Perturbation-Based Explanations of Prediction Models'. In: *Human and Machine Learning: Visible, Explainable, Trustworthy and Transparent*. Ed. by Jianlong Zhou and Fang Chen. Cham: Springer International Publishing, 2018, pp. 159–175. ISBN: 978-3-319-90403-0. DOI: 10.1007/978-3-319-90403-0_9.

[176] Francesco Ronzano, Francesco Barbieri, Endang Wahyu Pamungkas, Viviana Patti and Francesca Chiusaroli. 'Overview of the EVALITA 2018 Italian Emoji Prediction (ITAMoji) Task'. In: *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*. Vol. 2263. CEUR Workshop Proceedings. Turin, Italy, 2018.

[177] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò and Yoshua Bengio. 'Graph Attention Networks'. In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=rJXMpikCZ.

[178] Qing Wang, Tao Li, S. S. Iyengar, Larisa Shwartz and Genady Ya Grabarnik. 'Online IT Ticket Automation Recommendation Using Hierarchical Multi-armed Bandit Algorithms'. In: *Proceedings of the 2018 SIAM International Conference on Data Mining (SDM)*. 2018, pp. 657–665. DOI: 10.1137/1.9781611975321.74.

[179] Qing Wang, Chunqiu Zeng, S. S. Iyengar, Tao Li, Larisa Shwartz and Genady Ya Grabarnik. 'AISTAR: An Intelligent System for Online IT Ticket Automation Recommendation'. In: *IEEE International Conference on Big Data (IEEE BigData 2018), Seattle, WA, USA, December 10-13, 2018*. IEEE, 2018, pp. 1875–1884. DOI: 10.1109/BigData.2018.8622446.

[180] Akio Watanabe et al. 'Workflow Extraction for Service Operation Using Multiple Unstructured Trouble Tickets'. In: *IEICE Transactions on Information and Systems* E101.D.4 (2018), pp. 1030–1041. DOI: 10.1587/transinf.2017DAP0014.

[181] Chunyang Wu, Mark J. F. Gales, Anton Ragni, Penny Karanasou and Khe Chai Sim. 'Improving Interpretability and Regularization in Deep Learning'. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.2 (2018), pp. 256–265. DOI: 10.1109/TASLP.2017.2774919.

[182] Jian Xu and Rouying He. 'Expert recommendation for trouble ticket routing'. In: *Data & Knowledge Engineering* 116 (2018), pp. 205–218. ISSN: 0169-023X. DOI: 10.1016/j.datak.2018.06.004.

[183] Jian Xu, Rouying He, Wubai Zhou and Tao Li. 'Trouble Ticket Routing Models and Their Applications'. In: *IEEE Transactions on Network and Service Management* 15.2 (2018), pp. 530–543. DOI: 10.1109/TNSM.2018.2790956.

[184] Tim Miller. 'Explanation in artificial intelligence: Insights from the social sciences'. In: *Artificial Intelligence* 267 (1st Feb. 2019), pp. 1–38. ISSN: 0004-3702. DOI: 10.1016/j.artint.2018.07.007.

[185] Shane T. Mueller, Robert R. Hoffman, William Clancey, Abigail Emrey and Gary Klein. *Explanation in Human-AI Systems: A Literature Meta-Review, Synopsis of Key Ideas and Publications, and Bibliography for Explainable AI*. 5th Feb. 2019. DOI: 10.48550/arXiv.1902.01876.

[186] Judit Ács. *Exploring BERT's Vocabulary*. Judit Ács's blog. 19th Feb. 2019. URL: https://juditacs.github.io/2019/02/19/bert-tokenization-stats.html (visited on 01/07/2025).

[187] Liang Yao, Chengsheng Mao and Yuan Luo. 'Graph Convolutional Networks for Text Classification'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. AAAI'19/IAAI'19/EAAI'19. Issue: 01. Honolulu, Hawaii, USA: AAAI Press, 27th Feb. 2019, pp. 7370–7377. ISBN: 978-1-57735-809-1. DOI: 10.1609/aaai.v33i01.33017370.

[188] Scott M. Lundberg, Gabriel G. Erion and Su-In Lee. *Consistent Individualized Feature Attribution for Tree Ensembles*. version: 3. 7th Mar. 2019. DOI: 10.48550/arXiv.1802.03888.

[189] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes and Donald Brown. 'Text Classification Algorithms: A Survey'. In: *Information* 10.4 (23rd Apr. 2019). ISSN: 2078-2489. DOI: 10.3390/info10040150.

[190] Carrie J. Cai et al. 'Human-Centered Tools for Coping with Imperfect Algorithms During Medical Decision-Making'. In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI '19. New York, NY, USA: Association for Computing Machinery, 2nd May 2019, pp. 1–14. ISBN: 978-1-4503-5970-2. DOI: 10.1145/3290605.3300234.

[191] Cynthia Rudin. 'Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead'. In: *Nature Machine Intelligence* 1.5 (May 2019). Publisher: Nature Publishing Group, pp. 206–215. ISSN: 2522-5839. DOI: 10.1038/s42256-019-0048-x.

[192] Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. 'BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding'. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota, USA: Association for Computational Linguistics, 2nd June 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423.

[193] Sarthak Jain and Byron C. Wallace. 'Attention is not Explanation'. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. NAACL-HLT 2019. Ed. by Jill Burstein, Christy Doran and Thamar Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 3543–3556. DOI: 10.18653/v1/N19-1357.

[194] Yinhan Liu et al. 'RoBERTa: A Robustly Optimized BERT Pretraining Approach'. In: *arXiv* abs/1907.11692 (26th July 2019). DOI: 10.48550/ARXIV.1907.11692.

[195] Amirata Ghorbani, Abubakar Abid and James Zou. 'Interpretation of neural networks is fragile'. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.1 (July 2019), pp. 3681–3688. DOI: 10.1609/aaai.v33i01.33013681.

[196] Ganesh Jawahar, Benoît Sagot and Djamé Seddah. 'What Does BERT Learn about the Structure of Language?' In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 3651–3657. DOI: 10.18653/v1/P19-1356.

[197] Zeyu Li, Jyun-Yu Jiang, Yizhou Sun and Wei Wang. 'Personalized Question Routing via Heterogeneous Network Embedding'. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.1 (July 2019), pp. 192–199. DOI: 10.1609/aaai.v33i01.3301192.

[198] Atri Mandal, Nikhil Malhotra, Shivali Agarwal, Anupama Ray and Giriprasad Sridhara. 'Automated Dispatch of Helpdesk Email Tickets: Pushing the Limits with AI'. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.1 (July 2019), pp. 9381–9388. DOI: 10.1609/aaai.v33i01.33019381.

[199] Sofia Serrano and Noah A. Smith. 'Is Attention Interpretable?' In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. ACL 2019. Ed. by Anna Korhonen, David Traum and Lluís Màrquez. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 2931–2951. DOI: 10.18653/v1/P19-1282.

[200] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer and Daniel Weld. 'Errudite: Scalable, Reproducible, and Testable Error Analysis'. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 747–763. DOI: 10.18653/v1/P19-1073.

[201] Telmo Pires, Eva Schlinger and Dan Garrette. 'How Multilingual is Multilingual BERT?' In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, 18th Aug. 2019, pp. 4996–5001. DOI: 10.18653/v1/P19-1493.

[202] Emily M. Bender. 'The #BenderRule: On Naming the Languages We Study and Why It Matters'. In: *The Gradient* (14th Sept. 2019). URL: https://thegradient.pub/the-benderrule-on-naming-the-languages-we-study-and-why-it-matters/ (visited on 05/08/2025).

[203] Matei Cristian, Săcărea Christian and Tolciu Dumitru-Tudor. 'A Study in the Automation of Service Ticket Recognition using Natural Language Processing'. In: *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. Sept. 2019, pp. 1–6. DOI: 10.23919/SOFTCOM.2019.8903676.

[204] Hirotaka Tanaka, Hiroyuki Shinnou, Rui Cao, Jing Bai and Wen Ma. 'Document Classification by Word Embeddings of BERT'. In: *16th International Conference of the Pacific Association for Computational Linguistics, PACLING 2019*. Ed. by Le-Minh Nguyen, Xuan-Hieu Phan, Kôiti Hasida and Satoshi Tojo. Hanoi, Vietnam: Springer Singapore, 11th Oct. 2019, pp. 145–154. DOI: 10.1007/978-981-15-6168-9_13.

[205] W. James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl and Bin Yu. 'Definitions, methods, and applications in interpretable machine learning'. In: *Proceedings of the National Academy of Sciences* 116.44 (29th Oct. 2019). Publisher: Proceedings of the National Academy of Sciences, pp. 22071–22080. DOI: 10.1073/pnas.1900654116.

[206] Nils Reimers and Iryna Gurevych. 'Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks'. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. DOI: 10.18653/v1/D19-1410.

[207] Sarah Wiegreffe and Yuval Pinter. 'Attention is not not Explanation'. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. EMNLP-IJCNLP 2019. Ed. by Kentaro Inui, Jing Jiang, Vincent Ng and Xiaojun Wan. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 11–20. DOI: 10.18653/v1/D19-1002.

[208] Chaofan Chen, Oscar Li, Chaofan Tao, Alina Jade Barnett, Jonathan Su and Cynthia Rudin. 'This looks like that: deep learning for interpretable image recognition'. In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. 801. Red Hook, NY, USA: Curran Associates Inc., 8th Dec. 2019, pp. 8930–8941. (Visited on 21/07/2025).

[209] Ann-Kathrin Dombrowski, Maximilian Alber, Christopher J. Anders, Marcel Ackermann, Klaus-Robert Müller and Pan Kessel. 'Explanations can be manipulated and geometry is to blame'. In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. 1217. Red Hook, NY, USA: Curran Associates Inc., 8th Dec. 2019, pp. 13589–13600. (Visited on 02/09/2025).

[210] Alex Wang et al. 'SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems'. In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox and R. Garnett. 294. Vancouver, Canada: Curran Associates Inc., 8th Dec. 2019, pp. 3266–3280.

[211] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov and Quoc V. Le. 'XLNet: Generalized Autoregressive Pretraining for Language Understanding'. In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Issue: 517. Vancouver, Canada: Curran Associates Inc., 8th Dec. 2019.

[212] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Sai Suggala, David I. Inouye and Pradeep Ravikumar. 'On the (in)fidelity and sensitivity of explanations'. In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. 984. Red Hook, NY, USA: Curran Associates Inc., 8th Dec. 2019, pp. 10967–10978. (Visited on 02/09/2025).

[213] Anne Abeillé, Lionel Clément and Loïc Liégeois. 'Un corpus annoté pour le français: le French Treebank'. In: *Revue TAL* 60.2 (2019). Publisher: ATALA (Association pour le Traitement Automatique des Langues), pp. 19–43.

[214] Rami Aly, Steffen Remus and Chris Biemann. 'Hierarchical Multi-label Classification of Text with Capsule Networks'. In: *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28 - August 2, 2019, Volume 2: Student Research Workshop*. Ed. by Fernando Alva-Manchego, Eunsol Choi and Daniel Khashabi. Association for Computational Linguistics, 2019, pp. 323–330. DOI: 10.18653/v1/p19-2045.

[215] A. Bhowmik, S. Paul, D. Usha Nandini and S. Prince Mary. 'Study of the Management of Tickets in IT Administration'. In: *Conference of 1st International Conference on Frontiers in Materials and Smart System Technologies*. Vol. 590. ISSN: 17578981 Issue: 1. Institute of Physics Publishing, 2019, p. 012006. DOI: 10.1088/1757-899X/590/1/012006.

[216] Danielle Caled, Miguel Won, Bruno Martins and Mário J. Silva. 'A Hierarchical Label Network for Multi-label EuroVoc Classification of Legislative Contents'. In: *Digital Libraries for Open Knowledge*. Ed. by Antoine Doucet, Antoine Isaac, Koraljka Golub, Trond Aalberg and Adam Jatowt. Cham: Springer International Publishing, 2019, pp. 238–252. ISBN: 978-3-030-30760-8. DOI: 10.1007/978-3-030-30760-8_21.

