Cross-Dataset Data Augmentation for Convolutional Neural Networks Training

Andrea Gasparetto, Dalila Ressi, Filippo Bergamasco, Mara Pistellato
Luca Cosmo, Marco Boschetti, Enrico Ursella and Andrea Albarelli
Università Ca' Foscari Venezia
Dipartimento di Scienze Ambientali Informatica e Statistica
via Torino, 155 - Venice, Italy

***Abstract*—Within modern Deep Learning setups, data augmentation is the weapon of choice when dealing with narrow datasets or with a poor range of different samples. However, the benefits of data augmentation are abysmal when applied to a dataset which is inherently unable to cover all the categories to be classified with a significant number of samples. To deal with such desperate scenarios, we propose a possible last resort: Cross-Dataset Data Augmentation. That is, the creation of new samples by morphing observations from a different source into credible specimens for the training dataset. Of course specific and strict conditions must be satisfied for this trick to work. In this paper we propose a general set of strategies and rules for Cross-Dataset Data Augmentation and we demonstrate its feasibility over a concrete case study. Even without defining any new formal approach, we think that the preliminary results of our paper are worth to produce a broader discussion on this topic.**

I. Introduction

Convolutional Neural Networks (CNNs) are a very effective and versatile tool to address a wide range of Computer Vision classification problems. Unfortunately, such power comes at a price: CNNs need to be trained, and this task, in order to succeed, often requires a huge amount of available samples for each category to be recognized. Since this is not always the case, specialized techniques to deal with sample scarcity have been proposed in literature and widely studied during the last years. Two of the most important categories of approaches to tackle this problem are certainly Data Augmentation and Transfer Learning.

Data Augmentation is about variance: while the exact number of samples is strictly tied to the particular case-of-study, it is central for the data to have a good diversity to allow the network to efficiently generalize the object of interest. In other words, the more diversity the network can *see*, the better the results. Data Augmentation tackles the problem by creating new samples out of the available ones. In particular, new data is generated through the application of several transformations to the original images. Scaling, translation, rotation, flipping, noise addition, perspective transform, color balance are some of the most widely used transformation when it comes to augmentation [1]. To this end, these techniques can be regarded more as a way of simulating new capturing conditions rather than new samples for the categories of interest. But generic data augmentation [2] is not the only successfull data augmentation technique. Indeed, more advanced techniques have been proposed recently. In particular, Goodfellow *et al.* [3] show how to generate new samples after being trained on samples drawn from some distribution in their seminal work *Generative Adversial Networks*. Rogez and Schmid [4] propose a scheme that artificially inflate the data set by using domain specific synthesization to produce more training data. A similar approach have been proposed by Peng *et al.* [5].

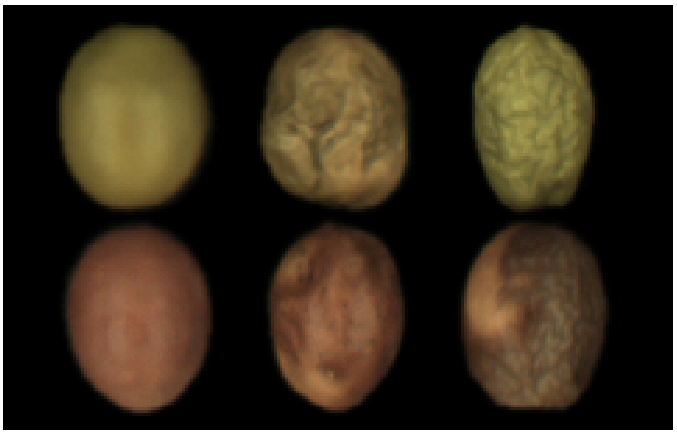
Fig. 1. Edremit and Pink are two families of delicious Turkish olives. They are affected by similar defect patterns which must be detected. However Pink are more rare and it is difficult to gather a significant dataset. Would it be possible to design a training process exploiting the abundance of Edremit to better classify Pink ?

