A comparison of graph-based semi-supervised learning for data augmentation

Willian Dihanster G. de Oliveira
Institute of Science and Technology
Federal University of São Paulo
São José dos Campos, Brazil
williandihanster@gmail.com

Otávio A. B. Penatti
Samsung R&D Institute
Campinas, Brazil
o.penatti@samsung.com

Lilian Berton
Institute of Science and Technology
Federal University of São Paulo
São José dos Campos, Brazil
lberton@unifesp.br

***Abstract*—In supervised learning, the algorithm accuracy usually improves with the size of the labeled dataset used for training the classifier. However, in many real-life scenarios, obtaining enough labeled data is costly or even not possible. In many circumstances, Data Augmentation (DA) techniques are usually employed, generating more labeled data for training machine learning algorithms. The common DA techniques are applied to already labeled data, generating simple variations of this data. For example, for image classification, image samples are rotated, cropped, flipped or other operators to generate variations of input image samples, and keeping their original labels. Other options are using Neural Networks algorithms that create new synthetic data or to employ Semi-supervised Learning (SSL) that label existing unlabeled data. In this paper, we perform a comparison among graph-based semi-supervised learning (GSSL) algorithms to augment the labeled dataset. The main advantage of using GSSL is that we can increase the training set by adding non-annotated images to the training set, therefore, we can benefit from the huge amount of unlabeled data available. Experiments are performed on five datasets for recognition of handwritten digits and letters (MNIST and EMINIST), animals (*Dogs vs Cats*), clothes (MNIST-Fashion) and remote sensing images (*Brazilian Coffee Scenes*), in which we compare different possibilities for DA, including the GSSL, Generative Adversarial Networks (GANs) and traditional Image Transformations (IT) applied on input labeled data. We also evaluated the impact of such techniques on different convolutional neural networks (CNN). Results indicate that, although all DA techniques performed well, GSSL was more robust to different image properties, presenting less accuracy variation across datasets.**

I. Introduction

Machine learning (ML) is an area of great importance nowadays, which is based on creating and modeling systems capable of learning from examples and automatically improve with experience. In ML, there is a learning hierarchy that can be divided into supervised, semi-supervised and unsupervised learning [1]. Among the most popular applications of ML, we can cite image classification [2], [3]. An example is the classification of coffee plantations from remote sensing images, which help farmers to better identify areas with coffee farming [4]. However, for this automatic recognition to work in the best possible way, a large amount of data is needed, especially labeled data to train a classifier. Specialized image and video classification tasks often have insufficient data, since obtaining good labels is a difficult and costly task, or the data access is restricted due to privacy concerns.

Several techniques for data augmentation (DA) have been employed to obtain more labeled data in image classification [2], [5], [6]. DA is the process of generating samples by transforming training data, to improve the accuracy and robustness of classifiers. Previous works have demonstrated the effectiveness of DA through simple techniques, such as cropping, rotating, and flipping input images [6]. The choice of the DA strategy is important to reach good accuracy and robustness properties, with a limited number of additional training samples [2]. However, constructing DA schemes which result in simple, fast algorithms and improve the classification results is a non-trivial task, since successful strategies vary with the nature of data present in different applications [7]. Moreover, there are not many comparative studies that show the performance differences of these different augmentations, especially using semi-supervised learning (SSL).

In this work, we employ three DA techniques in image classification: i) four different GSSL techniques to propagate the labels to unlabeled elements; ii) image transformation that expands the dataset by applying effects such as cut, rotate, etc; iii) GANs to create new samples [8]. We carried out an experimental evaluation using features extracted by CNN, i.e., ResNet, VGG16, VGG19, Xception. We analyzed the results using the three DA techniques to understand the advantages and disadvantages of each one in five datasets: digits and letters (MNIST and EMINIST), animals (*Dogs vs Cats*), clothes (MNIST-Fashion) and remote sensing images (*Brazilian Coffee Scenes*). The last one is very challenging since it is composed of satellite images of coffee plantations. The recognition of crop regions in remote sensing images still poses many challenges.

The remaining of the paper is organized as follows: Section II presents related work using DA. Section III briefly presents the DA techniques used in this work. Section IV describes the experimental setup employed in this paper. Section V presents the experimental results and the comparison among different data augmentation scheme, especially using graph-based SSL. Finally, Section VI presents conclusions and future work.

