On the Effect of Low-Ranked Documents: A New Sampling Function for Selective Gradient Boosting

Claudio Lucchese
claudio.lucchese@unive.it
Università Ca' Foscari Venezia
Venice, Italy

Federico Marcuzzi*
federico.marcuzzi@unive.it
Università Ca' Foscari Venezia
Venice, Italy

Salvatore Orlando
orlando@unive.it
Università Ca' Foscari Venezia
Venice, Italy

ABSTRACT

Learning to Rank is the task of learning a ranking function from a set of query-documents pairs. Generally, documents within a query are thousands but not all documents are informative for the learning phase. Different strategies were designed to select the most informative documents from the training set. However, most of them focused on reducing the size of the training set to speed up the learning phase, sacrificing effectiveness. A first attempt in this direction was achieved by Selective Gradient Boosting a learning algorithm that makes use of customisable sampling strategy to train effective ranking models. In this work, we propose a new sampling strategy called High_Low_Sampl for selecting negative examples applicable to Selective Gradient Boosting, without compromising model effectiveness. The proposed sampling strategy allows Selective Gradient Boosting to compose a new training set by selecting from the original one three document classes: the positive examples, high-ranked negative examples and low-ranked negative examples. The resulting dataset aims at minimizing the mis-ranking risk, i.e., enhancing the discriminative power of the learned model and maintaining generalisation to unseen instances. We demonstrated through an extensive experimental analysis on publicly available datasets, that the proposed selection algorithm is able to make the most of the negative examples within the training set and leads to models capable of obtaining statistically significant improvements in terms of NDCG, compared to the state of the art.

CCS CONCEPTS

• **Information systems** → **Learning to rank**; **Retrieval effectiveness**;

KEYWORDS

Learning to Rank, Sampling methods, Gradient Boosting

ACM Reference Format:
Claudio Lucchese, Federico Marcuzzi, and Salvatore Orlando. 2023. On the Effect of Low-Ranked Documents: A New Sampling Function for Selective Gradient Boosting. In *The 38th ACM/SIGAPP Symposium on Applied Computing (SAC '23), March 27-March 31, 2023, Tallinn, Estonia.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3555776.3577597

*Corresponding Author

SAC '23, March 27-March 31, 2023, Tallinn, Estonia

ACM ISBN 978-1-4503-9517-5/23/03…$15.00
https://doi.org/10.1145/3555776.3577597

1 INTRODUCTION

Information Retrieval (IR) is the field of research focused on extracting useful and structured information from a huge clutter of data. Ranking web pages, job positions, or ads in a useful manner for a specific query, employee, or user is a well-known problem in IR. In Machine Learning, the task of building a ranking function is known as Learning to Rank (LtR). The family of LtR solutions encompasses supervised learning algorithms focused on ordering (ranking) objects based on their relevance to user needs.

The main application of LtR is Web Search. Given a user query, the model returns a list of documents ordered by relevance to the query. Through gold standard datasets, where query and documents are represented as feature vectors paired with a relevance label, a LtR algorithm learns to assign a score to each query-document in order to optimise a ranking metric such as NDCG, ERR, etc. [14].

To reflect the real world, these gold datasets are made of thousands of queries and millions of documents. The aim of these datasets is to provide the learning algorithm with positive examples of different relevance grades (relevant to the user needs) and negative examples (irrelevant to the user) to learn a model that assigns document scores based on their relevance. However, not all documents within the dataset are good examples, and so they can compromise the learning phase and lead to poor model generalisation. Furthermore, a large number of documents increases the training time, which is particularly significant in contexts such as online learning.

The current trend of research in Learning to Rank focuses on improving models' effectiveness by designing new objective functions that better approximate discrete ranking metrics or on designing more efficient models through cache-aware algorithms and model pruning strategies. Less attention is given to understanding which documents are useful in the learning phase. The use of document selection strategies to improve the models' effectiveness has not been sufficiently explored. To fill this gap, we defined a new negative-examples sampling strategy for Selective Gradient Boosting (SelGB) called High_Low_Sampl. SelGB is a supervised learning algorithm for the generation of decision tree forests through gradient boosting, in order to solve the LtR task. The algorithm focuses on selecting the most-informative negative examples from the training set, and learning models from a small subset of examples. The new sampling strategy we designed selects a percentage of documents that are likely to be mis-ranked by the model and a percentage of documents that the model ranks

perfectly. The former is fundamental for the learning algorithm to minimise the probability of mis-rank between positive and negative examples. The latter prevents the model from overfitting difficult examples and allows it to generalise over simple ones. Through an extensive evaluation, we empirically proved that our new sampling strategy allows SelGB to train models that significantly improve NDCG over the ones generated with the sampling strategy defined in [12] and the ones learned by the state-of-the-art λ-MART algorithm. Nevertheless, the proposed algorithm allows to reduce the number of documents processed at each iteration of the learning algorithm with respect to λ-MART, with a consequent reduction of the training time.