[217] Sofi Defiyanti, Edi Winarko and Sigit Priyanta. 'A Survey of Hierarchical Classification Algorithms with Big-Bang Approach'. In: *2019 5th International Conference on Science and Technology (ICST)*. Vol. 1. 2019, pp. 1–6. DOI: 10.1109/ICST47872.2019.9166313.

[218] Wei Huang et al. 'Hierarchical Multi-label Text Classification: An Attention-based Recurrent Network Approach'. In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*. Ed. by Wenwu Zhu et al. ACM, 2019, pp. 1051–1060. DOI: 10.1145/3357384.3357885.

[219] Omayma Husain, Naomie Salim, Rose Alinda Alias, Samah Abdelsalam and Alzubair Hassan. 'Expert Finding Systems: A Systematic Review'. In: *Applied Sciences* 9.20 (2019). ISSN: 2076-3417. DOI: 10.3390/app9204250.

[220] Aditi Jain, Pratishtha Yadav and Hira Javed. 'Equivoque: Detection and Interpretation of English Puns'. In: *2019 8th International Conference System Modeling and Advancement in Research Trends (SMART)*. 2019, pp. 262–265. DOI: 10.1109/SMART46866.2019.9117433.

[221] Rafael Kallis, Andrea Di Sorbo, Gerardo Canfora and Sebastiano Panichella. 'Ticket Tagger: Machine Learning Driven Issue Classification'. In: *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. 2019, pp. 406–409. DOI: 10.1109/ICSME.2019.00070.

[222] Pieter-Jan Kindermans et al. 'The (Un)reliability of Saliency Methods'. In: *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Cham: Springer International Publishing, 2019, pp. 267–280. ISBN: 978-3-030-28954-6. DOI: 10.1007/978-3-030-28954-6_14.

[223] Atri Mandal, Shivali Agarwal, Nikhil Malhotra, Giriprasad Sridhara, Anupama Ray and Daivik Swarup. 'Improving IT Support by Enhancing Incident Management Process with Multi-modal Analysis'. In: *Service-Oriented Computing*. Ed. by Sami Yangui, Ismael Bouassida Rodriguez, Khalil Drira and Zahir Tari. Cham: Springer International Publishing, 2019, pp. 431–446. ISBN: 978-3-030-33702-5. DOI: 10.1007/978-3-030-33702-5_33.

[224] Senthil Mani, Anush Sankaran and Rahul Aralikatte. 'DeepTriage: Exploring the Effectiveness of Deep Learning for Bug Triaging'. In: *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*. CoDS-COMAD '19. event-place: Kolkata, India. New York, NY, USA: Association for Computing Machinery, 2019, pp. 171–179. ISBN: 978-1-4503-6207-8. DOI: 10.1145/3297001.3297023.

[225] Yuning Mao, Jingjing Tian, Jiawei Han and Xiang Ren. 'Hierarchical Text Classification with Reinforced Label Assignment'. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Ed. by Kentaro Inui, Jing Jiang, Vincent Ng and Xiaojun Wan. Association for Computational Linguistics, 2019, pp. 445–455. DOI: 10.18653/v1/D19-1042.

[226] M.A. Mukunthan and S. Selvakumar. 'Multilevel Petri net-based ticket assignment and IT management for improved IT organization support'. In: *Concurrency and Computation: Practice and Experience* 31.14 (2019), e5297. DOI: 10.1002/cpe.5297.

[227] Adam Paszke et al. 'PyTorch: An Imperative Style, High-Performance Deep Learning Library'. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[228] Malgorzata Pikies and Junade Ali. 'String similarity algorithms for a ticket classification system'. In: *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*. 2019, pp. 36–41. DOI: 10.1109/CoDIT.2019.8820497.

[229] Roger A. Stein, Patrícia Augustin Jaques and João Francisco Valiati. 'An analysis of hierarchical text classification using word embeddings'. In: *Inf. Sci.* 471 (2019), pp. 216–232. DOI: 10.1016/j.ins.2018.09.001.

[230] Qing Wang, Larisa Shwartz, Genady Ya. Grabarnik, Michael Nidd and Jinho Hwang. 'Leveraging AI in Service Automation Modeling: From Classical AI Through Deep Learning to Combination Models'. In: *Service-Oriented Computing*. Ed. by Sami Yangui, Ismael Bouassida Rodriguez, Khalil Drira and Zahir Tari. Cham: Springer International Publishing, 2019, pp. 186–201. ISBN: 978-3-030-33702-5. DOI: 10.1007/978-3-030-33702-5_14.

[231] Colin Werner, Ze Shi Li and Daniela Damian. 'Can a Machine Learn Through Customer Sentiment?: A Cost-Aware Approach to Predict Support Ticket Escalations'. In: *IEEE Software* 36.5 (2019), pp. 38–45. DOI: 10.1109/MS.2019.2923408.

[232] Ian James Bruce Young, Saturnino Luz and Nazir Lone. 'A systematic review of natural language processing for classification tasks in the field of incident reporting and adverse event analysis'. In: *International Journal of Medical Informatics* 132.103971 (2019). ISSN: 1386-5056. DOI: 10.1016/j.ijmedinf.2019.103971.

[233] Cristina Bosco, Silvia Ballarè, Massimo Cerruti, Eugenio Goria and Caterina Mauri. 'KIPoS @ EVALITA2020: Overview of the Task on KIParla Part of Speech Tagging'. In: *Proceedings of the Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*. Vol. 2765. CEUR Workshop Proceedings. Jan. 2020. DOI: 10.4000/books.aaccademia.7743.

[234] Himabindu Lakkaraju and Osbert Bastani. '"How do I fool you?": Manipulating User Trust via Misleading Black Box Explanations'. In: *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. AIES '20. New York, NY, USA: Association for Computing Machinery, 7th Feb. 2020, pp. 79–85. ISBN: 978-1-4503-7110-0. DOI: 10.1145/3375627.3375833.

[235] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh and Himabindu Lakkaraju. 'Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods'. In: *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. AIES '20. New York, NY, USA: Association for Computing Machinery, 7th Feb. 2020, pp. 180–186. ISBN: 978-1-4503-7110-0. DOI: 10.1145/3375627.3375830.

[236] Muhammad Pervez Akhter, Zheng Jiangbin, Irfan Raza Naqvi, Mohammed Abdelmajeed, Atif Mehmood and Muhammad Tariq Sadiq. 'Document-Level Text Classification Using Single-Layer Multisize Filters Convolutional Neural Network'. In: *IEEE Access* 8 (27th Feb. 2020), pp. 42689–42707. DOI: 10.1109/ACCESS.2020.2976744.

[237] Changhan Wang, Kyunghyun Cho and Jiatao Gu. 'Neural Machine Translation with Byte-Level Subwords'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. Issue: 05. New York, New York, USA, 7th Apr. 2020, pp. 9154–9160. DOI: 10.1609/aaai.v34i05.6451.

[238] Boli Chen, Xin Huang, Lin Xiao, Zixin Cai and Liping Jing. 'Hyperbolic Interaction Model for Hierarchical Multi-Label Classification'. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.5 (Apr. 2020), pp. 7496–7503. DOI: 10.1609/aaai.v34i05.6247.

[239] Michael Powell, Jamison A Rotz and Kevin D O'Malley. 'How Machine Learning Is Improving U.S. Navy Customer Support'. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 34.8 (Apr. 2020), pp. 13188–13195. DOI: 10.1609/aaai.v34i08.7023.

[240] Yaakov HaCohen-Kerner, Daniel Miller and Yair Yigal. 'The influence of preprocessing on text classification using a bag-of-words representation'. In: *PLOS ONE* 15.5 (1st May 2020). Publisher: Public Library of Science, pp. 1–22. DOI: 10.1371/journal.pone.0232525.

[241] Christoph Leonhardt and Andreas Blätte. 'ParisParl Corpus of Parliamentary Debates [dataset]'. In: (10th May 2020). Publisher: Zenodo Version Number: v0.1.0. DOI: 10.5281/zenodo.3819374.

[242] Maxime Amblard, Clément Beysson, Philippe de Groote, Bruno Guillaume and Sylvain Pogodalla. 'A French Version of the FraCaS Test Suite'. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 11th May 2020, pp. 5887–5895. ISBN: 979-10-95546-34-4. URL: https://aclanthology.org/2020.lrec-1.721.

[243] Patricia Chiril, Véronique Moriceau, Farah Benamara, Alda Mari, Gloria Origgi and Marlène Coulomb-Gully. 'An Annotated Corpus for Sexism Detection in French Tweets'. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 11th May 2020, pp. 1397–1403. ISBN: 979-10-95546-34-4.

[244] Marc Evrard, Rémi Uro, Nicolas Hervé and Béatrice Mazoyer. 'French Tweet Corpus for Automatic Stance Detection'. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 11th May 2020, pp. 6317–6322. ISBN: 979-10-95546-34-4.

[245] Rachel Keraron et al. 'Project PIAF: Building a Native French Question-Answering Dataset'. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 11th May 2020, pp. 5481–5490. ISBN: 979-10-95546-34-4.

[246] Hang Le et al. 'FlauBERT: Unsupervised Language Model Pre-training for French'. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 11th May 2020, pp. 2479–2490. ISBN: 979-10-95546-34-4. URL: https://aclanthology.org/2020.lrec-1.302.

[247] Béatrice Mazoyer, Julia Cagé, Nicolas Hervé and Céline Hudelot. 'A French Corpus for Event Detection on Twitter'. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, 11th May 2020, pp. 6220–6227. ISBN: 979-10-95546-34-4.

[248] Jining Yan, Lin Mu, Lizhe Wang, Rajiv Ranjan and Albert Y. Zomaya. 'Temporal Convolutional Networks for the Advance Prediction of ENSO'. In: *Scientific Reports* 10.1 (15th May 2020), p. 8055. ISSN: 2045-2322. DOI: 10.1038/s41598-020-65070-5.

[249] Alejandro Barredo Arrieta et al. 'Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI'. In: *Information Fusion* 58 (1st June 2020), pp. 82–115. ISSN: 1566-2535. DOI: 10.1016/j.inffus.2019.12.012.

[250] Xueping Ni, Changying Li, Huanyu Jiang and Fumiomi Takeda. 'Deep learning image segmentation and extraction of blueberry fruit traits associated with harvestability and yield'. In: *Hortic. Res.* 7.1 (1st July 2020), p. 110. ISSN: 2052-7276. DOI: 10.1038/s41438-020-0323-3.

[251] Alexis Conneau et al. 'Unsupervised Cross-lingual Representation Learning at Scale'. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 5th July 2020, pp. 8440–8451. DOI: 10.18653/v1/2020.acl-main.747.

[252] Qi Qin, Wenpeng Hu and Bing Liu. 'Feature Projection for Improved Text Classification'. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 5th July 2020, pp. 8161–8171. DOI: 10.18653/v1/2020.acl-main.726.

[253] Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat and Melvin Johnson. 'XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalisation'. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. Online: PMLR, 13th July 2020, pp. 4411–4421.

[254] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim and Percy Liang. 'Concept Bottleneck Models'. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13th July 2020, pp. 5338–5348. URL: https://proceedings.mlr.press/v119/koh20a.html.

[255] Umang Bhatt, Adrian Weller and José M. F. Moura. 'Evaluating and aggregating feature-based model explanations'. In: *Proceedings of the twenty-ninth international joint conference on artificial intelligence, IJCAI-20*. Ed. by Christian Bessiere. International Joint Conferences on Artificial Intelligence Organization, July 2020, pp. 3016–3022. DOI: 10.24963/ijcai.2020/417.

[256] Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher and Byron C. Wallace. 'ERASER: A Benchmark to Evaluate Rationalized NLP Models'. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Ed. by Dan Jurafsky, Joyce Chai, Natalie Schluter and Joel Tetreault. Online: Association for Computational Linguistics, July 2020, pp. 4443–4458. DOI: 10.18653/v1/2020.acl-main.408.

[257] Peter Hase and Mohit Bansal. 'Evaluating Explainable AI: Which Algorithmic Explanations Help Users Predict Model Behavior?' In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Ed. by Dan Jurafsky, Joyce Chai, Natalie Schluter and Joel Tetreault. Online: Association for Computational Linguistics, July 2020, pp. 5540–5552. DOI: 10.18653/v1/2020.acl-main.491.