II. Related Work

The performance of the algorithms in supervised learning usually can be improved according to the increase in the number of instances in a database. However, in real-life situations, we

do not always have sufficient examples available especially labeled ones. Data augmentation (DA) consists in transforming the available samples into new samples using label-preserving transformations.

Most DA methods adopted in image classification setups use techniques like cropping, mirroring, color casting, scaling and rotation for creating additional training images. In [5], the authors explored and compared multiple solutions to the problem of data augmentation in image classification. They considered the tiny-imagenet-200 data and MNIST datasets and employed image transformations and GANs to generate images of different styles. They proposed a method to allow a neural network to learn augmentations that best improve the classifier. They conclude that GANs and neural augmentations do not perform much better than traditional augmentations and consume much more computational resources.

In [6], authors considered Alexnet as the pre-training network model and a subset of CIFAR10 and ImageNet (10 categories) were selected as the original dataset. The data augmentation methods used in their paper included: GAN, flipping, cropping, shifting, PCA jittering, color jittering, noise, rotation, and some combinations. Authors found that cropping, flipping, GAN, rotation perform generally better than others.

In [9], authors proposed a greedy strategy that selects the best transformation from a set of candidate transformations resulting in a computationally expensive process. An automated way for finding transformation parameters that lead to increased accuracy and robustness of classifiers was proposed by [2]. They transformed samples by small transformations that induced maximal loss to the current classifier. Then, they performed a simple modification of the Stochastic Gradient Descent (SGD) algorithm to incorporate the proposed DA scheme in the training of deep neural network classifiers.

Different applications have employed DA. In [10], the authors studied augmentation of drone sounds to detect commercial hobby drones in real-life environments and help to detect drones used for malicious purposes. They recorded the sound produced by some popular commercial hobby drones, and then augmented this data with diverse environmental sound data to remedy the scarcity of drone sound data in the diverse environment. The research in [11] explored the use of a DA method for training a deep learning algorithm for gait recognition as biometric information. They generate synthetic video data with 6.5 million frames of real motion capture data, video data and 3D mesh models. Authors [12] used DA in the development of a model that artificially augments a dataset of real images with 2D human pose annotations using 3D Motion Capture data. In [13], the authors proposed a method to create a grid of $n \times n$ cells, in which each cell contains a different randomly rotated image and introduces a natural background in the newly created image. The dataset considered has aerial images of cows and natural scene backgrounds.

Few works presented semi-supervised methods for data augmentation. Authors in [14] proposed an approach that can synthesize large-scale labeled training datasets using 3D graphical engines based on a physically-valid low dimensional pose descriptor. They validated the dataset quality by performing human pose estimation using deep neural network models. Some works used data augmentation to improve semi-supervised learning (SSL) accuracy, like [15] that employed state-of-the-art data augmentation to generate diverse and realistic noise. They presented good results on text and vision datasets.

III. Data Augmentation (DA)

A DA technique is a method capable of augmenting the training dataset while preserving its original label and can be represented as the mapping presented in Equation 1. The fact that DA is based on techniques that preserve the original labels of an example means that if a given x has a label y, then $\phi(x)$ will also have a label y [16].

$$\phi : S \mapsto T \tag{1}$$

where S is the original training set and T is the augmented set from S. Then, the augmented dataset S', containing the original data and the data augmented by the technique ϕ is defined in Equation 2.

$$S' = S \cup T. \tag{2}$$

There are several approaches to generate more data from the training set. Following, some techniques are presented in more detail.

A. Graph-based Semi-Supervised Learning (GSSL)

To acquire a sufficiently large labeled training set is expensive, time-consuming, and often infeasible. Due to the abundance of the unlabeled data available, SSL is interested in how this data can be used to improve the accuracy of the classifiers produced [3], [17], [18]. The work-flow in Figure 1 describes the approach for DA employing SSL. The idea is to use a few labeled data together with a large amount of unlabeled data as input for an SSL algorithm, which will generate augmented labeled data. This augmented set can improve a supervised classifier prediction.

Here, we employed graph-based semi-supervised learning for DA. Graph-based SSL uses two different datasets for training [19]: the set of labeled data $X_l = \{(x_1, y_1), \ldots, (x_l, y_l)\}$ and the set of unlabeled data $X_u = \{x_{l+1}, \ldots, x_{l+u}\}$, where $y \in Y$ corresponds to the data labels, $l \ll u$, $X = l + u$. Given both sets as input, first we construct a graph $G = (V, E, W)$ where each example $x \in X$ corresponds to a vertex $v \in V$. E is the set of edges and W is a matrix with the corresponding weights for each edge. A complete weighted graph can be used or some strategy like k-nearest neighbors (kNN) for graph sparing, where each edge is connected to its k nearest neighbors [20]–[22]. Finally, a label propagation algorithm can run on the graph to spread the known labels to the unlabeled vertices. This way, we can increase the labeled set with the labels predicted by the algorithm: given a small labeled set and an unlabeled set as input, the output is an augmented training set.