In summary, the contribution we brought to improve the state of the art in LtR field can be summarised as follow:

- We designed High_Low_Sampl a new samples selection strategy build on top of SelGB but easily adaptable to other learning algorithms. This strategy allows SelGB to select from the training set: *i)* all the positive examples. *ii)* the most informative negative examples to highlight the differences between the positive examples. *iii)* the less informative negative examples in order to avoid the model from overfitting on difficult examples and achieving poor generalisation on unseen examples. We performed an extensive experimental evaluation to show that SelGB combined with our High_Low_Sampl outperforms its previous version and the reference λ-MART algorithm.
- We prove how SelGB equipped with High_Low_Sampl obtains a speed-up in the training process compared to λ-MART without compromising the model effectiveness.
- We show how the low-ranked negative examples selected by High_Low_Sampl allow the models to achieve a higher stability and a lower variance.

The rest of the paper is structured as follows: in Section 2 we discuss the related work. In Section 3 we introduce our new sampling strategy designed on the body of Selective Gradient Boosting. In section 4 we provide an extensive evaluation of Selective Gradient Boosting equipped with our sampling strategy against state of the art and draw the final considerations about this work in Section 5.

2 RELATED WORK

The present work is an improvement over the Selective Gradient Boosting (SelGB) algorithm by Lucchese *et al.* [12]. SelGB is based on the gradient-boosted algorithm λ-MART [18] which can be considered the state-of-the-art in LtR algorithms for the re-ranking stage of the IR pipeline. SelGB implements a dynamic sampling strategy called Sel_Sampl that at each stage of the gradient boosting aimed at dealing with highly unbalanced datasets. Only the positive examples and the currently high-ranked negative ones are considered for growing the next tree of the gradient-boosted forest. The authors showed how this sampling strategy retrieves from a very unbalanced set of documents (with a prevalence of negative examples) only those examples that are useful for the training process by bringing a more effective model and reducing training time due to a reduced number of documents being processed. To prove

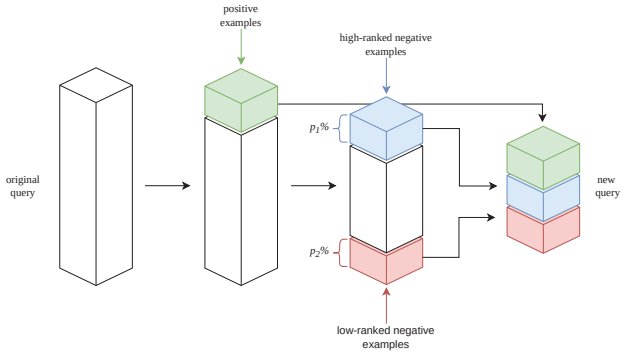

Figure 1: The figure shows how High_Low_Sampl composes a new query by selecting from the original three document classes: the positive examples (in green), the high-ranked negative examples (in blue), and the low-ranked negative examples (in red).

the performance of their learning algorithm and sampling strategy, they published Istella-X [13] the largest public LtR dataset ever released in terms of number of documents per query, with 99.83% negative examples. However, as we show in the experimental section, the algorithm does not bring significant improvements in smaller datasets like MSLR-30K [16], or it may even perform worse than λ-MART as in Yahoo! Learning to Rank Challenge Set 1 [3]. In this work, we discuss how an improved selection of negative examples can provide improvement also on smaller and less unbalanced datasets.

It is worth mentioning that SelGB already proved to be more effective than related sampling strategies such as (Gradient One-Sided Sampling) GOSS [9], or stochastic gradient boosting [6].

Other works previously proposed different document selection strategies that statically pick a set of informative documents to feed the training process [1, 7]. Moreover, (Surrender on Outliers and Rank) SOUR [15], a sampling strategy that removes from the training set those documents that harm the training process. The presented strategy is instead more effective and dynamically selected informative documents at every boosting round.

Finally, negative sampling is now a very common data augmentation approach for improving the robustness [10], for dealing with scarcity of negative examples [17], or for managing unlabelled instances in a semi-supervised scenario [4], and many other tasks. This is not the case of this work as we do not propose a data generation method but a novel strategy to select existing not-relevant instances among the pool of available examples in the given dataset.