A very different approach to the same task is the one taken by Transfer Learning [6]. In this case the problem is solved by means of a machine learning method where a model developed for a task is reused as the starting point for a model on a second task, or, as defined in [7], *transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned*. From a more practical point of view, transfer learning involves the usage of pre-trained models (which are usually general enough to cope with different tasks) as the starting point, allowing to re-train just a part of the network to improve the performance on a specific setting.

The advantages are twofold. First, training a network from scratch is a time and resource consuming task, in particular if such training happens on a very large dataset. Secondly, training a network for a more general purpose task allows to use huge database freely available, like ImageNet [8] (which counts millions labelled images belonging to a thousand different categories).

Transfer learning comes into two different flavours. The first takes the name of *Develop Model Approach*. This technique expects to use a related predictive modelling problem (for which a lot of data is available) to train the network. Next, the model is reused as the starting point for a model of our task of interest, usually after a fine tuning step. The second approach, which is the most common one, takes the name of *Pre-trained Model Approach*, and makes use of model trained on large and challenging datasets. The use of datasets with a great variety of objects allows to train networks which are capable to generalize different task. Again, these network are used as a starting point and re-trained to cope with the specific case-of-study that must be dealt with.

The main idea underlying this paper is to spread a bridge between these two widely adopted strategies. Specifically we are dealing with the case where very few samples are available for the family of objects to be classified, still large datasets exists for similar, but not identical, objects. Within this scenario we are seeking to exploit the information from the second set (and from networks trained on it) to obtain a network capable of classifying objects from the first set. For this process to be feasible a few conditions must hold. Broadly speaking, our ideal scenario provides objects that are mostly similar between the two datasets except for some features. Such set of inter-dataset distinctive features must be disjoint with the set of the inter-category features that are to be used for classification within each dataset. These inter-category features, in turn, must be shared between datasets.

While, at this stage, it is difficult to generalize some cross-dataset data augmentation approach, with this paper we try to define some basic principles and to define a general recipe to adopt to deal with similar scenarios. In particular, we are interested in comparing different combinations of augmentation strategies to see which one yields better results. In order to perform this task, we apply cross-dataset data augmentation to solve a real-world problem: the classification of olive fruits that can be affected by specific defects. By doing this, we supply additional collateral contributions by designing two novel network architectures to cope with fruit defect detection with special attention to performance and by releasing the datasets used in this work.

II. Cross-Dataset Data Augmentation

According to the generally adopted semantic within the Deep Learning field, data augmentation is any synthetic alteration of dataset images, performed with the goal of inflating the number of samples to be supplied to the training process.

In this paper we introduce the concept of *Cross-Dataset Data Augmentation* (CDDA), a term used to describe a novel pre-processing strategy involving the transfer of information between two datasets by means of image processing. The term recall the idea of a data augmentation step which, instead of processing the samples internal to a dataset, works by transforming samples between a source and a destination dataset.

The main assumption is the existence of a strong relation between the source and destination datasets. That is, we assume that they share most of the features that are significant to perform a certain classification task, while they differ with respect to features that can be ignored by the classifier.

Specifically, we model each dataset as a (possibly infinite) set of features (see Fig. 2). Some feature are global, such as color histograms, orientation of the blob, PCA components, etc. Some others are local, such as intensity maxima, corners, and so on.

Given two datasets, A and B, each of their features can be deemed to belong to one of the following three sets:

- *Dataset Characteristic Features (DCF)*: these are features that distinguish one dataset from the other. The idea is that these features should not take a role in the classification we want to perform. For instance, if we want to classify vehicles with respect to the size of the wheels, then the number of the wheels is a DCF. In fact, it distinguishes bikes from cars, still it is not a constraint if the classification is about the size and not the number of wheels. Of course, this does not mean that DCFs are not distinctive in general. Of course a classifier trained to distinguish bikes from cars will exploit them. They are just not distinctive with respect to the specific classification task we are dealing with.
- *Category Characteristic Features (CCF)*: these are features that are important for the classification task we are considering. For CDDA to work, CCFs must be shared between datasets A and B. Going back to the example with vehicles and wheel sizes, the radius of the tire is a DCF and the number of spokes could be one, depending on the classifier. In principle a classifier that uses CCFs to distinguish A from B should not work as they are preserved between the two datasets.