[258] Alon Jacovi and Yoav Goldberg. 'Towards Faithfully Interpretable NLP Systems: How Should We Define and Evaluate Faithfulness?' In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Ed. by Dan Jurafsky, Joyce Chai, Natalie Schluter and Joel Tetreault. Online: Association for Computational Linguistics, July 2020, pp. 4198–4205. DOI: 10.18653/v1/2020.acl-main.386.

[259] Sawan Kumar and Partha Talukdar. 'NILE : Natural Language Inference with Faithful Natural Language Explanations'. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. ACL 2020. Ed. by Dan Jurafsky, Joyce Chai, Natalie Schluter and Joel Tetreault. Online: Association for Computational Linguistics, July 2020, pp. 8730–8742. DOI: 10.18653/v1/2020.acl-main.771.

[260] Mike Lewis et al. 'BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension'. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Conference Name: The 58th Annual Meeting Of The Association For Computational Linguistics. July 2020. URL: https://virtual.acl2020.org/paper_main.703.html (visited on 06/09/2025).

[261] Natalia Grabar, Clément Dalloux and Vincent Claveau. 'CAS: corpus of clinical cases in French'. In: *Journal of Biomedical Semantics* 11.1 (6th Aug. 2020), p. 7. ISSN: 2041-1480. DOI: 10.1186/s13326-020-00225-x.

[262] Shivali Agarwal, Jayachandu Bandlamudi, Atri Mandal, Anupama Ray and Giriprasad Sridhara. 'Automated Assignment of Helpdesk Email Tickets: An AI Lifecycle Case Study'. In: *AI Magazine* 41.3 (Sept. 2020), pp. 45–62. DOI: 10.1609/aimag.v41i3.5321.

[263] José Jiménez-Luna, Francesca Grisoni and Gisbert Schneider. 'Drug discovery with explainable artificial intelligence'. In: *Nature Machine Intelligence* 2.10 (Oct. 2020). Publisher: Nature Publishing Group, pp. 573–584. ISSN: 2522-5839. DOI: 10.1038/s42256-020-00236-4.

[264] Ian Tenney et al. 'The Language Interpretability Tool: Extensible, Interactive Visualizations and Analysis for NLP Models'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 107–118. DOI: 10.18653/v1/2020.emnlp-demos.15.

[265] Martin d'Hoffschmidt, Wacim Belblidia, Quentin Heinrich, Tom Brendlé and Maxime Vidal. 'FQuAD: French Question Answering Dataset'. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, 16th Nov. 2020, pp. 1193–1208. DOI: 10.18653/v1/2020.findings-emnlp.107.

[266] Phillip Keung, Yichao Lu, György Szarvas and Noah A. Smith. 'The Multilingual Amazon Reviews Corpus'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 16th Nov. 2020, pp. 4563–4568. DOI: 10.18653/v1/2020.emnlp-main.369.

[267] Yaobo Liang et al. 'XGLUE: A New Benchmark Dataset for Cross-lingual Pre-training, Understanding and Generation'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 16th Nov. 2020, pp. 6008–6018. DOI: 10.18653/v1/2020.emnlp-main.484.

[268] Alessandro Raganato, Tommaso Pasini, Jose Camacho-Collados and Mohammad Taher Pilehvar. 'XL-WiC: A Multilingual Benchmark for Evaluating Semantic Contextualization'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 16th Nov. 2020, pp. 7193–7206. DOI: 10.18653/v1/2020.emnlp-main.584.

[269] Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski and Jacopo Staiano. 'MLSUM: The Multilingual Summarization Corpus'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 16th Nov. 2020, pp. 8051–8067. DOI: 10.18653/v1/2020.emnlp-main.647.

[270] Thomas Wolf et al. 'Transformers: State-of-the-Art Natural Language Processing'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, 16th Nov. 2020, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6.

[271] Bichen Wu et al. *Visual Transformers: Token-based Image Representation and Processing for Computer Vision*. 20th Nov. 2020. DOI: 10.48550/arXiv.2006.03677.

[272] Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa Anke and Leonardo Neves. 'TweetEval: Unified Benchmark and Comparative Evaluation for Tweet Classification'. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Findings 2020. Ed. by Trevor Cohn, Yulan He and Yang Liu. Online: Association for Computational Linguistics, Nov. 2020, pp. 1644–1650. DOI: 10.18653/v1/2020.findings-emnlp.148.

[273] Wenhu Chen, Yu Su, Xifeng Yan and William Yang Wang. 'KGPT: Knowledge-Grounded Pre-Training for Data-to-Text Generation'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. EMNLP 2020. Ed. by Bonnie Webber, Trevor Cohn, Yulan He and Yang Liu. Online: Association for Computational Linguistics, Nov. 2020, pp. 8635–8648. DOI: 10.18653/v1/2020.emnlp-main.697.

[274] Lorella Viola and Antonio Maria Fiscarelli. 'From digitised sources to digital data: Behind the scenes of (critically) enriching a digital heritage collection'. In: *Proceedings of the International Conference Collect and Connect: Archives and Collections in a Digital Age*. Vol. 2810. CEUR Workshop Proceedings. ISSN: 1613-0073. Online, Nov. 2020.

[275] Wietse de Vries, Andreas van Cranenburgh and Malvina Nissim. 'What's so special about BERT's layers? A closer look at the NLP pipeline in monolingual and multilingual models'. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, Nov. 2020, pp. 4339–4350. DOI: 10.18653/v1/2020.findings-emnlp.389.

[276] Eric Wallace, Matt Gardner and Sameer Singh. 'Interpreting Predictions of NLP Models'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*. Ed. by Aline Villavicencio and Benjamin Van Durme. Online: Association for Computational Linguistics, Nov. 2020, pp. 20–23. DOI: 10.18653/v1/2020.emnlp-tutorials.3.

[277] Zirui Wang, Zachary C. Lipton and Yulia Tsvetkov. 'On Negative Interference in Multilingual Models: Findings and A Meta-Learning Treatment'. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. EMNLP 2020. Ed. by Bonnie Webber, Trevor Cohn, Yulan He and Yang Liu. Online: Association for Computational Linguistics, Nov. 2020, pp. 4438–4450. DOI: 10.18653/v1/2020.emnlp-main.359.

[278] Larissa Chazette and Kurt Schneider. 'Explainability as a non-functional requirement: challenges and recommendations'. In: *Requirements Engineering* 25.4 (1st Dec. 2020), pp. 493–514. ISSN: 1432-010X. DOI: 10.1007/s00766-020-00333-1.

[279] Tom Brown et al. 'Language Models are Few-Shot Learners'. In: *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*. Vol. 33. Online: Curran Associates, Inc., 6th Dec. 2020, pp. 1877–1901. ISBN: 978-1-7138-2954-6.

[280] Elisabetta Fersini, Debora Nozza and Paolo Rosso. 'AMI @ EVALITA2020: Automatic Misogyny Identification'. In: *Proceedings of the Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2020)*. Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Ed. by Valerio Basile, Danilo Croce, Maria Di Maro and Lucia C. Passaro. Vol. 2765. CEUR Workshop Proceedings. ISSN: 1613-0073. Online event, December: CEUR, Dec. 2020. URL: https://ceur-ws.org/Vol-2765/#161 (visited on 05/08/2025).

[281] Manuela Sanguinetti et al. 'HaSpeeDe 2 @ EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task'. In: *Proceedings of the Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*. Vol. 2765. CEUR Workshop Proceedings. Online, Dec. 2020. DOI: 10.4000/BOOKS.AACCADEMIA.6897.

[282] Dominique Brunato, Cristiano Chesi, Felice Dell'Orletta, Simonetta Montemagni, Giulia Venturi and Roberto Zamparelli. 'AcCompl-it @ EVALITA2020: Overview of the Acceptability & Complexity Evaluation Task for Italian'. In: *Proceedings of the Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*. Vol. 2765. CEUR Workshop Proceedings. Online, 2020.

[283] Lorenzo De Mattei, Graziella De Martino, Andrea Iovine, Alessio Miaschi, Marco Polignano and Giulia Rambelli. 'ATE ABSITA@ EVALITA2020: Overview of the Aspect Term Extraction and Aspect-based Sentiment Analysis Task'. In: *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020), Online. CEUR. org* (2020).

[284] Nicolas Ferland, Wenting Sun, Xuancheng Fan, Lule Yu and Jieneng Yang. 'Automatically Resolve Trouble Tickets with Hybrid NLP'. In: *2020 IEEE Symposium Series on Computational Intelligence, SSCI 2020, Canberra, Australia, December 1-4, 2020*. IEEE, 2020, pp. 1334–1340. DOI: 10.1109/SSCI47803.2020.9308327.

[285] Jibing Gong et al. 'Hierarchical Graph Transformer-Based Deep Learning Model for Large-Scale Multi-Label Text Classification'. In: *IEEE Access* 8 (2020), pp. 30885–30896. DOI: 10.1109/ACCESS.2020.2972751.

[286] Jianglei Han, Jing Li and Aixin Sun. 'UFTR: A Unified Framework for Ticket Routing'. In: *arXiv* (2020). Publisher: arXiv. DOI: 10.48550/ARXIV.2003.00703.

[287] Jianglei Han and Aixin Sun. 'DeepRouting: A Deep Neural Network Approach for Ticket Routing in Expert Network'. In: *2020 IEEE International Conference on Services Computing (SCC)*. 2020, pp. 386–393. DOI: 10.1109/SCC49832.2020.00057.

[288] Jared Kaplan et al. 'Scaling Laws for Neural Language Models'. In: *arXiv* abs/2001.08361 (2020). DOI: 10.48550/ARXIV.2001.08361.

[289] Mirko Lai, Alessandra Teresa Cignarella, Delia Irazú Hernández Farías, Cristina Bosco, Viviana Patti and Paolo Rosso. 'Multilingual stance detection in social media political debates'. In: *Computer Speech & Language* 63 (2020), p. 101075. ISSN: 0885-2308. DOI: 10.1016/j.csl.2020.101075.

[290] Yinglong Ma, Jingpeng Zhao and Beihong Jin. 'A Hierarchical Fine-Tuning Approach Based on Joint Embedding of Words and Parent Categories for Hierarchical Multi-label Text Classification'. In: *Artificial Neural Networks and Machine Learning - ICANN 2020 - 29th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 15-18, 2020, Proceedings, Part II*. Ed. by Igor Farkas, Paolo Masulli and Stefan Wermter. Vol. 12397. Lecture Notes in Computer Science. Springer, 2020, pp. 746–757. DOI: 10.1007/978-3-030-61616-8_60.

[291] Stefano Menini, Giovanni Moretti, Rachele Sprugnoli and Sara Tonelli. 'DaDoEval @ EVALITA 2020: Same-Genre and Cross-Genre Dating of Historical Documents'. In: *Proceedings of the Seventh Evaluation Campaign of Natural Language Processing and Speech Tools for Italian*. Vol. 2765. CEUR Workshop Proceedings. Online, 2020.

[292] Colin Raffel et al. 'Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer'. In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

[293] Aleksandra Revina, Krisztian Buza and Vera G. Meister. 'IT Ticket Classification: The Simpler, the Better'. In: *IEEE Access* 8 (2020), pp. 193380–193395. DOI: 10.1109/ACCESS.2020.3032840.

[294] Ribana Roscher, Bastian Bohn, Marco F. Duarte and Jochen Garcke. 'Explainable Machine Learning for Scientific Insights and Discoveries'. In: *IEEE Access* 8 (2020), pp. 42200–42216. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2976199.

[295] Jian Xu, Jiapeng Mu and Gaorong Chen. 'A multi-view similarity measure framework for trouble ticket mining'. In: *Data & Knowledge Engineering* 127.101800 (2020). ISSN: 0169-023X. DOI: 10.1016/j.datak.2020.101800.

[296] Jie Zhou et al. 'Graph neural networks: A review of methods and applications'. In: *AI Open* 1 (2020), pp. 57–81. ISSN: 2666-6510. DOI: 10.1016/j.aiopen.2021.01.001.

[297] Jie Zhou et al. 'Hierarchy-Aware Global Model for Hierarchical Text Classification'. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Ed. by Dan Jurafsky, Joyce Chai, Natalie Schluter and Joel R. Tetreault. Association for Computational Linguistics, 2020, pp. 1106–1117. DOI: 10.18653/v1/2020.acl-main.104.