3 SAMPLING STRATEGY

Lucchese *et al.* [12] created Selective Gradient Boosting (SelGB) a learning algorithm that exploits a negative-examples sampling strategy called Sel_Sampl to train effective ensembles of decision trees. In Algorithm 1 we report the pseudo-code of Selective Gradient Boosting. The algorithm iterates over a fixed number of iterations N and after every n iterations creates a new training set $\mathcal{D}^*$ through the sampling function (line 14). The new dataset $\mathcal{D}^*$ is used to train the model in subsequent n iterations (from

Algorithm 1 Selective Gradient Boosting

1: **function** SELECTIVE GRADIENT BOOSTING($\mathcal{D}$, N, n, p_1, p_2)
2: **Input**
3: $\mathcal{D}$: training dataset
4: N . ensemble size
5: n : # iterations between consecutive sampling steps
6: p_1 : % high-ranked negative examples at each sampling
7: p_2 : % low-ranked negative examples at each sampling
8: **Output**
9: $\mathcal{E}$: trained ensemble
10: $\mathcal{E} \leftarrow \emptyset$
11: $\mathcal{D}^* \leftarrow \mathcal{D}$
12: **for** m = 1 to N **do**
13: **if** (m mod n) = 0 **then**
14: $\mathcal{D}^* \leftarrow$ HIGH_LOW_SAMPL($\mathcal{D}$, $\mathcal{E}$, p_1, p_2)
15: $\{\lambda_i\} \leftarrow \lambda$-gradients for each $\boldsymbol{x}_i \in \mathcal{D}^*$
16: $\mathcal{R}^* = \{(\boldsymbol{x}_i, \lambda_i)\}$, for all $\boldsymbol{x}_i$ occurring in $\mathcal{D}^*$
17: $t_m \leftarrow$ fit a regression tree to $\mathcal{R}^*$
18: $\mathcal{E} \leftarrow \mathcal{E} \cup t_m$
19: **return** $\mathcal{E}$

line 15 to 18). From the above, it can be deduced that the core of SELGB is the sampling function. The main contribution of this work is the new sampling function HIGH_LOW_SAMPL($\mathcal{D}, \mathcal{E}, p_1, p_2$) (line 14). This function creates a subset $\mathcal{D}^*$ of the original training set $\mathcal{D}$ to perform the next n iteration of the learning algorithm. In detail, as SEL_SAMPL does, HIGH_LOW_SAMPL creates $\mathcal{D}^*$ by selecting from $\mathcal{D}$ all the positive examples $\mathcal{D}^+$ and a subset of the negative examples $\mathcal{D}^-$. The main difference with SEL_SAMPL is the way the negative examples are selected from $\mathcal{D}^-$. For every query q, SEL_SAMPL selects from $\mathcal{D}^-$ the $p\% \cdot |\mathcal{D}^-|$ examples with the highest score estimated by the current model $\mathcal{E}$. Instead HIGH_LOW_SAMPL selects the $p_1\% \cdot |\mathcal{D}^-|$ examples with the highest score and the $p_2\% \cdot |\mathcal{D}^-|$ examples with the lowest score. Also in this case the scores are estimated by the current model $\mathcal{E}$. Trivially, if $p_1 = p$ and $p_2 = 0\%$ the two strategy coincide. The final cardinality of the subset $\mathcal{D}^*$ is $|\mathcal{D}^*| = |\mathcal{D}^+| + (p_1 + p_2)\% \cdot |\mathcal{D}^-|$. The newly created training set $\mathcal{D}^*$ is used to perform the next n iterations of the learning algorithm, and then a new sampling is performed. Figure 1 graphically depicts how HIGH_LOW_SAMPL creates the new queries of the datasets.

SEL_SAMPL selects the high-ranked negative examples to enhance the discrimination between relevant and irrelevant documents and consequently minimise the mis-ranking risk. Nevertheless avoiding the model to see the low-ranked negative examples during training can lead to overfitting on the most-informative negative examples. In HIGH_LOW_SAMPL we extended this idea by introducing negative examples with low scores to balance the attention of the model and prevent it to focus only on high-ranked examples.

The pseudo-code in Algorithm 1 is designed to work on GBDTs but can be easily generalised to handle other learning algorithms. Therefore, any learning algorithm that processes instances iteratively during the training phase can be applied (e.g. boosting iteration in GBDTs and epoch in NNs).

Table 1: Datasets properties.