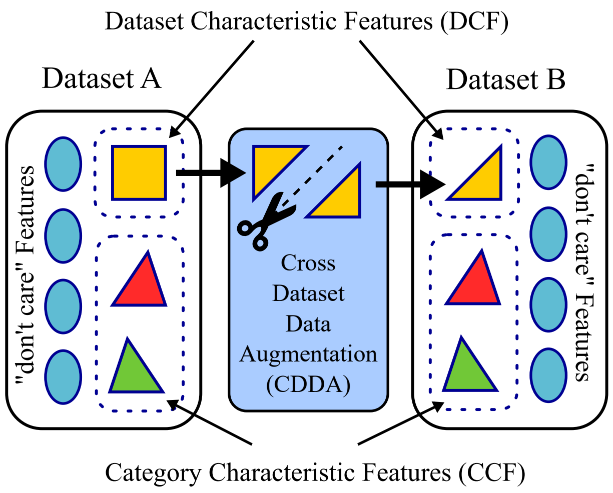
Dataset Characteristic Features (DCF)
Dataset A
Dataset B
"don't care" Features
Cross Dataset Data Augmentation (CDDA)
"don't care" Features
Category Characteristic Features (CCF)

Fig. 2. A graphical representation of the main concept underlying Cross-Dataset Data Augmentation.

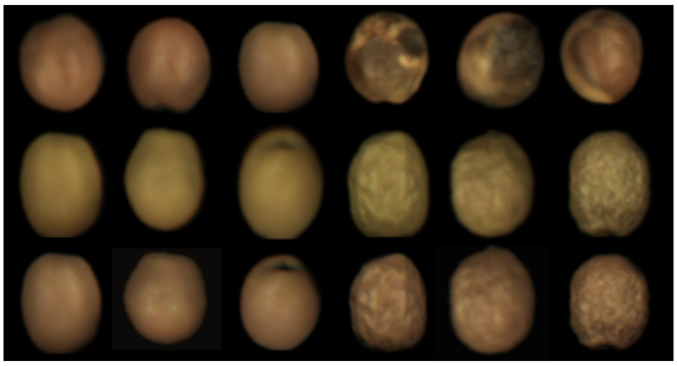

Fig. 3. Some samples drawn from the dataset used in the experimental session. The rows show, respectively, sample from the Red dataset (good on the left, wrinkled on the right), from the Green dataset and the same images from the second row transformed according to the proposed Cross-Dataset Data Augmentation approach.

- *"don't care" features*: these are simply the features that are in the datasets, but are not significant for a specific classification task.

Let A and B two *strongly related* datasets, with B that contains too few samples to be effectively used to train a CNN. We assume the two datasets to be strongly related if there exists a set of features in common (CCF) and a disjointed set of features that allows to distinct samples belonging to the two (DCF). Then, the CDDA is some image processing function $f_{CDDA} : A \to B$ that transforms samples belonging to the dataset A into samples that could belong to the dataset B. This is achieved by applying a transformation to the DCF features of the sample. This results in a new dataset with higher cardinality that could be used to train a more efficient CNN. Our bet is that the newly forged dataset allows to train more efficient networks which outperform networks trained using standard data augmentation strategies. Of course the nature of f_{CDDA} strongly depends on the characteristics of A and B and it is not possible to give a general rule at this point. However it is possible to experiment the effectiveness of the concept with a real-world case.

In our specific case-of-study, we address the problem of low cardinality dataset in an image classification task. While a more in-depth introduction of the datasets can be found in Section III, we anticipate that the dataset contains olive images belonging to two different families. Figure 3 shows some sample images from the dataset. As will be shown in Section V, the number of samples are not enough to achieve satisfying accuracy level, in particular when compared to the results obtained on a similar dataset (Green olives dataset). On the other hand, the Green dataset can be considered *strongly related* to the Red dataset and thus a CDDA approach can be used to transform samples from the Green dataset to sample that belongs to the Red dataset.