[298] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song and Jacob Steinhardt. *Measuring Massive Multitask Language Understanding*. 12th Jan. 2021. DOI: 10.48550/arXiv.2009.03300.

[299] Pantelis Linardatos, Vasilis Papastefanopoulos and Sotiris Kotsiantis. 'Explainable AI: A Review of Machine Learning Interpretability Methods'. In: *Entropy* 23.1 (Jan. 2021). Number: 1 Publisher: Multidisciplinary Digital Publishing Institute, p. 18. ISSN: 1099-4300. DOI: 10.3390/e23010018.

[300] Zhongang Qi, Saeed Khorram and Li Fuxin. 'Embedding deep networks into visual explanations'. In: *Artificial Intelligence* 292 (1st Mar. 2021), p. 103435. ISSN: 0004-3702. DOI: 10.1016/j.artint.2020.103435.

[301] Fabio Tamburini. 'How "BERTology" Changed the State-of-the-Art also for Italian NLP'. In: *Proceedings of the Seventh Italian Conference on Computational Linguistics, CLiC-it 2020*. Vol. 2769. CEUR Workshop Proceedings. Online, 1st Mar. 2021. DOI: https://dx.doi.org/10.4000/books.aaccademia.8920.

[302] Lorella Viola and Antonio Maria Fiscarelli. *ChroniclItaly 3.0. A deep-learning, contextually enriched digital heritage collection of Italian immigrant newspapers published in the USA, 1898-1936*. 11th Mar. 2021. URL: https://zenodo.org/records/4596345 (visited on 31/07/2025).

[303] Negin Ghasemi, Ramin Fatourechi and Saeedeh Momtazi. 'User Embedding for Expert Finding in Community Question Answering'. In: *ACM Transactions on Knowledge Discovery from Data* 15.4 (Mar. 2021). Place: New York, NY, USA Publisher: Association for Computing Machinery. ISSN: 1556-4681. DOI: 10.1145/3441302.

[304] Sina Mohseni, Jeremy E Block and Eric Ragan. 'Quantitative Evaluation of Machine Learning Explanations: A Human-Grounded Benchmark'. In: *Proceedings of the 26th International Conference on Intelligent User Interfaces*. IUI '21. New York, NY, USA: Association for Computing Machinery, 14th Apr. 2021, pp. 22–31. ISBN: 978-1-4503-8017-1. DOI: 10.1145/3397481.3450689.

[305] Enja Kokalj, Blaž Škrlj, Nada Lavrač, Senja Pollak and Marko Robnik-Šikonja. 'BERT meets Shapley: Extending SHAP Explanations to Transformer-based Classifiers'. In: *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation*. Online: Assoc. for Computational Linguistics, Apr. 2021, pp. 16–21.

[306] Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu and Jianfeng Gao. 'Deep Learning–Based Text Classification: A Comprehensive Review'. In: *ACM Comput. Surv.* 54.3 (Apr. 2021). Place: New York, NY, USA Publisher: Association for Computing Machinery, pp. 1–40. ISSN: 0360-0300. DOI: 10.1145/3439726.

[307] Sampo Pyysalo, Jenna Kanerva, Antti Virtanen and Filip Ginter. 'WikiBERT Models: Deep Transfer Learning for Many Languages'. In: *Proceedings of the 23rd Nordic Conference on Computational Linguistics (NoDaLiDa)*. NoDaLiDa 2021. Ed. by Simon Dobnik and Lilja Øvrelid. Reykjavik, Iceland (Online): Linköping University Electronic Press, Sweden, 31st May 2021, pp. 1–10. URL: https://aclanthology.org/2021.nodalida-main.1/ (visited on 08/08/2025).

[308] Mattia Samory, Indira Sen, Julian Kohne, Fabian Flöck and Claudia Wagner. '"Call me sexist, but..." : Revisiting Sexism Detection Using Psychological Scales and Adversarial Samples'. In: *Proceedings of the International AAAI Conference on Web and Social Media* 15.1 (May 2021), pp. 573–584. DOI: 10.1609/icwsm.v15i1.18085.

[309] Yingren Huang, Jiaojiao Chen, Shaomin Zheng, Yun Xue and Xiaohui Hu. 'Hierarchical multi-attention networks for document classification'. In: *International Journal of Machine Learning and Cybernetics* 12.6 (1st June 2021), pp. 1639–1647. ISSN: 1868-808X. DOI: 10.1007/s13042-020-01260-x.

[310] Shang-Chi Tsai, Chao-Wei Huang and Yun-Nung Chen. 'Modeling Diagnostic Label Correlation for Automatic ICD Coding'. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, June 2021, pp. 4043–4052. DOI: 10.18653/v1/2021.naacl-main.318.

[311] Rita Sevastjanova, Aikaterini-Lida Kalouli, Christin Beck, Hanna Schäfer and Mennatallah El-Assady. 'Explaining Contextualization in Language Models using Visual Analytics'. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, 1st Aug. 2021, pp. 464–476. DOI: 10.18653/v1/2021.acl-long.39.

[312] Vinayshekhar Bannihatti Kumar, Mohan Yarramsetty, Sharon Sun and Anukul Goel. 'SupportNet: Neural Networks for Summary Generation and Key Segment Extraction from Technical Support Tickets'. In: *Proceedings of The 4th Workshop on e-Commerce and NLP*. Online: Association for Computational Linguistics, 5th Aug. 2021, pp. 164–173. DOI: 10.18653/v1/2021.ecnlp-1.20.

[313] J. A. Meaney, Steven Wilson, Luis Chiruzzo, Adam Lopez and Walid Magdy. 'SemEval 2021 Task 7: HaHackathon, Detecting and Rating Humor and Offense'. In: *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*. Ed. by Alexis Palmer, Nathan Schneider, Natalie Schluter, Guy Emerson, Aurelie Herbelot and Xiaodan Zhu. Online: Association for Computational Linguistics, Aug. 2021, pp. 105–119. DOI: 10.18653/v1/2021.semeval-1.9.

[314] Alexis Palmer, Nathan Schneider, Natalie Schluter, Guy Emerson, Aurelie Herbelot and Xiaodan Zhu, eds. *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*. Online: Association for Computational Linguistics, Aug. 2021. ISBN: 978-1-954085-70-1.

[315] Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder and Iryna Gurevych. 'How Good is Your Tokenizer? On the Monolingual Performance of Multilingual Language Models'. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 3118–3135. DOI: 10.18653/v1/2021.acl-long.243.

[316] Sina Mohseni, Niloofar Zarei and Eric D. Ragan. 'A Multidisciplinary Survey and Framework for Design and Evaluation of Explainable AI Systems'. In: *ACM Trans. Interact. Intell. Syst.* 11.3 (3rd Sept. 2021), 24:1–24:45. ISSN: 2160-6455. DOI: 10.1145/3387166.

[317] Guanhua Chen et al. 'Zero-Shot Cross-Lingual Transfer of Neural Machine Translation with Multilingual Pretrained Encoders'. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2021. Ed. by Marie-Francine Moens, Xuanjing Huang, Lucia Specia and Scott Wen-tau Yih. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 15–26. DOI: 10.18653/v1/2021.emnlp-main.2.

[318] Dheeraj Rajagopal, Vidhisha Balachandran, Eduard H Hovy and Yulia Tsvetkov. 'SELFEXPLAIN: a self-explaining architecture for neural text classifiers'. In: *Proceedings of the 2021 conference on empirical methods in natural language processing*. Ed. by Marie-Francine Moens, Xuanjing Huang, Lucia Specia and Scott Wen-tau Yih. Online and Punta Cana, Dominican Repub-

lic: Association for Computational Linguistics, Nov. 2021, pp. 836–850. DOI: 10.18653/v1/2021.emnlp-main.64.

[319] Sarah Wiegreffe, Ana Marasović and Noah A. Smith. 'Measuring Association Between Labels and Free-Text Rationales'. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2021. Ed. by Marie-Francine Moens, Xuanjing Huang, Lucia Specia and Scott Wen-tau Yih. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 10266–10284. DOI: 10.18653/v1/2021.emnlp-main.804.

[320] Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana and Geoffrey E. Hinton. 'Neural additive models: interpretable machine learning with neural nets'. In: *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS '21. Red Hook, NY, USA: Curran Associates Inc., 6th Dec. 2021, pp. 4699–4711. ISBN: 978-1-7138-4539-3. (Visited on 21/07/2025).

[321] Sabrina J. Mielke et al. 'Between words and characters: A Brief History of Open-Vocabulary Modeling and Tokenization in NLP'. In: *arXiv* abs/2112.10508 (20th Dec. 2021). DOI: 10.48550/arXiv.2112.10508.

[322] Mulugeta Weldezgina Asres, Million Abayneh Mengistu, Pino Castrogiovanni, Lorenzo Bottaccioli, Enrico Macii, Edoardo Patti and Andrea Acquaviva. 'Supporting Telecommunication Alarm Management System With Trouble Ticket Prediction'. In: *IEEE Transactions on Industrial Informatics* 17.2 (2021), pp. 1459–1469. DOI: 10.1109/TII.2020.2996942.

[323] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major and Shmargaret Shmitchell. 'On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?' In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. FAccT '21. event-place: Virtual Event, Canada. New York, NY, USA: Association for Computing Machinery, 2021, pp. 610–623. ISBN: 978-1-4503-8309-7. DOI: 10.1145/3442188.3445922.

[324] Washington Cunha et al. 'On the cost-effectiveness of neural and non-neural approaches and representations for text classification: A comprehensive comparative study'. In: *Information Processing & Management* 58.3 (2021), p. 102481. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2020.102481.

[325] Alexey Dosovitskiy et al. 'An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale'. In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=YicbFdNTTy.

[326] Juan Manuel Durán and Karin Rolanda Jongsma. 'Who is afraid of black box algorithms? On the epistemological and ethical basis of trust in medical AI'. In: *Journal of Medical Ethics* 47.5 (2021). Publisher: Institute of Medical Eth-

ics eprint: https://jme.bmj.com/content/47/5/329.full.pdf, pp. 329–335. ISSN: 0306-6800. DOI: 10.1136/medethics-2020-106820.

[327] Feras Al-Hawari and Hala Barham. 'A machine learning based help desk system for IT service management'. In: *Journal of King Saud University - Computer and Information Sciences* 33.6 (2021), pp. 702–718. ISSN: 1319-1578. DOI: https://doi.org/10.1016/j.jksuci.2019.04.001.

[328] Dipankar Kundu, Rajat Kumar Pal and Deba Prasad Mandal. 'Topic sensitive hybrid expertise retrieval system in community question answering services'. In: *Knowledge-Based Systems* 211.106535 (2021). ISSN: 0950-7051. DOI: 10.1016/j.knosys.2020.106535.

[329] Shayne Longpre, Yi Lu and Joachim Daiber. 'MKQA: A Linguistically Diverse Benchmark for Multilingual Open Domain Question Answering'. In: *Transactions of the Association for Computational Linguistics* 9 (2021). Place: Cambridge, MA Publisher: MIT Press, pp. 1389–1406. DOI: 10.1162/tacl_a_00433.

[330] Jie Meng, Yan Li, Chen Liu, Yaru Dong, Zishuo Wang and Yao Zhang. 'Classification of Customer Service Tickets in Power System Based on Character and Word Level Semantic Understanding'. In: *2021 China International Conference on Electricity Distribution (CICED)*. 2021, pp. 1062–1066. DOI: 10.1109/CICED50259.2021.9556759.

[331] Rida Miraj and Masaki Aono. 'Combining BERT and Multiple Embedding Methods with the Deep Neural Network for Humor Detection'. In: *Knowledge Science, Engineering and Management*. Ed. by Han Qiu, Cheng Zhang, Zongming Fei, Meikang Qiu and Sun-Yuan Kung. Cham: Springer International Publishing, 2021, pp. 53–61. ISBN: 978-3-030-82153-1.

[332] Meike Nauta, Annemarie Jutte, Jesper Provoost and Christin Seifert. 'This Looks Like That, Because ... Explaining Prototypes for Interpretable Image Recognition'. In: *Machine Learning and Principles and Practice of Knowledge Discovery in Databases*. Springer International Publishing, 2021, pp. 441–456.