	ISTELLA-X	MSLR-30K	YAHOO!
#features	220	136	519
#queries	10,000	31,531	29,921
avg. query length	2,679.14	119.60	23.72
#documents	26,791,447	3,771,125	709,877
%negative examples	99.8%	51.5%	26.1%

4 EXPERIMENTAL SETUP

In this section, we presented the datasets used for the evaluation. We introduced the state of the art and the other competitors and how we performed the hyperparameter tuning. All models were trained through the LightGBM open source library [8]. We used RankEval analysis framework [11] to measure the statistical significance of the improvement brought by HIGH_LOW_SAMPL to SELGB with respect to the others through a Fisher's randomisation test [5] with a one-sided p-value (p = 0.01).

4.1 Datasets

We performed our experiments on three publicly available datasets: ISTELLA-X [13], MSLR WEB30K FOLD 1 [16] and YAHOO! LEARNING TO RANK CHALLENGE SET 1 [3], summarised in Table 1. For each dataset, documents are labelled with graded relevance ranging from 0 to 4, where 0 refers to negative examples and relevance greater than 1 refers to positive examples. All datasets have a different percentage of negative examples: ISTELLA-X with 99.8% out of 26,791,447, MSLR-30K with 51.5% out of 3,771,125 and YAHOO! with 26.1% out of 709,877. The datasets come with a predefined training (60%), validation (20%) and test (20%) split used to perform learning, hyperparameter tuning/model selection and comparisons.

4.2 Models

As a baseline, we use the LightGBM [8] software implementation of LambdaMART, to which we refer as λ-MART [18]. For sake of simplicity, from hereinafter, we will refer to the original algorithm proposed in [12] as SELGB$_S$, and to SELECTIVE GRADIENT BOOSTING equipped with our sampling strategy HIGH_LOW_SAMPL as SELGB$_{HL}$. Both SELGB$_S$ and SELGB$_{HL}$ are implemented on top of LightGBM. For all the learning strategies we kept the default gradient normalisation applied by LightGBM library. Gradient normalisation has an important impact on model performance. This explains why the performances in this work are higher than those reported in [12].

Hyperparameters tuning follows previous works [2, 12]: max_bin is 255, σ is 1, min_sum_hessian_in_leaf is 0, and lambdamart_norm is set to true. For ISTELLA-X, YAHOO!, and MSLR-30K respectively: learning_rate: 0.05, 0.02, and 0.02, num_leaves: 64, 200, and 400, and min_data_in_leaf: 20, 50, and 50. Each model is trained up to 1,000 trees with early stopping criteria based on the model performance on the validation set. The hyperparameter n of both SELGB$_S$ and SELGB$_{HL}$ is set to 1. After hyperparameter tuning, we chose the best models through model selection on the validation sets.

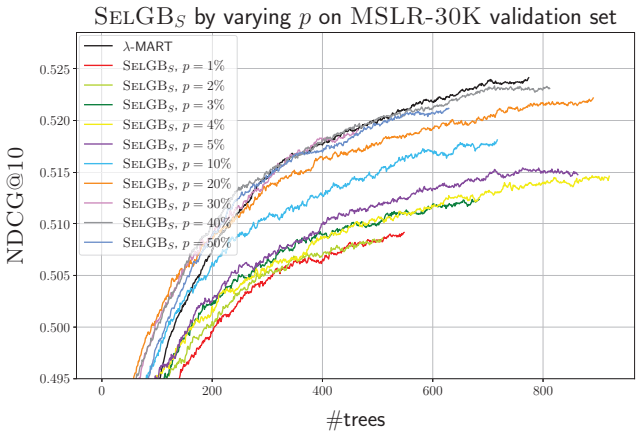

Figure 2: Performance of SELGB$_S$ on MSLR-30K validation set, by varying hyperparameter p.

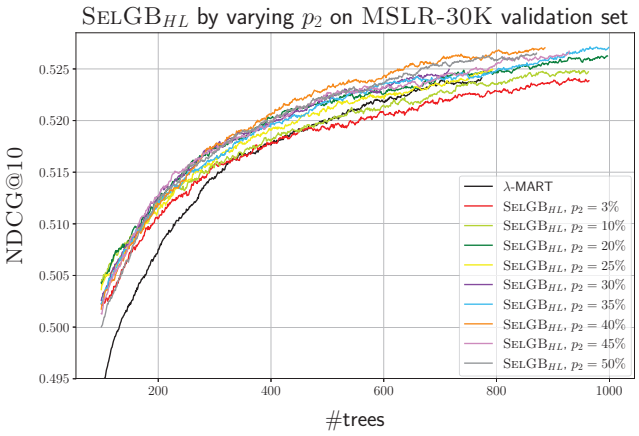

Figure 3: Performance of SELGB$_{HL}$ on MSLR-30K validation set, with $p_1 = 20\%$ and by varying hyperparameter p_2.