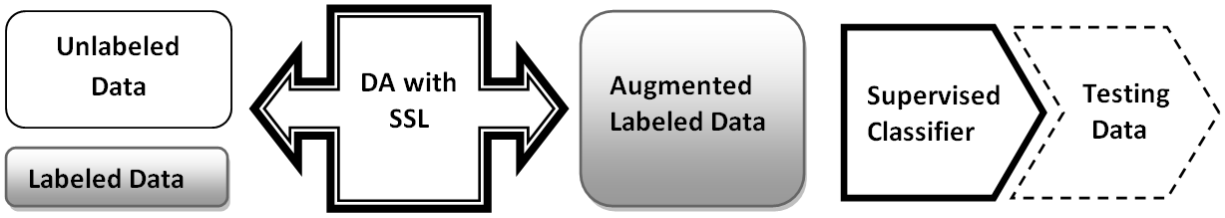

Fig. 1: Work-flow for data augmentation (DA) process using semi-supervised learning (SSL): given a small set of labeled data together with a high amount of unlabeled data, SSL is capable of producing augmented labeled datasets. The augmented labeled data will be used to train some supervised classifier to generate better predictions.

We considered the following GSSL algorithms:

- *Label Propagation* (LP) [23]: it occurs via the iteration of the following:

$$\widehat{Y}^{(t+1)} = P\,\widehat{Y}^{(t)}; \tag{3}$$

  Here, $P_{ij} = \frac{P_{ij}}{\sum_k P_{ik}}$ and $\widehat{Y}^{(0)} = Y = [Y_l, Y_u]^\top$ is the matrix representing the initial labeling.
- *Local and Global Consistency* (LGC) [24]: minimizes the following cost function:

$$\mathcal{Q}(F) = \frac{1}{2}(tr(F^T \mathcal{L} F) + \mu(F - Y^2)) \tag{4}$$

  where $\mathcal{L}$ is the unnormalized graph Laplacian. The parameter $\mu \in (0, \infty)$ controls the trade-off between fitting labels versus enforcing the graph smoothness by minimizing local differences. A matrix $F = \{F_1^T, \ldots, F_n^T\}^T$ corresponds to a classification on the dataset X.
- *Gaussian Fields and Harmonic Functions* (GFHF) [25]: builds upon the initial LP. Here authors show that the derived solution is harmonic, which means that the value of the classifying function f at each unlabeled instance is a weighted average of its neighbors at unlabeled points:

$$f(j) = \frac{1}{d_j} \sum W_{ij} f(i) \tag{5}$$

  for $j = l+1, \ldots, l+u$.
- *OMNI-Prop* (OMNI) [26]: it works by iteratively updating the scores of vertex i holding a label y (represented by a self-score q_{iy}) and the neighborhood of i holds this same label (represented by the follow score δ_{iy}).

$$q_{iy} = \frac{\sum_{j=1}^n A_{ij}\delta_{jy} + \lambda b_y}{\sum_{j=1}^n A_{ij} + \lambda} \tag{6}$$

  and,

$$\delta_{jy} = \frac{\sum_{i=1}^n A_{ij} q_{iy} + \lambda b_y}{\sum_{i=1}^n A_{iy} + \lambda} \tag{7}$$

  where Aij is the ij-element in the adjacency matrix; b_y in each equation is the prior score; and λ is the prior strength parameter, which controls the updates of b_y.

B. Image transformations (IT)

Image transformation is a simple and widely used approach for DA. Here, image transformations are applied to an original labeled image, generating a new set of images with the same label. Thus, an increased dataset is obtained, composed of the original images plus the edited ones. The image transformations are usually divided into two categories: photometric techniques and geometric techniques [16]. Photometrics apply effects such as noise, blur, color effects (e.g., brightness, saturation, color jittering), etc. Geometry uses, for example, rotations, translations, scales, flips, etc.