[333] Q. Perez, P.-A. Jean, C. Urtado and S. Vauttier. 'Bug or not bug? That is the question'. In: *2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC)*. Journal Abbreviation: IEEE International Conference on Program Comprehension. 2021, pp. 47–58. DOI: 10.1109/ICPC52881.2021.00014.

[334] Marco Pota, Mirko Ventura, Rosario Catelli and Massimo Esposito. 'An Effective BERT-Based Pipeline for Twitter Sentiment Analysis: A Case Study in Italian'. In: *Sensors* 21.1 (2021). ISSN: 1424-8220. DOI: 10.3390/s21010133.

[335] Lorenzo Ricciardi Celsi, Andrea Caliciotti, Matteo D'Onorio, Eugenio Scocchi, Nour Alhuda Sulieman and Massimo Villari. 'On Predicting Ticket Reopening for Improving Customer Service in 5G Fiber Optic Networks'. In: *Future Internet* 13.10 (2021). ISSN: 1999-5903. URL: https://www.mdpi.com/1999-5903/13/10/259.

[336] Peyman Rostami and Mahmood Neshati. 'Intern retrieval from community question answering websites: A new variation of expert finding problem'. In: *Expert Systems with Applications* 181.115044 (2021). ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2021.115044.

[337] Vivien Sainte Fare Garnot and Loic Landrieu. 'Leveraging Class Hierarchies with Metric-Guided Prototype Learning'. In: *32th British Machine Vision Conference*. Journal Abbreviation: arXiv. Online: BMVA Press, 2021. URL: https://arxiv.org/abs/2007.03047.

[338] Mattia Samory. *The 'Call me sexist but' Dataset (CMSB)*. 2021. DOI: 10.7802/2251.

[339] Anders Søgaard. *Explainable Natural Language Processing*. Synthesis Lectures on Human Language Technologies. ISSN: 1947-4040, 1947-4059. Cham: Springer International Publishing, 2021. ISBN: 978-3-031-02180-0. DOI: 10.1007/978-3-031-02180-0.

[340] D.-T. Tolciu, C. Săcărea and C. Matei. 'Analysis of patterns and similarities in service tickets using natural language processing'. In: *Journal of Communications Software and Systems* 17.1 (2021). Publisher: Croatian Communications and Information Society, pp. 29–35. ISSN: 18456421. DOI: 10.24138/JCOMSS.V17I1.1024.

[341] Xuepeng Wang, Li Zhao, Bing Liu, Tao Chen, Feng Zhang and Di Wang. 'Concept-Based Label Embedding via Dynamic Routing for Hierarchical Text Classification'. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Ed. by Chengqing Zong, Fei Xia, Wenjie Li and Roberto Navigli. Association for Computational Linguistics, 2021, pp. 5010–5019. DOI: 10.18653/v1/2021.acl-long.388.

[342] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang and Philip S. Yu. 'A Comprehensive Survey on Graph Neural Networks'. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.1 (2021), pp. 4–24. DOI: 10.1109/TNNLS.2020.2978386.

[343] Libo YANG. 'Fuzzy Output Support Vector Machine Based Incident Ticket Classification'. In: *IEICE Transactions on Information and Systems* E104.D.1 (2021), pp. 146–151. DOI: 10.1587/transinf.2020EDP7044.

[344] Yipeng Yu, Zixun Sun, Chi Sun and Wenqiang Liu. 'Hierarchical Multilabel Text Classification via Multitask Learning'. In: *33rd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2021, Washington, DC, USA, November 1-3, 2021*. IEEE, 2021, pp. 1138–1143. DOI: 10.1109/ICTAI52525.2021.00180.

[345] Yu Zhang, Zhihong Shen, Yuxiao Dong, Kuansan Wang and Jiawei Han. 'MATCH: Metadata-Aware Text Classification in A Large Hierarchy'. In: *Proceedings of the Web Conference 2021*. WWW '21. event-place: Ljubljana, Slovenia. New York, NY, USA: Association for Computing Machinery, 2021, pp. 3246–3257. ISBN: 978-1-4503-8312-7. DOI: 10.1145/3442381.3449979.

[346] Paolo Zicari, Gianluigi Folino, Massimo Guarascio and Luigi Pontieri. 'Discovering Accurate Deep Learning Based Predictive Models for Automatic Customer Support Ticket Classification'. In: *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. SAC '21. event-place: Virtual Event, Republic of Korea. New York, NY, USA: Association for Computing Machinery, 2021, pp. 1098–1101. ISBN: 978-1-4503-8104-8. DOI: 10.1145/3412841.3442109.

[347] Romal Thoppilan et al. *LaMDA: Language Models for Dialog Applications*. 10th Feb. 2022. DOI: 10.48550/arXiv.2201.08239.

[348] Carlos Zednik and Hannes Boelsen. 'Scientific Exploration and Explainable Artificial Intelligence'. In: *Minds and Machines* 32.1 (1st Mar. 2022), pp. 219–239. ISSN: 1572-8641. DOI: 10.1007/s11023-021-09583-6.

[349] Linting Xue et al. 'ByT5: Towards a Token-Free Future with Pre-trained Byte-to-Byte Models'. In: *Transactions of the Association for Computational Linguistics* 10 (Mar. 2022), pp. 291–306. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00461.

[350] Qian Li et al. 'A Survey on Text Classification: From Traditional to Deep Learning'. In: *ACM Trans. Intell. Syst. Technol.* 13.2 (8th Apr. 2022), 31:1–31:41. ISSN: 2157-6904. DOI: 10.1145/3495162.

[351] Qingzi Vera Liao and Varshney Kush R. *Human-Centered Explainable AI (XAI): From Algorithms to User Experiences*. Apr. 2022. DOI: 10.48550/arXiv.2110.10790.

[352] Adrien Bibal, Rémi Cardon, David Alfter, Rodrigo Wilkens, Xiaoou Wang, Thomas François and Patrick Watrin. 'Is Attention Explanation? An Introduction to the Debate'. In: *Proceedings of the 60th Annual Meeting of the ACL (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 3889–3900. DOI: 10.18653/v1/2022.acl-long.269.

[353] Aaron Chan et al. 'UNIREX: A Unified Learning Framework for Language Model Rationale Extraction'. In: *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*. BigScience 2022. Ed. by Angela Fan, Suzana Ilic, Thomas Wolf and Matthias Gallé. virtual+Dublin: Association for Computational Linguistics, May 2022, pp. 51–67. DOI: 10.18653/v1/2022.bigscience-1.5.

[354] Chun Sik Chan, Huanqi Kong and Liang Guanqing. 'A Comparative Study of Faithfulness Metrics for Model Interpretability Methods'. In: *Proceedings of the 60th Annual Meeting of the ACL (Volume 1: Long Papers)*. Dublin, Ireland: Assoc. for Computational Linguistics, May 2022, pp. 5029–5038. DOI: 10.18653/v1/2022.acl-long.345.

[355] Sayan Ghosh and Shashank Srivastava. 'ePiC: Employing Proverbs in Context as a Benchmark for Abstract Language Understanding'. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Smaranda Muresan, Preslav Nakov and Aline Villavicencio. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 3989–4004. DOI: 10.18653/v1/2022.acl-long.276.

[356] Dipesh Kumar and Avijit Thawani. 'BPE beyond Word Boundary: How NOT to use Multi Word Expressions in Neural Machine Translation'. In: *Proceedings of the Third Workshop on Insights from Negative Results in NLP*. insights 2022. Ed. by Shabnam Tafreshi, João Sedoc, Anna Rogers, Aleksandr Drozd, Anna Rumshisky and Arjun Akula. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 172–179. DOI: 10.18653/v1/2022.insights-1.24.

[357] Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke and Jose Camacho-collados. 'TimeLMs: Diachronic Language Models from Twitter'. In: *Proceedings of the 60th Annual Meeting of the ACL: System Demonstrations*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 251–260. DOI: 10.18653/v1/2022.acl-demo.25.

[358] Zihan Wang, Peiyi Wang, Lianzhe Huang, Xin Sun and Houfeng Wang. 'Incorporating Hierarchy into Text Encoder: a Contrastive Learning Approach for Hierarchical Text Classification'. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 7109–7119. DOI: 10.18653/v1/2022.acl-long.491.

[359] Bodhisattwa Prasad Majumder, Oana Camburu, Thomas Lukasiewicz and Julian Mcauley. 'Knowledge-Grounded Self-Rationalization via Extractive and Natural Language Explanations'. In: *Proceedings of the 39th International Conference on Machine Learning*. International Conference on Machine Learning. ISSN: 2640-3498. PMLR, 28th June 2022, pp. 14786–14801. URL:

https://proceedings.mlr.press/v162/majumder22a.html (visited on 10/07/2024).

[360] Stack Exchange Inc. *Stack Exchange Data Dump*. June 2022. URL: https://archive.org/details/stackexchange.

[361] Alessandro Zangari, Matteo Marcuzzo, Andrea Albarelli and Andrea Gasparetto. *It/Fr/En-Wiki-100 datasets*. 6th July 2022. URL: https://zenodo.org/records/7244893 (visited on 16/06/2025).

[362] Astrid Bertrand, Rafik Belloum, James R. Eagan and Winston Maxwell. 'How Cognitive Biases Affect XAI-assisted Decision-making: A Systematic Review'. In: *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*. AIES '22. New York, NY, USA: Association for Computing Machinery, 27th July 2022, pp. 78–91. ISBN: 978-1-4503-9247-1. DOI: 10.1145/3514094.3534164.

[363] Anirudh Mittal, Yufei Tian and Nanyun Peng. 'AmbiPun: Generating Humorous Puns with Ambiguous Context'. In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Marine Carpuat, Marie-Catherine de Marneffe and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 1053–1062. DOI: 10.18653/v1/2022.naacl-main.77.

[364] Sarah Wiegreffe, Jack Hessel, Swabha Swayamdipta, Mark Riedl and Yejin Choi. 'Reframing Human-AI Collaboration for Generating Free-Text Explanations'. In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. NAACL-HLT 2022. Ed. by Marine Carpuat, Marie-Catherine de Marneffe and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 632–658. DOI: 10.18653/v1/2022.naacl-main.47.

[365] N Saranya, K Srinivasan and S K Pravin Kumar. 'Banana ripeness stage identification: a deep learning approach'. In: *J. Ambient Intell. Humaniz. Comput.* 13.8 (Aug. 2022), pp. 4033–4039. DOI: 10.1007/s12652-021-03267-w.

[366] Kacper Sokol and Peter Flach. *Explainability Is in the Mind of the Beholder: Establishing the Foundations of Explainable Artificial Intelligence*. 9th Sept. 2022. DOI: 10.48550/arXiv.2112.14466.

[367] Walid S. Saba. *New Research Vindicates Fodor and Pylyshyn: No Explainable AI Without 'Structured Semantics' – Communications of the ACM*. Communications of the ACM. 14th Sept. 2022. URL: https://cacm.acm.org/blogcacm/new-research-vindicates-fodor-and-pylyshyn-no-explainable-ai-without-structured-semantics/ (visited on 21/07/2025).

[368] Enayat Rajabi and Kobra Etminani. 'Knowledge-graph-based explainable AI: A systematic review'. In: *Journal of Information Science* (24th Sept. 2022). Publisher: SAGE Publications Ltd, p. 01655515221112844. ISSN: 0165-5515. DOI: 10.1177/01655515221112844.

[369] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan and Mubarak Shah. 'Transformers in Vision: A Survey'. In: *ACM Comput. Surv.* 54.10 (Sept. 2022). Place: New York, NY, USA Publisher: Association for Computing Machinery. ISSN: 0360-0300. DOI: 10.1145/3505244.

[370] Zhuosheng Zhang, Aston Zhang, Mu Li and Alex Smola. *Automatic Chain of Thought Prompting in Large Language Models*. 7th Oct. 2022. DOI: 10.48550/arXiv.2210.03493.

[371] Dimosthenis Antypas, Asahi Ushio, Jose Camacho-Collados, Vitor Silva, Leonardo Neves and Francesco Barbieri. 'Twitter Topic Classification'. In: *Proceedings of the 29th International Conference on Computational Linguistics*. Ed. by Nicoletta Calzolari et al. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 3386–3400.

[372] Edoardo Mosca, Ferenc Szigeti, Stella Tragianni, Daniel Gallagher and Georg Groh. 'SHAP-Based Explanation Methods: A Review for NLP Interpretability'. In: *Proceedings of the 29th International Conference on Computational Linguistics*. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 4593–4603.