4.3 Evaluation

For each dataset, we performed hypertuning on the parameter p of SELGB$_S$. In Figure 2 we evaluated the effectiveness of SELGB$_S$ on MSLR-30K validation set, by varying the parameter p. Performances are evaluated in terms of NDCG@10. For the sake of completeness, in Figure 2 we also added the baseline λ-MART.

As can be seen from Figure 2, the performances of SELGB$_S$ degrade as p decreases (i.e. as the number of removed negative examples increases). Thus, SELGB$_S$ is not able to improve the performance in terms of NDCG@10 compared to the baseline on the MSLR-30K dataset. We observed the same behaviour on YAHOO!. Instead, the performances on ISTELLA-X are in line with those reported in the original work [12].

As concern hyperparameters p_1 and p_2 of SELGB$_{HL}$, in order to avoid a quadratic grid search, we select the best three p values from SELGB$_S$ hypertuning. The p_1 values are: 20%, 30% and 40% for MSLR-30K and YAHOO! and 0.25%, 0.5% and 1.0% as p_1 for ISTELLA-X. Then we tuned the value of p_2 for each value of p_1. In Figure 3 we drew the performance of SELGB$_{HL}$ on MSLR-30K, with $p_1 = 20\%$ (the best value for p_1 on MSLR-30K) and by varying p_2.

Also in this case we added the baseline. For clarity, Figure 3 shows only the models that obtain NDCG@10 higher than the baseline on the validation set.

In Table 2 we summarised for each dataset, the performance obtained by the best models trained with each learning algorithm. The models were selected through model selection based on the performance obtained on the validation set. Model performances are in terms of NDCG@10 evaluated on the test set.

SELGB$_{HL}$ with $p_1 = 20\%$ and $p_2 = 40\%$, is the best model trained with our new samples selection strategy, and it achieves a statistically significant improvement with respect to both the baseline and SELGB$_S$. Training with SELGB$_{HL}$ allows to have more effective models also at the beginning of the learning phase. This phenomenon is particularly evident in Figure 3, where the performance of the models is drawn incrementally.

We observed similar behaviour on YAHOO!, where SELGB$_{HL}$ is able to achieve statistically significant improvement with different ensemble sizes over both λ-MART and SELGB$_S$. On the contrary, on ISTELLA-X we observed a different trend. Both SELGB$_{HL}$ and SELGB$_S$ achieve statistically significant improvement in effectiveness with respect to λ-MART, but there is no statistical evidence that SELGB$_S$ is better than SELGB$_{HL}$ and *vice versa*.

Table 2: Comparison in terms of NDCG@10 between SELGB$_{HL}$ against the state of the art. Statistically significant differences w.r.t. SELGB$_{HL}$ according to Fisher's randomisation test [5], with a one-sided p-value and p = 0.01, are marked with *.

Algorithm	#Trees 150	350	550	Full
		MSLR-30K		
λ-MART	0.5038*	0.5186*	0.5231*	0.5246*
SELGB$_S$, $p = 40\%$	0.5089	0.5197	0.5231*	0.5262*
SELGB$_{HL}$, $p_1 = 20\%$, $p_2 = 40\%$	0.5098	0.5209	0.5259	0.5297
		YAHOO!		
λ-MART	0.7804*	0.7889	0.7929	0.7946*
SELGB$_S$, $p = 30\%$	0.7819	0.7892	0.7915*	0.7937*
SELGB$_{HL}$, $p_1 = 40\%$ $p_2 = 30\%$	0.7829	0.7896	0.7925	0.7958
		ISTELLA-X		
λ-MART	0.7501*	0.7664*	0.7722	0.7723*
SELGB$_S$, $p = 1\%$	0.7732	0.7858	N/D	0.7865
SELGB$_{HL}$, $p_1 = 1\%$ $p_2 = 2\%$	0.7715	0.7839	N/D	0.7847

4.4 Training Time

In this section, we evaluated the models' performance in terms of efficiency. Since each algorithm trains a model on a different size of the training set, we summarised in Table 3 both the training time per tree (one tree for each boosting iteration) and the total training time. All experiments are performed in a single thread on a machine equipped with Intel(R) Xeon(R) Silver 4110 CPU @ 2.10GHz and operating system Ubuntu 20.04.4 LTS.