Geometric transformations are easily implemented and help overcome positional biases. However, there are some applications (e.g., medical) where the biases are more complex than positional and translational variances. We also need to take care of some geometric transformations such as rotation or random cropping to make sure they have not altered the label of the image (ex. 6 versus 9).

C. GANs

Generative modeling refers to the practice of creating artificial instances from a dataset such that they retain similar characteristics to the original set. The GAN architecture first proposed by Ian Goodfellow [8] is a framework for generative modeling through adversarial training. It is composed of a generator $G(z)$, and a discriminator $D(x)$. The main goal of the discriminator is to determine whether the input was sampled from the true distribution $p(x)$, or if was produced by G. The GAN architecture uses a deep learning model (ex. CNN) in the generator and discriminator networks. It is able to produce acceptable images on a simple image dataset such as the MNIST handwritten digits. There have been many new architectures proposed for expanding on the concept of GANs and producing higher resolution output images, such as DCGANs, Progressively Growing GANs, CycleGANs, and Conditional GANs [27]. However, GANs can present some drawbacks, since they require a substantial amount of data to train, depending on how limited the initial dataset is, GANs may not be a practical solution. Depending on the configuration and the application, it can fail to produce quality results for higher resolution, more complicated datasets.

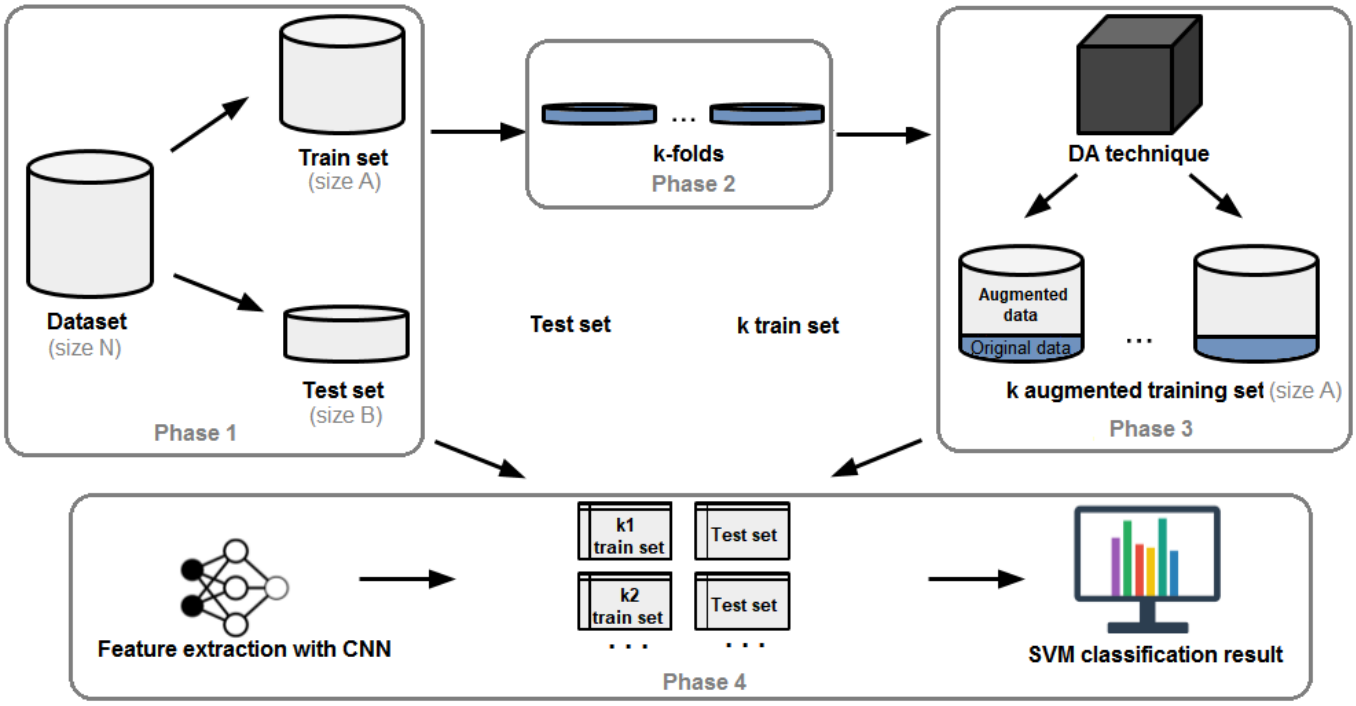

Fig. 2: Work-flow: Phase 1 divides the dataset into train and test sets using holdout (train set gets size A and test set gets size B). Phase 2: divides the train set into k sets containing only 10% of the original data each one. Phase 3: applies different DA techniques into the k folds until the set reaches the original train size A. Phase 4: applies CNN feature extraction into the k training sets and the original test set, and then performs SVM classification to compare the results.