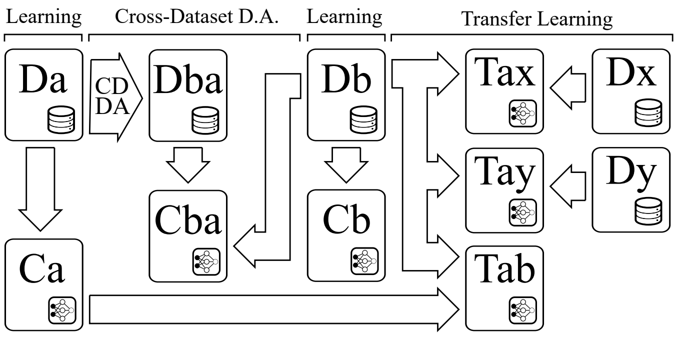

Fig. 4. An overview of the possible strategies for data augmentation, including both traditional methods and cross-dataset data augmentation. *D* blocks represent dataset used to train a network, while a *C* block is a classifier obtained after the optimization process.

In this first stage of the study, we decided to focus on the two most trivial differences between samples belonging to the two datasets, namely the shape and the colour of the olives. Thus, we define a two-step transformation function. Both transformations include a learning phase in which a standard representation of that particular feature is learned.

In the first step, we compute a set of parameters tied to the geometry (the shape) of the olives. In particular, after the extraction of the boundary from the images, we fit a bounding box and retrieve both the two perpendicular axes and their orientation. Then, we compute the *mean bounding box* representative of the red olives used to compute the affine transformation between each green olive bounding box to the mean red olive one. This results in the creation of several *reddish-shaped* green olives.

To deal with the colour difference, we simply learn the mean colour histogram for each channel and then apply a histogram equalizer in order to match the colour histogram of the green olives with the mean colour histogram of the red ones. The third row in Figure 3 shows the result of the application of these two steps to the olives in the second row.

III. Datasets Overview

The olives data evaluated in this paper consists of two fruit varieties: Endremit olives (characterized by a greenish texture) and Pink olives (which are more reddish). The olive images were captured in a controlled environment. A carousel transports the olives through an obscuration box while simultaneously rotating them along their major axis (using a system of parallel rolls). The camera has both an rgb and a infra-red sensor used to take 8 images for each olive sample. This results in a collection of images from different angles as the machine let the olives spin around their major axis. Since not specifically related to our thesis, we restrict the analysis on the rgb channels only, and use the infra-red just for background-foreground segmentation. Nevertheless, we will release the complete (full-channel) dataset along with this paper.

We created a test bed to compare the performance of the proposed data augmentation strategy with respect to some of the more well-established data augmentation techniques. In particular, we will test a set of different datasets (which are the result of different pre-processing strategies) on both pre-trained networks (through transfer learning) and *ad hoc* networks, trained from scratch. The performance are assessed

	Test 1 Green+Red | TRed	Test 2 Green|TRed	Test 3 Green|VRed|TRed	Test 4 G2R|TRed	Test 5 G2R|VRed|TRed	Test 6 G2R+Red|VRed|TRed
FruitDef	80.157	59.175	59.461	85.591	86.387	98.782
FruitDefV2.0	82.479	62.899	63.227	86.555	88.866	98.739
AlexNet	89.748	95.582	96.301	97.991	97.629	96.504
VGG16	90.555	95.537	94.139	98.396	99.581	97.699
VGG19	94.685	88.730	89.859	98.744	98.814	98.773
GoogleNet	96.892	96.093	98.372	99.650	99.650	99.028
ResNet50	96.683	97.350	96.373	98.605	98.117	99.744
Inception-v3	97.301	98.465	98.601	99.930	99.841	99.744

TABLE I
PERFORMANCE ACHIEVED BY SEVERAL NETWORKS IN TERMS OF CLASSIFICATION ACCURACY. TEST 4 TO 6 SHOW THE RESULTS ON THE DATASET CREATED USING THE PROPOSED DATA AUGMENTATION APPROACH.

in terms of classification accuracy. In the proposed case-of-study, the datasets contain olive images belonging to two different categories, the ones labelled as *Good* and the the ones labelled as *Wrinkled*. The former contains defect-less olives, while the latter contains olives with wrinkles on the surface.