Table 3: Algorithms efficiency in terms of training time. *best* **stands for the hyperparameters found through model selection with respect to models' effectiveness.**

Algorithm	Dataset		
	Istella-X	MSLR-30K	Yahoo!
	training time per tree		
λ-MART	13.2 sec	2.8 sec	1.8 sec
SelGB$_S$, *best*	3.7 sec	2.2 sec	1.6 sec
SelGB$_{HL}$, *best*	3.9 sec	2.3 sec	1.6 sec
	total training time		
λ-MART	128.7 min	35.7 min	21.3 min
SelGB$_S$, *best*	25.3 min	30.0 min	18.3 min
SelGB$_{HL}$, *best*	28.5 min	34.3 min	22.7 min
	percentage of training set		
λ-MART	100.0%	100.0%	100.0%
SelGB$_S$, *best*	1.1%	58.4%	81.6%
SelGB$_{HL}$, *best*	3.1%	79.0%	91.2%

In Table 3 we refer as *best* to the hyperparameters found through model selection on the validation set with respect to models' effectiveness. The best hyperparameters are summarised in Table 2.

The asymptotic complexity of High_Low_Sampl and Sel_Sampl is $O(n \log n)$ with n the number of negative examples with the query. In detail, the cost of the strategies lies in the cost of the sorting algorithm used to rank the negative example to retrieve the low-ranked and high-ranked documents. However, this cost can be totally amortised since, at each iteration of the learning algorithm, the algorithm itself orders the examples to evaluate the performance of the model. Consequently, both High_Low_Sampl and Sel_Sampl have no overhead on the training time. Moreover, due to the lower number of documents being processed at training time, with the same number of iterations, the cost of training SelGB$_S$ and SelGB$_{HL}$ is less than or equal to that of λ-MART .

Finally, if we are more interested in efficiency at training time, rather than improving model effectiveness, it is possible to apply a more aggressive sampling (i.e. $p_2 = 3\%$ or 10% as shown in Figure 3) and obtain similar performance to λ-MART.

4.5 Query Analysis

In this last part of the experimental section, we analysed the impact that low-ranked negative examples have on the model prediction and so why High_Low_Sampl is preferable to Sel_Sampl. In particular, we wanted to highlight how ignoring the low-ranked negative examples can harm the model's effectiveness. We analysed this phenomenon on models trained with both SelGB$_S$ and SelGB$_{HL}$ and we showed how the model trained with SelGB$_{HL}$ is more stable and effective due to the introduction of the low-ranked negative examples.

To do so, we sampled a subset of queries from MSLR-30K training set and we observed how documents' ranks vary after each tree in both SelGB$_S$ and SelGB$_{HL}$. As hyperparameters, we used the best values found through model selection, the same reported in Table 2. We set $p = 40\%$ for Sel_Sampl and $p_1 = 20\%$ and $p_2 = 40\%$ for High_Low_Sampl. In Figure 4 we show the result of this analysis for one specific query (query ID: 1349), but we highlight that the same behaviour was observed in all queries. Figure 4 shows with a colour the label of the document ranked at the i-th position (on the x axis) after the t-th tree of the forest (on the y axis). The green colour stands for a relevant document with label > 0. For the sake of simplicity, we sampled only queries where positive documents have the same relevance label. The red colour stands for those documents that are consistently scored low, i.e., those documents that occur in the bottom p_2 portion of the rank positions after each of the 1000 trees. We computed this set for SelGB$_{HL}$ (on the right-hand side) and we report their rank positions also for SelGB$_S$ (left-hand side). We denote this set with $\mathcal{D}_{HL}^{-\cap}$. The light grey colour is used for the remaining non-relevant documents. The width of each rank position is proportional to the logarithmic discount factor of NDCG, so as to highlight the top-ranked positions that contribute the most to the quality of the ranking.

There are a number of interesting insights we can provide with this fine-grained analysis (see Figure 4).

- *i)* the documents within the set $\mathcal{D}_{HL}^{-\cap}$ are roughly the same across boosting iterations. This might be expected from its definition. In fact, after 1000 trees, $\mathcal{D}_{HL}^{-\cap}$ always cover the 50% of the lowest p_2 portion of the ranking. Computing the same statistic on SelGB$_S$ leads to a poor 10%, and just 15% at iteration 66.
- *ii)* the negative examples in $\mathcal{D}_{HL}^{-\cap}$ have a very stable rank in SelGB$_{HL}$ (red squares, right-hand side), but the same documents have a cluttered behaviour during the training with SelGB$_S$ (red squares, left-hand side).
- *iii)* also positive examples not placed in the top-k position (rightmost green traces) have a very stable behaviour in SelGB$_{HL}$ (right chart); this does not hold for SelGB$_S$ (left chart), where their ranking varies significantly, tree after tree.
- *iv)* in SelGB$_{HL}$, some positive examples require a few more iterations to reach the top-k position of the list, but in the end, they reach higher positions than in SelGB$_S$. As a result, equipping Selective Gradient Boosting with the proposed sampling strategy High_Low_Sampl produces more effective models.