IV. Experimental Setup

Figure 2 summarizes the methodology used in this work, which can be divided into four phases presented below.

- Phase 1: Each dataset will be randomly separated between train and test sets using holdout sampling. Therefore, the training set will have size A and the test set will have a size B.
- Phase 2: The train set will be divided into $k = 5$ disjoint sets, each one considering only 10% of the train set.
- Phase 3: Finally, each of these k sets will be applied to a different DA technique, generating new data until the set reaches the original size of the train set.
- Phase 4: The augmented training sets go through classification experiments, with the test set defined at the beginning. For each augmented set, the SVM classifier is applied. In the end, each technique will be evaluated and compared, calculating the average results.

A. DA generation

In Phase 3, three DA techniques are employed:

- Graph-based Semi-supervised Learning (GSSL): From each training set with 10% of labeled data randomly selected, a GSSL algorithm is applied to label the other 90% of the unlabeled data. This set with predicted labels is used as an augmented set. The kNN approach was employed to construct the graphs. We considered $k = 5$.
- Image Transformation (IT): From each training set with 10% of labeled data randomly selected, IT techniques are applied to generate increased data until reaching the original training data size. We used the following IT: rotation, cut, translation, zoom, flip and fill.
- Generative Adversarial Networks (GANs): From each training set with 10% of labeled data randomly selected, a GAN is applied. We considered the HyperGAN[1]. Some of the main default parameters of the implementation are as follows, we only change the number of iterations, batch size, and image size:
  - Discriminator D: 4 layers, $tanh$ as final activation, 1 fully connected layer;
  - Generator G: 64 as final depth, $tanh$ as final activation;
  - Training: learning rate D (ηD) = 0.0001, learning rate G (ηG) = 0.0001, 5,000 iterations, *batchsize* = variable with the size of the images, image dimension = same as the dataset considered;

[1]https://github.com/HyperGAN/HyperGAN

B. Feature extraction with CNN

For Phase 4, we used different convolutional neural networks (CNNs) pre-trained on ImageNet as feature extractors [28]. We used as feature vectors, the output of the last layer before the classification layer. The CNNs considered here were ResNet50, VGG16, VGG19, and Xception:

- VGG16 and VGG19: are CNNs with 16 and 19 *weight layers*, respectively. There are 13 convolution layers for VGG16 and 16 for VGG19, with 3 fully connected layers. The *pooling* is of type *max pooling*. The feature vector generated by the network has dimensionality 1×4096.
- ResNet50: is a CNN with 50 *weight layers*, being 49 convolution and 1 fully connected layer. The *pooling* is of type *global average pooling*. The feature vector generated by the network has a dimensionality of 1×2048.
- Xception: is a CNN with 36 layers structured in 14 modules. The *pooling* is of type *max pooling* and *global average pooling*. The feature vector generated by the network has dimensionality 1×4096.

C. Evaluation

The k augmented sets with their features extracted by a CNN are used for the SVM classification experiments with validation by the original test set. Then, with the results, the accuracy measure and the standard deviation were calculated for further comparison and evaluation among all approaches. Each result considers the average of the k augmented training sets generated by each DA technique and for each CNN, given a dataset.

D. Implementation and execution environment

We used the TensorFlow[2] and Keras[3] libraries both for feature extraction with the CNNs and for applying IT. The scikit-learn[4] for SVM classifier and NumPy[5] for vectors and arrays.

The experiments were performed on a notebook with Intel I5 processor 7GHz 2.5GHz, 8GB RAM and NVIDIA GEFORCE 920MX GPU with Windows10 and Python 3.7.

E. Datasets

Brazilian Coffee Scenes [4]: this dataset is composed of remote sensing images of coffee plantations in cities of Minas Gerais, Brazil. The dataset consists of 2876 examples (50% of the class *coffee* and 50% of the class *non coffee*). We divided into train and test sets with 70% and 30% of data, respectively. Thus, the train set had 2014 examples and the test set had 862 examples, both sets had 50% from each class. Figure 3 shows some examples of the images captured by the sensor.