The first dataset (which will be referred to as *Green*) contains 14340 *green* olives, of which 7580 belongs to the *good* category while 6760 are *wrinkled*. The second dataset, which is much smaller, contains a different type of olives, which will be called the *red* ones. The *Red* dataset consists of 108 olives, divided into 60 belonging to the *good* class and 48 which show surface wrinkles.

As introduced in Section II, our Cross-Dataset Data Augmentation counts two independent pre-processing steps that are pipelined together in order to transform a numerous but different dataset of images (the Green dataset) into the less-numerous dataset containing red olives. We will refer to this dataset as the *Green2Red* (*G2R* in shorts) dataset.

The performance of each net is assessed against eight different datasets. Each dataset is a combination of olives took from the *Green*, the *Red* and the *Green2Red* datasets. Images contained in a dataset are usually splitted into three subsets. The first, the most numerous one, contains the images which will be used to learn the weights of the net and is referred to as the *training set*. Basically, these are the sample data used to fit the model. The second is the *validation set*. It contains the sample data used both to provide an unbiased evaluation (at least at the beginning) of a model fit on the training dataset and tuning model hyper-parameters. The last set is the test set, which contains the sample data used to provide an unbiased evaluation of the final model. Each network has been trained and tested on the following aggregated datasets.

- *Green*: the networks are trained and tested only on olives contained in the Green dataset. The green dataset contains a large number of samples, giving a basic idea of the performance that can be achieved in this specific case-of-study in the best possible scenario.
- *Red*: the dataset contains all (and only) the olives belonging to the Red dataset. The images are divided into the three sub-datasets introduced above, training (90%), validation (5%) and test (5%). Like for the previous one, the performance achieved on this dataset should be used only as a reference for the worst scenario for this particular case-of-study.
- Test 1 (*Green+Red*): in this test, the dataset contains olives from both the Green and Red dataset. Red olives are distributed consistently among the training, the validation and the test set (with a $8:1:1$ ratio).
- Test 2 (*Green*|*TRed*): the dataset contains all the olives belonging to the Green dataset, which are splitted into a training and validation set, while the test set contains only red olives.
- Test 3 (*Green*|*VRed*|*TRed*): the difference between the previous dataset and the current one is that a subset of red olives is used both for validation and test purposes, while the training set contains only Green olives.
- Test 4 (*G2R*|*TRed*): in this test, the training and the validation set images are picked from the *Green2Red* dataset (*i.e.* green olives after the proposed preprocessing steps), while for the testing phase we used only red olives.
- Test 5 (*G2R*|*VRed*|*TRed*): in the fifth test we train the networks on the *Green2Red* dataset, while the red olives are used in both validation and test set.
- Test 6 (*G2R+Red*|*VRed*|*TRed*): in the final test, we split olives belonging to the *Red* dataset among the training, the validation and the test set. Like in *Test 1*, red olives are consistently distributed with the same ratio.

Each test has been created to answer to a specific question. In particular, test 1 to 3 should answer to the question: *are the two dataset similar enough to allow networks mainly trained on the most numerous dataset to perform well on the less numerous one?* On the other hand, test 4 and 5 show how the proposed approach tackle the problem of low sample number datasets in our specific scenario. All datasets have been augmented with scale, random rotations, PCA transformation, flipping and translation.

The cross-dataset data augmentation and the tests on the neural networks (pre-trained nets and the *ad hoc* ones) are implemented in Matlab using the Neural Network Toolbox. Both the source code and the datasets are publicly available on the authors web pages.

IV. Network Architectures

To assess the efficacy of the proposed cross-category feature transfer framework, we compare the classification accuracy achieved by the current state-of-the-art pre-trained networks for image classification alongside with the results achieved by two custom networks developed and trained from scratch for the specific task of defective fruits detection.

AlexNet [9] is a network proposed by Krizhevsky *et al.* which won ILSVRC 2012 [10], achieving highest classification performance. AlexNet has 8 layers with learnable weights (5 convolutional layers and 3 fully connected layers). As pros, this network is fast for re-training and classifying new images. On the other hand, the network results large and not as accurate as newer pre-trained models.

GoogleNet [11] main advantages are its simplicity (9 identical and relative simple blocks), the parallelism of the network (each block is structured in 4 parallel pathway) and its efficiency in terms of both computation time and memory usage (it allows GoogleNet to beat both VGG and ResNet execution-time wise). The high efficiency comes at a small cost in term of model accuracy on ILSVRC 2012.