[373] Hyung Won Chung et al. *Scaling Instruction-Finetuned Language Models*. 6th Dec. 2022. DOI: 10.48550/arXiv.2210.11416.

[374] Alessandro Zangari, Matteo Marcuzzo, Matteo Rizzo, Andrea Albarelli and Andrea Gasparetto. *Hierarchical Text Classification corpora*. 14th Dec. 2022. URL: https://zenodo.org/records/7319519 (visited on 16/06/2025).

[375] Andreas Madsen et al. 'Post-hoc Interpretability for Neural NLP: A Survey'. In: *ACM Computing Surveys* 55.8 (23rd Dec. 2022). Publisher: Association for Computing Machinery, pp. 1–42. DOI: 10.1145/3546577.

[376] Jiao Sun et al. 'Context-Situated Pun Generation'. In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 4635–4648. DOI: 10.18653/v1/2022.emnlp-main.306.

[377] Jiao Sun et al. 'ExPUNations: Augmenting Puns with Keywords and Explanations'. In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2022. Ed. by Yoav Goldberg, Zornitsa Kozareva and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for

Computational Linguistics, Dec. 2022, pp. 4590–4605. DOI: 10.18653/v1/2022.emnlp-main.304.

[378] Yufei Tian, Divyanshu Sheth and Nanyun Peng. 'A Unified Framework for Pun Generation with Humor Principles'. In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. Ed. by Yoav Goldberg, Zornitsa Kozareva and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 3253–3261. DOI: 10.18653/v1/2022.findings-emnlp.237.

[379] Syed S. Ali Zaidi, Muhammad Moazam Fraz, Muhammad Shahzad and Sharifullah Khan. 'A multiapproach generalized framework for automated solution suggestion of support tickets'. In: *International Journal of Intelligent Systems* 37.6 (2022), pp. 3654–3681. DOI: 10.1002/int.22701.

[380] Arian Askari, Suzan Verberne and Gabriella Pasi. 'Expert Finding in Legal Community Question Answering'. In: *Advances in Information Retrieval*. Ed. by Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg and Vinay Setty. Cham: Springer International Publishing, 2022, pp. 22–30. ISBN: 978-3-030-99739-7.

[381] Aakanksha Chowdhery et al. 'PaLM: Scaling Language Modeling with Pathways'. In: *arXiv* (2022). DOI: 10.48550/ARXIV.2204.02311.

[382] Zohreh Fallahnejad and Hamid Beigy. 'Attention-based skill translation models for expert finding'. In: *Expert Systems with Applications* 193 (2022), p. 116433. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2021.116433.

[383] Simon Fuchs, Clemens Drieschner and Holger Wittges. 'Improving Support Ticket Systems Using Machine Learning: A Literature Review'. In: *Proceedings of the 55th Hawaii International Conference on System Sciences*. Honolulu, HI 96822: ScholarSpace, 2022, pp. 1893–1902. URL: https://hdl.handle.net/10125/79570.

[384] Clarissa Faye G. Gamboa et al. 'Further Enhancement of KNN Algorithm Based on Clustering Applied to IT Support Ticket Routing'. In: *2022 3rd International Conference on Computing, Networks and Internet of Things (CNIOT)*. 2022, pp. 186–190. DOI: 10.1109/CNIOT55862.2022.00040.

[385] Takeshi Kojima, Shixiang (Shane) Gu, Machel Reid, Yutaka Matsuo and Yusuke Iwasawa. 'Large Language Models are Zero-Shot Reasoners'. In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho and A. Oh. Vol. 35. Curran Associates, Inc., 2022, pp. 22199–22213. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/8bb0d291acd4acf06ef112099c16f326-Paper-Conference.pdf.

[386] Yue Liu, Weize Tang, Zitu Liu, Lin Ding and Aihua Tang. 'High-quality domain expert finding method in CQA based on multi-granularity semantic analysis and interest drift'. In: *Information Sciences* 596 (2022), pp. 395–413. ISSN: 0020-0255. DOI: 10.1016/j.ins.2022.02.039.

[387] Ian E. Nielsen, Dimah Dera, Ghulam Rasool, Ravi P. Ramachandran and Nidhal Carla Bouaynaya. 'Robust Explainability: A tutorial on gradient-based attribution methods for deep neural networks'. In: *IEEE Signal Processing Magazine* 39.4 (2022), pp. 73–84. DOI: 10.1109/MSP.2022.3142719.

[388] Qiyao Peng, Hongtao Liu, Yinghui Wang, Hongyan Xu, Pengfei Jiao, Minglai Shao and Wenjun Wang. 'Towards a Multi-View Attentive Matching for Personalized Expert Finding'. In: *Proceedings of the ACM Web Conference 2022*. WWW '22. event-place: Virtual Event, Lyon, France. New York, NY, USA: Association for Computing Machinery, 2022, pp. 2131–2140. ISBN: 978-1-4503-9096-5. DOI: 10.1145/3485447.3512086.

[389] Jason Wei et al. 'Chain-of-thought prompting elicits reasoning in large language models'. In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*. NIPS '22. event-place: New Orleans, LA, USA. Red Hook, NY, USA: Curran Associates Inc., 2022. ISBN: 978-1-7138-7108-8.

[390] Chao Yu, Yi Shen and Yue Mao. 'Constrained Sequence-to-Tree Generation for Hierarchical Text Classification'. In: *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '22. event-place: Madrid, Spain. New York, NY, USA: Association for Computing Machinery, 2022, pp. 1865–1869. ISBN: 978-1-4503-8732-3. DOI: 10.1145/3477495.3531765.

[391] Jun Yuan, Jesse Vig and Nazneen Rajani. 'ISEA: An Interactive Pipeline for Semantic Error Analysis of NLP Models'. In: *27th International Conference on Intelligent User Interfaces*. IUI '22. event-place: Helsinki, Finland. New York, NY, USA: Association for Computing Machinery, 2022, pp. 878–888. ISBN: 978-1-4503-9144-3. DOI: 10.1145/3490099.3511146.

[392] Xinyi Zhang, Jiahao Xu, Charlie Soh and Lihui Chen. 'LA-HCN: Label-based Attention for Hierarchical Multi-label Text Classification Neural Network'. In: *Expert Syst. Appl.* 187 (2022), p. 115922. DOI: 10.1016/j.eswa.2021.115922.

[393] Erik Cambria, Lorenzo Malandri, Fabio Mercorio, Mario Mezzanzanica and Navid Nobani. 'A survey on XAI and natural language explanations'. In: *Information Processing & Management* 60.1 (1st Jan. 2023), p. 103111. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2022.103111.

[394] Rishabh Misra and Prahal Arora. 'Sarcasm detection using news headlines dataset'. In: *AI Open* 4 (1st Jan. 2023), pp. 13–18. ISSN: 2666-6510. DOI: 10.1016/j.aiopen.2023.01.001.

[395] Jerry Wei et al. *Larger language models do in-context learning differently*. 8th Mar. 2023. DOI: 10.48550/arXiv.2303.03846.

[396] Jingun Kwon, Hidetaka Kamigaito, Young-In Song and Manabu Okumura. 'Hierarchical Label Generation for Text Classification'. In: *Findings of the Association for Computational Linguistics: EACL 2023*. Ed. by Andreas Vlachos and Isabelle Augenstein. Dubrovnik, Croatia: Association for Computational Linguistics, May 2023, pp. 625–632. DOI: 10.18653/v1/2023.findings-eacl.46.

[397] Shaked Yehezkel and Yuval Pinter. 'Incorporating Context into Subword Vocabularies'. In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. EACL 2023. Ed. by Andreas Vlachos and Isabelle Augenstein. Dubrovnik, Croatia: Association for Computational Linguistics, May 2023, pp. 623–635. DOI: 10.18653/v1/2023.eacl-main.45.

[398] Dan Ley, Leonard Tang, Matthew Nazari, Hongjin Lin, Suraj Srinivas and Himabindu Lakkaraju. *Consistent Explanations in the Face of Model Indeterminacy via Ensembling*. 13th June 2023. DOI: 10.48550/arXiv.2306.06193.

[399] Tamera Lanham et al. *Measuring Faithfulness in Chain-of-Thought Reasoning*. 17th July 2023. DOI: 10.48550/arXiv.2307.13702.

[400] Ke Ji, Yixin Lian, Jingsheng Gao and Baoyuan Wang. 'Hierarchical Verbalizer for Few-Shot Hierarchical Text Classification'. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 2918–2933. DOI: 10.18653/v1/2023.acl-long.164.

[401] Thanh-Tung Nguyen, Viktor Schlegel, Abhinav Ramesh Kashyap and Stefan Winkler. 'A Two-Stage Decoder for Efficient ICD Coding'. In: *Findings of the Association for Computational Linguistics: ACL 2023*. Ed. by Anna Rogers, Jordan Boyd-Graber and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 4658–4665. DOI: 10.18653/v1/2023.findings-acl.285.

[402] Matteo Rizzo, Alberto Veneri, Andrea Albarelli, Claudio Lucchese, Marco Nobile and Cristina Conati. 'A Theoretical Framework for AI Models Explainability with Application in Biomedicine'. In: *2023 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. Aug. 2023, pp. 1–9. DOI: 10.1109/CIBCB56990.2023.10264877.

[403] Sai Gurrapu, Ajay Kulkarni, Lifu Huang, Ismini Lourentzou and Feras A. Batarseh. 'Rationalization for explainable NLP: a survey'. In: *Frontiers in Artificial Intelligence* 6 (25th Sept. 2023). Publisher: Frontiers. ISSN: 2624-8212. DOI: 10.3389/frai.2023.1225093.

[404] Albert Q. Jiang et al. *Mistral 7B*. 10th Oct. 2023. DOI: 10.48550/arXiv.2310.06825.

[405] Masayasu Muraoka, Bishwaranjan Bhattacharjee, Michele Merler, Graeme Blackwood, Yulong Li and Yang Zhao. 'Cross-Lingual Transfer of Large Language Model by Visually-Derived Supervision Toward Low-Resource Languages'. In: *Proceedings of the 31st ACM International Conference on Multimedia*. MM '23. New York, NY, USA: Association for Computing Machinery, 27th Oct. 2023, pp. 3637–3646. ISBN: 979-8-4007-0108-5. DOI: 10.1145/3581783.3611992.

[406] Rattana Pukdee, Dylan Sam, J. Zico Kolter, Maria-Florina Balcan and Pradeep Ravikumar. 'Learning with explanation constraints'. In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS '23. Red Hook, NY, USA: Curran Associates Inc., 10th Dec. 2023, pp. 49883–49926. (Visited on 27/09/2025).

[407] Brian Barr, Noah Fatsi, Leif Hancox-Li, Peter Richter and Daniel Proano. 'The Disagreement Problem in Faithfulness Metrics'. In: NeurIPS Workshop: XAI in Action: Past, Present, and Future Applications. NeurIPS XAIA 2023. New Orleans, Louisiana, United States: Openreview, 15th Dec. 2023. URL: https://openreview.net/forum?id=KPtW2SU0my (visited on 09/06/2025).

[408] Solon Barocas, Moritz Hardt and Arvind Narayanan. *Fairness and Machine Learning: Limitations and Opportunities*. Cambridge, MA, USA: MIT Press, 19th Dec. 2023. 340 pp. ISBN: 978-0-262-04861-3. URL: https://mitpress.mit.edu/9780262048613/fairness-and-machine-learning/ (visited on 22/09/2025).

[409] Alisa Liu et al. 'We're Afraid Language Models Aren't Modeling Ambiguity'. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Houda Bouamor, Juan Pino and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 790–807. DOI: 10.18653/v1/2023.emnlp-main.51.

[410] Guanqi Chen, Lei Yang, Guanhua Chen and Jia Pan. 'Evaluating Explanation Methods for Vision-and-Language Navigation'. In: *ECAI 2023*. IOS Press, 2023, pp. 389–396. DOI: 10.3233/FAIA230295.

[411] Yuyan Chen, Zhixu Li, Jiaqing Liang, Yanghua Xiao, Bang Liu and Yunwen Chen. 'Can Pre-trained Language Models Understand Chinese Humor?' In: *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. WSDM '23. event-place: Singapore, Singapore. New York,

NY, USA: Association for Computing Machinery, 2023, pp. 465–480. ISBN: 978-1-4503-9407-9. DOI: 10.1145/3539597.3570431.