All of the above confirms that the ranking provided by SelGB$_{HL}$ is of higher accuracy than SelGB$_S$. Some of the reasons for this accuracy might be found in the stability provided by the lowest-ranked document. The set of lowest-ranked documents provides the model with interesting information about the training process that translated into higher stability, reduced variance, and the opportunity to better refine the predicted document scores.

5 CONCLUSION

We developed a new negative-examples selection strategy called High_Low_Sampl for Selective Gradient Boosting. We have shown how the selected documents can learn more effective models in terms of NDCG. Through extensive experiments, we empirically proved the proposed strategy has a statistically significant increase in performance compared to the state of the art. Moreover, we have demonstrated that the proposed strategy improves at the same time both the effectiveness of the trained models and the efficiency of the

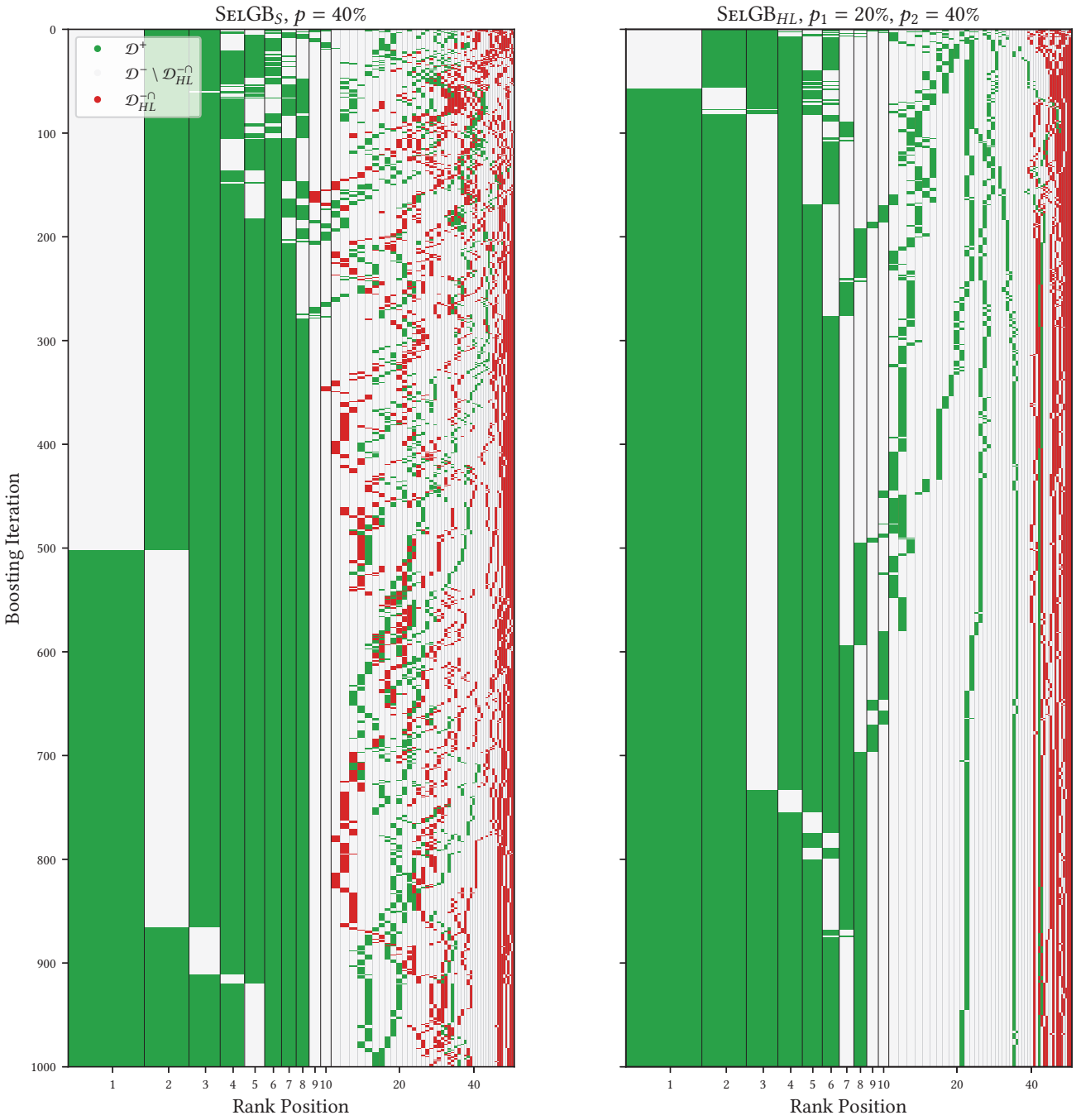

Figure 4: Document rankings (x axis) at each boosting iteration (y axis) for both SelGB$_S$ (left) and SelGB$_{HL}$ (right) for query 1349 sampled from MSLR-30K training set. The rank position widths are proportional to the logarithmic discount factor of NDCG. In green colour are query documents with relevance > 0. In red colour are documents consistently scored low by High_Low_Sampl in 1000 boosting iterations. In light grey colour are the remaining non-relevant documents.

training phase. Finally, we have shown High_Low_Sampl retrieves useful information from the training set that enriches the learning process and brings to models with higher stability and less variance.