ResNet [12] (Residual Network) is a very deep network proposed by He *et al*. It comes into two variants: ResNet-50 (which counts 50 layers) and ResNet-101 (101 layers deep). The complexity of the networks represents its major drawback. Only the simpler version has been used in the experimental session.

VGG network [13] also comes into two different variants, VGG-16 and VGG-19. The former has 16 layers with learnable weights: 13 convolutional layers and 3 fully connected layers. VGG-19 has 19 layers with learnable weights: 16 convolutional layers and 3 fully connected layers. In both networks, all convolutional layers have filters of size 3-by-3. VGG networks are larger and typically slower than other pretrained networks (in particular with respect to GoogleNet and ResNet). We test both versions in our experimental session.

Finally, we test the **Inception-v3** [14] networks, which is an evolution of the GoogleNet architecture. Compared to GoogLeNet, Inception-v3 is larger, deeper, typically slower, but more accurate on the original ILSVRC data set. Inception-v3 is 48 layers deep.

All the above networks have been trained on the ImageNet database (or on a subset of it).

Furthermore, we propose two novel network structures to tackle our specific case-of-study. A discussion about the reasons that lead us to develop the *ad hoc* networks can be found in Section V, while now we provide some insight on the network architectures. The first proposed architecture counts 6 computational blocks and will be referred as *FruitDef*. The first 4 blocks are convolutional blocks. In particular, odd blocks contain a convolutional layer followed by a *max-pooling* layer (which halves the previous layer dimension) and finally an activation layer (*ReLU*). The even blocks are pretty much the same, but we use *average-pooling* as pooling layer. The fifth block contains only 2 layers: a convolutional layer and a *ReLU* layer. While the last block is a fully connected layer with a *soft max* layer which yields the classification result. The first 3 convolutional layers counts 32 filters of dimension 4×4, while the last two have 64 filters of the same dimension.

	Green	Red
FruitDef	99.868	94.143
FruitDefV2.0	99.736	94.214
AlexNet	100	94.948
VGG16	96.513	94.948
VGG19	100	91.428
GoogleNet	99.861	86.807
ResNet50	99.650	96.190
Inception-v3	97.350	94.413

TABLE II
Performance achieved on the base datasets. The performance achieved on the Green dataset are the best-case-scenario performance (thanks to the high number of samples in the dataset).

The second network (*FruitDefV2.0*) counts 8 macro-blocks. The last two blocks are the same as the simpler version ones, while the first 6 blocks are convolutional blocks. Differently from the network introduced before, only block 2, 3 and 6 contain a pooling layer (average, max and average respectively). In block two, we use two *dropout* layers (at the beginning and at the end of the block) with respectively 35% and 45% dropout percentages. Each block but the last one contains an activation layer (*ReLU*) right after the pooling/convolution layer. Both networks are trained from scratch using as datasets the ones introduced in Section III.

V. Experimental results

Figure 4 shows a schematic representation of the tests that were conducted in order to assess the performance of the proposed approach. The *D*s blocks represent a dataset, while *C*s blocks represent the classifier learned on it. Without loss of generality, a possible strategy is to use a numerous dataset

	Green	Green2Red
FruitDef	88.421	93.684
FruitDefV2.0	90.336	99.370

TABLE III
Transfer learning test on custom networks.

(*i.e. a*) to train a classifier. This classifier can be used as it is in the *strongly related* but less numerous dataset (*i.e. b*). On the other hand, the same network can be re-trained (using transfer learning) in order to improve the performance on the less numerous dataset.

Table I shows classification accuracies obtained with the different datasets on different network architectures. The results we are most interested in are the ones yield by the *ad hoc* networks, which, as you can see, perform slightly worse than the pre-trained networks. In an industrial application like the one requested for our case-of-study, run-time performance are central in the development of the network to be used. Hence, even if the pre-trained networks are able to achieve state-of-the-art performance classification-wise, they are too slow and not practical to use in a real world scenario. This is due to both the fact that the input images are much bigger than the original image dimension (about 3 times bigger) and the network structure of the pre-trained networks are much more complex than the *ad hoc* networks we proposed. The adoption of simpler networks whose input image dimension is equal to the original dimension of the images allows us to perform the classification task an order of magnitude faster (about $0.1ms$) on the machine the tests were performed (3.5 Ghz Intel quad core with nVidia 980 GTX).