[412] Liana Ermakova, Anne-Gwenn Bosser, Adam Jatowt and Tristan Miller. 'The JOKER Corpus: English-French Parallel Data for Multilingual Wordplay Recognition'. In: *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '23. event-place: Taipei, Taiwan. New York, NY, USA: Association for Computing Machinery, 2023, pp. 2796–2806. ISBN: 978-1-4503-9408-6. DOI: 10.1145/3539618.3591885.

[413] Liana Ermakova, Tristan Miller, Anne-Gwenn Bosser, Victor Manuel Palma Preciado, Grigori Sidorov and Adam Jatowt. 'Overview of JOKER 2023 Automatic Wordplay Analysis Task 2 – Pun Location and Interpretation'. In: *Working Notes of the Conference and Labs of the Evaluation Forum (CLEF 2023)*. Ed. by Mohammad Aliannejadi, Michalis Vlachos, Guglielmo Faggioli and Nicola Ferro. Vol. 3497. ISSN: 1613-0073. Thessaloniki, Greece: CEUR workshop proceedings, 2023, pp. 1804–1817.

[414] Timo Freiesleben and Gunnar König. 'Dear XAI Community, We Need to Talk!' In: *Explainable Artificial Intelligence*. Ed. by Luca Longo. Cham: Springer Nature Switzerland, 2023, pp. 48–65. ISBN: 978-3-031-44064-9. DOI: 10.1007/978-3-031-44064-9_3.

[415] José Ángel González, Encarna Segarra, Fernando García-Granada, Emilio Sanchis and Lluís-F. Hurtado. 'Attentional Extractive Summarization'. In: *Applied Sciences* 13.3 (2023). ISSN: 2076-3417. DOI: 10.3390/app13031458.

[416] Pavel Ircing and Jan Švec. 'Will XAI Provide Real Explanation or Just a Plausible Rationalization?' In: *Pattern Recognition*. Ed. by Huimin Lu, Michael Blumenstein, Sung-Bae Cho, Cheng-Lin Liu, Yasushi Yagi and Tohru Kamiya. Cham: Springer Nature Switzerland, 2023, pp. 358–368. ISBN: 978-3-031-47665-5. DOI: 10.1007/978-3-031-47665-5_29.

[417] OpenAI et al. *GPT-4 Technical Report*. 2023. URL: https://arxiv.org/abs/2303.08774.

[418] Walid S. Saba. 'Stochastic LLMs do not Understand Language: Towards Symbolic, Explainable and Ontologically Based LLMs'. In: *Conceptual Modeling*. Ed. by João Paulo A. Almeida, José Borbinha, Giancarlo Guizzardi, Sebastian Link and Jelena Zdravkovic. Cham: Springer Nature Switzerland, 2023, pp. 3–19. ISBN: 978-3-031-47262-6. DOI: 10.1007/978-3-031-47262-6_1.

[419] Miles Turpin, Julian Michael, Ethan Perez and Samuel Bowman. 'Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting'. In: *Advances in neural information processing systems*. Ed. by A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt and S. Levine. Vol. 36. Curran Associates, Inc., 2023, pp. 74952–74965. URL:

https://proceedings.neurips.cc/paper_files/paper/2023/file/ed3fea9033a80fea1376299fa7863f4a-Paper-Conference.pdf.

[420] BigScience Workshop, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić and Daniel Hesslow et al. *BLOOM: A 176B-Parameter Open-Access Multilingual Language Model*. eprint: 2211.05100. 2023. URL: https://arxiv.org/abs/2211.05100.

[421] Olesja Lammert, Birte Richter, Christian Schütze, Kirsten Thommes and Britta Wrede. 'Humans in XAI: increased reliance in decision-making under uncertainty by using explanation strategies'. In: *Frontiers in Behavioral Economics* 3 (8th Mar. 2024). Publisher: Frontiers. ISSN: 2813-5296. DOI: 10.3389/frbhe.2024.1377075.

[422] Haiyan Zhao et al. 'Explainability for Large Language Models: A Survey'. In: *ACM Transactions on Intelligent Systems and Technology* 15.2 (30th Apr. 2024), pp. 1–38. ISSN: 2157-6904, 2157-6912. DOI: 10.1145/3639372.

[423] Gabriel Laberge, Yann Batiste Pequignot, Mario Marchand and Foutse Khomh. 'Tackling the XAI Disagreement Problem with Regional Explanations'. In: *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*. Ed. by Sanjoy Dasgupta, Stephan Mandt and Yingzhen Li. Vol. 238. Proceedings of Machine Learning Research. PMLR, 2nd May 2024, pp. 2017–2025. URL: https://proceedings.mlr.press/v238/laberge24a.html.

[424] Gordon Burtch, Dokyun Lee and Zhichen Chen. 'The consequences of generative AI for online knowledge communities'. In: *Scientific Reports* 14.1 (6th May 2024). Publisher: Nature Publishing Group, p. 10413. ISSN: 2045-2322. DOI: 10.1038/s41598-024-61221-0.

[425] Alexandra Zytek, Sara Pidò and Kalyan Veeramachaneni. *LLMs for XAI: Future Directions for Explaining Explanations*. 9th May 2024. DOI: 10.48550/arXiv.2405.06064.

[426] Elize Herrewijnen, Dong Nguyen, Floris Bex and Kees van Deemter. 'Human-annotated rationales and explainable text classification: a survey'. In: *Frontiers in Artificial Intelligence* 7 (May 2024). ISSN: 2624-8212. DOI: 10.3389/frai.2024.1260952.

[427] Joel Rorseth, Parke Godfrey, Lukasz Golab, Divesh Srivastava and Jaroslaw Szlichta. 'Towards Explainability in Retrieval-Augmented LLMs'. In: *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. ISSN: 2375-026X. May 2024, pp. 5669–5670. DOI: 10.1109/ICDE60146.2024.00466.

[428] Piotr Mirowski, Juliette Love, Kory Mathewson and Shakir Mohamed. 'A Robot Walks into a Bar: Can Language Models Serve as Creativity SupportTools for Comedy? An Evaluation of LLMs' Humour Alignment with Comedians'. In: *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*. FAccT '24. New York, NY, USA: Association for Computing Machinery, 5th June 2024, pp. 1622–1636. ISBN: 979-8-4007-0450-5. DOI: 10.1145/3630106.3658993.

[429] European Union. 'Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence and amending Regulations (EC) No 300/2008, (EU) No 167/2013, (EU) No 168/2013, (EU) 2018/858, (EU) 2018/1139 and (EU) 2019/2144 and Directives 2014/90/EU, (EU) 2016/797 and (EU) 2020/1828 (Artificial Intelligence Act) (Text with EEA relevance)'. Legislative Body: CONSIL, EP. 13th June 2024. URL: http://data.europa.eu/eli/reg/2024/1689/oj/eng (visited on 05/08/2025).

[430] Mehdi Ali et al. 'Tokenizer Choice For LLM Training: Negligible or Crucial?' In: *Findings of the Association for Computational Linguistics: NAACL 2024*. Findings 2024. Ed. by Kevin Duh, Helena Gomez and Steven Bethard. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 3907–3924. DOI: 10.18653/v1/2024.findings-naacl.247.

[431] Viju Sudhi, Sinchana Ramakanth Bhat, Max Rudat and Roman Teucher. 'RAG-Ex: A Generic Framework for Explaining Retrieval Augmented Generation'. In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '24. New York, NY, USA: Association for Computing Machinery, 11th July 2024, pp. 2776–2780. ISBN: 979-8-4007-0431-4. DOI: 10.1145/3626772.3657660.

[432] Erick Mendez Guzman, Viktor Schlegel and Riza Batista-Navarro. 'From outputs to insights: a survey of rationalization approaches for explainable text classification'. In: *Frontiers in Artificial Intelligence* 7 (23rd July 2024). Publisher: Frontiers. ISSN: 2624-8212. DOI: 10.3389/frai.2024.1363531.

[433] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson and Yarin Gal. 'AI models collapse when trained on recursively generated data'. In: *Nature* 631.8022 (July 2024). Publisher: Nature Publishing Group, pp. 755–759. ISSN: 1476-4687. DOI: 10.1038/s41586-024-07566-y.

[434] Shree Singhi and Anupriya Kumari. *Strengthening Interpretability: An Investigative Study of Integrated Gradient Methods*. 29th Aug. 2024. DOI: 10.48550/arXiv.2409.09043.

[435] Wenxuan Wang, Zhaopeng Tu, Chang Chen, Youliang Yuan, Jen-tse Huang, Wenxiang Jiao and Michael Lyu. 'All Languages Matter: On the Multilingual Safety of LLMs'. In: *Findings of the Association for Computational Linguistics: ACL 2024*. Findings 2024. Ed. by Lun-Wei Ku, Andre Martins and Vivek

Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 5865–5877. DOI: 10.18653/v1/2024.findings-acl.349.

[436] Tong Zhang et al. 'CLAMBER: A Benchmark of Identifying and Clarifying Ambiguous Information Needs in Large Language Models'. In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 10746–10766. DOI: 10.18653/v1/2024.acl-long.578.

[437] Zhexin Zhang et al. 'SafetyBench: Evaluating the Safety of Large Language Models'. In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2024. Ed. by Lun-Wei Ku, Andre Martins and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 15537–15553. DOI: 10.18653/v1/2024.acl-long.830.

[438] Jenia Kim, Henry Maathuis and Danielle Sent. 'Human-centered evaluation of explainable AI applications: a systematic review'. In: *Frontiers in Artificial Intelligence* 7 (17th Oct. 2024). Publisher: Frontiers. ISSN: 2624-8212. DOI: 10.3389/frai.2024.1456486.

[439] Andreas Madsen, Himabindu Lakkaraju, Siva Reddy and Sarath Chandar. *Interpretability Needs a New Paradigm*. 13th Nov. 2024. DOI: 10.48550/arXiv.2405.05386.

[440] Kaijie Zhu et al. 'PromptRobust: Towards Evaluating the Robustness of Large Language Models on Adversarial Prompts'. In: *Proceedings of the 1st ACM Workshop on Large AI Systems and Models with Privacy and Safety Analysis*. LAMPS '24. New York, NY, USA: Association for Computing Machinery, 19th Nov. 2024, pp. 57–68. ISBN: 979-8-4007-1209-8. DOI: 10.1145/3689217.3690621.

[441] Lars Malmqvist. *Sycophancy in Large Language Models: Causes and Mitigations*. 22nd Nov. 2024. DOI: 10.48550/arXiv.2411.15287.

[442] Joanne Boisson et al. 'How Are Metaphors Processed by Language Models? The Case of Analogies'. In: *Proceedings of the 28th Conference on Computational Natural Language Learning*. Ed. by Libby Barak and Malihe Alikhani. Miami, FL, USA: Association for Computational Linguistics, Nov. 2024, pp. 365–387. DOI: 10.18653/v1/2024.conll-1.28.

[443] Tyler A. Chang, Catherine Arnett, Zhuowen Tu and Ben Bergen. 'When Is Multilinguality a Curse? Language Modeling for 250 High- and Low-Resource Languages'. In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2024. Ed. by Yaser Al-Onaizan, Mohit Bansal and Yun-Nung Chen. Miami, Florida, USA: Association for

Computational Linguistics, Nov. 2024, pp. 4074–4096. DOI: 10.18653/v1/2024.emnlp-main.236.

[444] Miriam Horovicz and Roni Goldshmidt. 'TokenSHAP: Interpreting Large Language Models with Monte Carlo Shapley Value Estimation'. In: *Proceedings of the 1st Workshop on NLP for Science (NLP4Science)*. Ed. by Lotem Peled-Cohen, Nitay Calderon, Shir Lissak and Roi Reichart. Miami, FL, USA: Association for Computational Linguistics, Nov. 2024, pp. 1–8. DOI: 10.18653/v1/2024.nlp4science-1.1.

[445] Zhijun Xu, Siyu Yuan, Lingjie Chen and Deqing Yang. '"A good pun is its own reword": Can Large Language Models Understand Puns?' In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 11766–11782. DOI: 10.18653/v1/2024.emnlp-main.657.