ACKNOWLEDGEMENT

This work was partially supported by iNEST (Interconnected NordEst Innovation Ecosystem, Project ID: ECS 00000043), funded by PNRR (Mission 4.2, Investment 1.5) NextGeneration EU.

REFERENCES

[1] Javed A. Aslam, Evangelos Kanoulas, Virgiliu Pavlu, Stefan Savev, and Emine Yilmaz. 2009. Document selection methodologies for efficient and effective learning-to-rank. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009*, James Allan, Javed A. Aslam, Mark Sanderson, ChengXiang Zhai, and Justin Zobel (Eds.). ACM, 468–475. https://doi.org/10.1145/1571941.1572022

[2] Sebastian Bruch. 2021. An Alternative Cross Entropy Loss for Learning-to-Rank. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 118–126. https://doi.org/10.1145/3442381.3449794

[3] Olivier Chapelle and Yi Chang. 2011. Yahoo! Learning to Rank Challenge Overview. In *Proceedings of the Yahoo! Learning to Rank Challenge, held at ICML 2010, Haifa, Israel, June 25, 2010 (JMLR Proceedings)*, Olivier Chapelle, Yi Chang, and Tie-Yan Liu (Eds.), Vol. 14. JMLR.org, 1–24. http://proceedings.mlr.press/v14/chapelle11a.html

[4] John Chen, Vatsal Shah, and Anastasios Kyrillidis. 2020. Negative sampling in semi-supervised learning. In *International Conference on Machine Learning*. PMLR, 1704–1714.

[5] Ronald A. Fisher. 1935. *The design of experiments. 1935.* Oliver and Boyd, Edinburgh.

[6] Jerome H Friedman. 2002. Stochastic gradient boosting. *Computational statistics & data analysis* 38, 4 (2002), 367–378.

[7] Muhammad Ibrahim and Mark Carman. 2014. Undersampling techniques to re-balance training data for large scale learning-to-rank. In *Asia Information Retrieval Symposium*. Springer, 444–457.

[8] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 3146–3154. https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html

[9] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 30 (2017).

[10] Yang Liu and Hongyi Guo. 2020. Peer loss functions: Learning from noisy labels without knowing noise rates. In *International Conference on Machine Learning*. PMLR, 6226–6236.

[11] Claudio Lucchese, Cristina Ioana Muntean, Franco Maria Nardini, Raffaele Perego, and Salvatore Trani. 2017. RankEval: An Evaluation and Analysis Framework for Learning-to-Rank Solutions. In *SIGIR 2017: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

[12] Claudio Lucchese, Franco Maria Nardini, Raffaele Perego, Salvatore Orlando, and Salvatore Trani. 2018. Selective Gradient Boosting for Effective Learning to Rank. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz (Eds.). ACM, 155–164. https://doi.org/10.1145/3209978.3210048

[13] Claudio Lucchese, Franco Maria Nardini, Raffaele Perego, Salvatore Orlando, and Salvatore Trani. 2018. Selective Gradient Boosting for Effective Learning to Rank. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz (Eds.). ACM, 155–164. https://doi.org/10.1145/3209978.3210048

[14] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge University Press. https://doi.org/10.1017/CBO9780511809071

[15] Federico Marcuzzi, Claudio Lucchese, and Salvatore Orlando. 2022. Filtering out Outliers in Learning to Rank. In *Proceedings of the 2022 ACM SIGIR International Conference on Theory of Information Retrieval (ICTIR '22)*. Association for Computing Machinery, New York, NY, USA, 214–222. https://doi.org/10.1145/3539813.3545127

[16] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 Datasets. *CoRR* abs/1306.2597 (2013). http://arxiv.org/abs/1306.2597

[17] Peifeng Wang, Shuangyin Li, and Rong Pan. 2018. Incorporating gan for negative sampling in knowledge representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[18] Qiang Wu, Christopher J. C. Burges, Krysta M. Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Inf. Retr.* 13, 3 (2010), 254–270. https://doi.org/10.1007/s10791-009-9112-1