The best performance are achieved by the most complex networks, like the Inception-v3. On the other hand, even simpler and faster pre-trained networks achieve similar performance. Simpler networks (architecture-wise) are the one that get the most benefit from the proposed approach. All pre-trained networks show a performance gain using a CDDA approach, even if the mean gain is lower overall with respect to the gain on the custom networks.

Finally, we compare the performance of *ad hoc* networks on two last scenarios. The test involves training from scratch the same networks directly on a dataset of objects of interest (in opposite to general purpose dataset) followed by a transfer learning step on a more specific problem. In particular, we trained the custom networks on both the Green dataset and the Green dataset plus the Green2Red dataset. Then, transfer learning is performed using the olives in the Red dataset. Performance achieved are shown in Table III.

VI. Conclusion

In this paper we proposed a new data augmentation approach in which the additional samples are created by mapping elements from a different pool instead of being generated from the dataset itself. While traditional data augmentation uses a function to generate unseen samples from the available ones, we take advantage of a different dataset which we assume can be mapped to the former by means of an image processing function. We observed that such approach tends to outperform the classical data augmentation, probably because it has better chance to preserve features that are relevant for the classification but not easily embeddable into the generator function. The advantage is particularly visible in small custom CNNs, where the training phase is sensitive to the even distribution of different sample categories.

We tested this approach in a real-world classification scenario of food quality assessment. In our case, one of the two species of olives were far more common than the other so that the direct training was ineffective in practice. Since the two species differ in term of shape and color, but not on the nature of their defect to classify, we investigated a mapping approach instead of the generative one. Results show that, especially for small networks, the former gives better results in almost all the tests.

References

[1] L. S. Yaeger, R. F. Lyon, and B. J. Webb, “Effective training of a neural network character classifier for word recognition,” in *Advances in Neural Information Processing Systems 9*, M. C. Mozer, M. I. Jordan, and T. Petsche, Eds. MIT Press, 1997, pp. 807–816. [Online]. Available: http://papers.nips.cc/paper/1250-effective-training-of-a-neural-network-character-classifier-for-word-recognition.pdf

[2] L. Taylor and G. Nitschke, “Improving deep learning using generic data augmentation,” *CoRR*, vol. abs/1708.06020, 2017. [Online]. Available: http://arxiv.org/abs/1708.06020

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[4] G. Rogez and C. Schmid, “Mocap-guided data augmentation for 3d pose estimation in the wild,” in *NIPS*, 2016.

[5] X. Peng, B. Sun, K. Ali, and K. Saenko, “Learning deep object detectors from 3d models,” in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ser. ICCV ’15. Washington, DC, USA: IEEE Computer Society, 2015, pp. 1278–1286. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2015.151

[6] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?” *CoRR*, vol. abs/1411.1792, 2014. [Online]. Available: http://arxiv.org/abs/1411.1792

[7] E. S. Olivas, J. D. M. Guerrero, M. M. Sober, J. R. M. Benedito, and A. J. S. Lopez, *Handbook Of Research On Machine Learning Applications and Trends: Algorithms, Methods and Techniques - 2 Volumes*. Hershey, PA: Information Science Reference - Imprint of: IGI Publishing, 2009.

[8] J. Deng, W. Dong, R. Socher, L. jia Li, K. Li, and L. Fei-fei, “Imagenet: A large-scale hierarchical image database,” in *In CVPR*, 2009.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS’12. USA: Curran Associates Inc., 2012, pp. 1097–1105. [Online]. Available: http://dl.acm.org/citation.cfm?id=2999134.2999257

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “ImageNet Large Scale Visual Recognition Challenge,” *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1–9.

[12] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[13] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[14] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” *CoRR*, vol. abs/1512.00567, 2015. [Online]. Available: http://arxiv.org/abs/1512.00567