[446] Ryan Rony Dsilva. 'From Sentence Embeddings to Large Language Models to Detect and Understand Wordplay'. In: *Experimental IR Meets Multilinguality, Multimodality, and Interaction*. Ed. by Lorraine Goeuriot et al. Cham: Springer Nature Switzerland, 2024, pp. 205–214. ISBN: 978-3-031-71736-9.

[447] Aaron Grattafiori et al. *The Llama 3 Herd of Models*. eprint: 2407.21783. 2024. URL: https://arxiv.org/abs/2407.21783.

[448] OpenAI et al. *GPT-4o System Card*. eprint: 2410.21276. 2024. URL: https://arxiv.org/abs/2410.21276.

[449] Marek Pawlicki, Aleksandra Pawlicka, Federica Uccello, Sebastian Szelest, Salvatore D'Antonio, Rafał Kozik and Michał Choraś. 'Evaluating the necessity of the multiple metrics for assessing explainable AI: A critical examination'. In: *Neurocomputing* 602 (2024), p. 128282. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2024.128282.

[450] David Rein et al. 'GPQA: a graduate-level google-proof Q&a benchmark'. In: *First conference on language modeling*. 2024. URL: https://openreview.net/forum?id=Ti67584b98.

[451] Qwen et al. *Qwen2.5 Technical Report*. 3rd Jan. 2025. DOI: 10.48550/arXiv.2412.15115.

[452] Kaiyu Huang et al. *A Survey on Large Language Models with Multilingualism: Recent Advances and New Frontiers*. version: 2. 7th Jan. 2025. DOI: 10.48550/arXiv.2405.10936.

[453] Libo Qin et al. 'A survey of multilingual large language models'. In: *Patterns* 6.1 (10th Jan. 2025). Publisher: Elsevier. ISSN: 2666-3899. DOI: 10.1016/j.patter.2024.101118.

[454] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd (draft). Online manuscript released January 12, 2025. Jan. 2025. URL: https://web.stanford.edu/~jurafsky/slp3/.

[455] Hazel H. Kim. 'How Ambiguous Are the Rationales for Natural Language Reasoning? A Simple Approach to Handling Rationale Uncertainty'. In: *Proceedings of the 31st International Conference on Computational Linguistics*. Ed. by Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio and Steven Schockaert. Abu Dhabi, UAE: Association for Computational Linguistics, Jan. 2025, pp. 10047–10053. URL: https://aclanthology.org/2025.coling-main.671/.

[456] Tobias Leemann, Alina Fastowski, Felix Pfeiffer and Gjergji Kasneci. 'Attention mechanisms don't learn additive models: Rethinking feature importance for transformers'. In: *Transactions on Machine Learning Research* (Jan. 2025). ISSN: 2835-8856. URL: https://openreview.net/forum?id=yawWz4qWkF.

[457] Menan Velayuthan and Kengatharaiyer Sarveswaran. 'Egalitarian Language Representation in Language Models: It All Begins with Tokenizers'. In: *Proceedings of the 31st International Conference on Computational Linguistics*. COLING 2025. Ed. by Owen Rambow, Leo Wanner, Marianna Apidianaki, Hend Al-Khalifa, Barbara Di Eugenio and Steven Schockaert. Abu Dhabi, UAE: Association for Computational Linguistics, Jan. 2025, pp. 5987–5996. URL: https://aclanthology.org/2025.coling-main.400/ (visited on 29/07/2025).

[458] Edmund Mills, Shiye Su, Stuart Russell and Scott Emmons. *ALMANACS: A Simulatability Benchmark for Language Model Explainability*. 2nd Feb. 2025. DOI: 10.48550/arXiv.2312.12747.

[459] He Zhu, Jinxiang Xia, Ruomei Liu and Bowen Deng. 'SPIRIT: Structural Entropy Guided Prefix Tuning for Hierarchical Text Classification'. In: *Entropy* 27.2 (Feb. 2025). Publisher: Multidisciplinary Digital Publishing Institute, p. 128. ISSN: 1099-4300. DOI: 10.3390/e27020128.

[460] Wayne Xin Zhao et al. *A Survey of Large Language Models*. 11th Mar. 2025. DOI: 10.48550/arXiv.2303.18223.

[461] Craig W. Schmidt, Varshini Reddy, Chris Tanner and Yuval Pinter. *Boundless Byte Pair Encoding: Breaking the Pre-tokenization Barrier*. 31st Mar. 2025. DOI: 10.48550/arXiv.2504.00178.

[462] Yunyi Zhang, Ruozhen Yang, Xueqiang Xu, Rui Li, Jinfeng Xiao, Jiaming Shen and Jiawei Han. 'TELEClass: Taxonomy Enrichment and LLM-Enhanced Hierarchical Text Classification with Minimal Supervision'. In: *Proceedings of the ACM on Web Conference 2025*. WWW '25. New York, NY,

USA: Association for Computing Machinery, 22nd Apr. 2025, pp. 2032–2042. ISBN: 979-8-4007-1274-6. DOI: 10.1145/3696410.3714940.

[463] Luping Wang, Sheng Chen, Linnan Jiang, Shu Pan, Runze Cai, Sen Yang and Fei Yang. 'Parameter-efficient fine-tuning in large language models: a survey of methodologies'. In: *Artificial Intelligence Review* 58.8 (3rd May 2025), p. 227. ISSN: 1573-7462. DOI: 10.1007/s10462-025-11236-4.

[464] Mrinank Sharma et al. *Towards Understanding Sycophancy in Language Models*. 10th May 2025. DOI: 10.48550/arXiv.2310.13548.

[465] Punam Bedi, Anjali Thukral and Shivani Dhiman. 'XLR-KGDD: leveraging LLM and RAG for knowledge graph-based explainable disease diagnosis using multimodal clinical information'. In: *Knowledge and Information Systems* (13th May 2025). ISSN: 0219-3116. DOI: 10.1007/s10115-025-02465-8.

[466] Myra Cheng, Sunny Yu, Cinoo Lee, Pranav Khadpe, Lujain Ibrahim and Dan Jurafsky. *Social Sycophancy: A Broader Understanding of LLM Sycophancy*. 20th May 2025. DOI: 10.48550/arXiv.2505.13995.

[467] Weina Jin, Xiaoxiao Li and Ghassan Hamarneh. *Why is plausibility surprisingly problematic as an XAI criterion?* 30th May 2025. DOI: 10.48550/arXiv.2303.17707.

[468] Théophile Blard. *french-sentiment-analysis-with-bert*. 12th June 2025. URL: https://github.com/TheophileBlard/french-sentiment-analysis-with-bert (visited on 07/08/2025).

[469] Kexin Quan, Pavithra Ramakrishnan and Jessie Chin. 'Can AI Take a Joke—Or Make One? A Study of Humor Generation and Recognition in LLMs'. In: *Proceedings of the 2025 Conference on Creativity and Cognition*. C&C '25. New York, NY, USA: Association for Computing Machinery, 22nd June 2025, pp. 431–437. ISBN: 979-8-4007-1289-0. DOI: 10.1145/3698061.3734388.

[470] Luel Hagos Beyene, Vivek Verma, Min Ma, Jesujoba O. Alabi, Fabian David Schmidt, Joyce Nakatumba-Nabende and David Ifeoluwa Adelani. *mSTEB: Massively Multilingual Evaluation of LLMs on Speech and Text Tasks*. version: 2. 25th June 2025. DOI: 10.48550/arXiv.2506.08400.

[471] Benjamin Warner et al. 'Smarter, Better, Faster, Longer: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Context Finetuning and Inference'. In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2025. Ed. by Wanxiang Che, Joyce Nabende, Ekaterina Shutova and Mohammad Taher Pilehvar. Vienna, Austria: Association for Computational Linguistics, July 2025, pp. 2526–2547. ISBN: 979-8-89176-251-0. URL: https://aclanthology.org/2025.acl-long.127/ (visited on 30/07/2025).

[472] Chantal Shaib, Vinith M. Suriyakumar, Levent Sagun, Byron C. Wallace and Marzyeh Ghassemi. *Learning the Wrong Lessons: Syntactic-Domain Spurious Correlations in Language Models*. 26th Sept. 2025. DOI: 10.48550/arXiv.2509.21155.

[473] Lucas Charpentier et al. 'Findings of the Third BabyLM Challenge: Accelerating Language Modeling Research with Cognitively Plausible Data'. In: *Proceedings of the First BabyLM Workshop*. BabyLM 2025. Ed. by Lucas Charpentier et al. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 399–420. DOI: 10.18653/v1/2025.babylm-main.28.

[474] Martin Tutek, Fateme Hashemi Chaleshtori, Ana Marasovic and Yonatan Belinkov. 'Measuring Chain of Thought Faithfulness by Unlearning Reasoning Steps'. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. EMNLP 2025. Ed. by Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose and Violet Peng. Suzhou, China: Association for Computational Linguistics, Nov. 2025, pp. 9946–9971. ISBN: 979-8-89176-332-6. DOI: 10.18653/v1/2025.emnlp-main.504.

[475] Rohan Anil et al. *Gemini: A Family of Highly Capable Multimodal Models*. 2025. URL: https://arxiv.org/abs/2312.11805.

[476] DeepSeek-AI et al. *DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning*. 2025. URL: https://arxiv.org/abs/2501.12948.

[477] Patrícia Gameiro, Marcio Lima Inácio, Hugo Gonçalo Oliveira and Ana Alves. 'Sequence Labeling for Pun Location and Detection in Portuguese'. In: *Progress in Artificial Intelligence*. Ed. by Manuel Filipe Santos, José Machado, Paulo Novais, Paulo Cortez and Pedro Miguel Moreira. Cham: Springer Nature Switzerland, 2025, pp. 254–266. ISBN: 978-3-031-73503-5.

[478] Disen Liao and Freda Shi. 'How tokenization limits phonological knowledge representation in language models and how to improve them'. In: *Tokenization workshop (TokShop)*. ICML 2025. Vancouver, BC, Canada, 2025. URL: https://openreview.net/forum?id=ziGpKA7tvN.

[479] Christoph Molnar. *Interpretable Machine Learning: A Guide For Making Black Box Models Explainable*. 3rd. München: Christoph Molnar, 2025. 392 pp. ISBN: 978-3-911578-03-5. URL: https://christophm.github.io/interpretable-ml-book.

[480] Mihai Nadăş, Laura Dioşan and Andreea Tomescu. 'Synthetic Data Generation Using Large Language Models: Advances in Text and Code'. In: *IEEE Access* 13 (2025), pp. 134615–134633. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2025.3589503.

[481] Md. Tauseef Qamar, Shahab Saquib Sohail, Gunjan Ansari and Chandni Saxena. 'The Language of Nuance: Exploring the Limits of Large Language Models in Handling Ambiguity'. In: *Big Data Analytics in Astronomy, Science, and Engineering*. Ed. by Shelly Sachdeva, Yutaka Watanobe and Subhash Bhalla. Cham: Springer Nature Switzerland, 2025, pp. 180–192. ISBN: 978-3-031-86193-2.

[482] Christos Troussas, Akrivi Krouska and Cleo Sgouropoulou. 'A Novel Framework of Human–Computer Interaction and Human-Centered Artificial Intelligence in Learning Technology'. In: *Human-Computer Interaction and Augmented Intelligence: The Paradigm of Interactive Machine Learning in Educational Software*. Ed. by Christos Troussas, Akrivi Krouska and Cleo Sgouropoulou. Cham: Springer Nature Switzerland, 2025, pp. 387–431. ISBN: 978-3-031-84453-9. DOI: 10.1007/978-3-031-84453-9_9.

[483] Ashwin Vaswani, Yashas Samaga, Gaurav Aggarwal, Praneeth Netrapalli and Narayan Hegde. 'All Mistakes are not Equal: Comprehensive Hierarchy Aware Multilabel Predictions (CHAMP)'. In: *Pattern Recognition*. Ed. by Apostolos Antonacopoulos, Subhasis Chaudhuri, Rama Chellappa, Cheng-Lin Liu, Saumik Bhattacharya and Umapada Pal. Cham: Springer Nature Switzerland, 2025, pp. 264–282. ISBN: 978-3-031-78107-0. DOI: 10.1007/978-3-031-78107-0_17.