Dogs vs Cats [29]: this dataset contains images of dogs and cats. We considered 10,000 training examples and 2,000 test examples, both sets had 50% from each class. Figure 4 presents some examples.

[2]https://www.tensorflow.org/
[3]https://keras.io/
[4]https://scikit-learn.org/stable/
[5]https://www.numpy.org/

Fig. 3: Example of coffee (a - b) and non-coffee (c - d) images from *Brazilian Coffee Scenes* dataset [4].

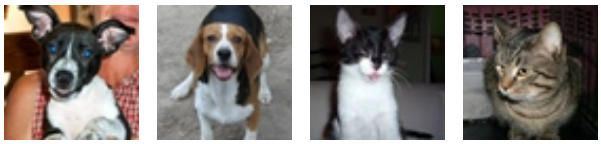

Fig. 4: Example of *dogs vs cats* dataset [29].

MNIST [30]: This dataset consists of images of handwritten digits. Here, we only used examples of the digits 3 and 8, with 7,000 examples in total. We divided into train and test sets with 70% and 30% of data, respectively. Thus, the train set had 5,600 examples and the test set had 1,400 examples, both sets had 50% from each class. Figure 5 shows some examples.

Fig. 5: Examples of MNIST dataset [30].

Extend MNIST - EMNIST [31]: this dataset contains images made up of handwritten digits and letters. We only used examples of the letters "D", "G", "O" and "Q" (which bear some resemblance and would be a challenge for the classifier), with 5,600 examples of each class and a total of 22,400 examples. We use the train and test partition from the author, thus the train set has 19,200 examples and the test set has 3,200 examples. Figure 6 shows some examples of the images.

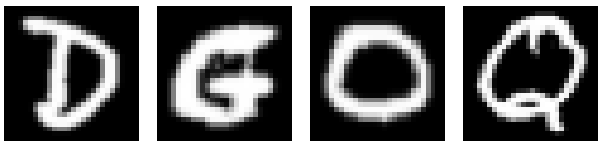

Fig. 6: Examples of EMNIST dataset [31].

MNIST-Fashion [32]: this dataset contains images in gray levels associated with 5 classes: ankle boot, coat, dress, shirt, and sneaker, presented in Figure 7. Each class has 7,000 examples totaling 35,000 examples. We use the train and test partition from the author, thus the train set has 30,000 examples and the test set has 5,000 examples.

V. Results

This section presents the results comparing the DA techniques Graph-based Semi-Supervised Learning (GSSL) - con-

TABLE I: Classification accuracy considering different DA techniques in five different datasets. All the DA techniques generated 90% of the training set using as source only 10% of the original training set. As a baseline, the last column shows results when the whole training set was used (no data augmentation). We can observe that the DA techniques perform very well, in many cases obtaining accuracy values very close to the baseline. GSSL techniques also presented good results.

	IT	GSSL(LP)	GSSL(LGC)	GSSL(GFHF)	GSSL(OMNI)	HyperGAN	Baseline
Brazilian Coffee Scenes							
ResNet50	**87.35 ± 1.79**	79.79 ± 1.26	81.33 ± 2.19	84.04 ± 1.17	82.54 ± 1.08	-	88.59 ± 1.05
VGG16	79.42 ± 2.27	68.59 ± 1.73	70.92 ± 1.99	72.95 ± 3.93	66.93 ± 3.77	-	85.50 ± 2.57
VGG19	81.06 ± 2.69	72.01 ± 4.36	68.15 ± 4.98	76.47 ± 2.35	68.82 ± 2.15	-	85.71 ± 1.98
Xception	77.53 ± 2.10	71.62 ± 0.98	66.79 ± 0.47	73.42 ± 0.56	72.84 ± 0.66	-	82.3 ± 0.78
Dogs vs Cats							
ResNet50	97.85 ± 0.14	98.07 ± 0.17	**98.46 ± 0.05**	98.37 ± 0.05	98.36 ± 0.07	-	99.20 ± 0.11
VGG16	97.47 ± 0.26	97.46 ± 0.09	98.19 ± 0.13	98.18 ± 0.14	98.19 ± 0.16	-	98.81 ± 0.14
VGG19	97.6 ± 0.15	97.55 ± 0.09	98.06 ± 0.23	97.87 ± 0.09	97.82 ± 0.08	-	98.63 ± 0.08
Xception	65.34 ± 0.62	55.63 ± 0.86	60.82 ± 0.78	62.33 ± 0.70	62.26 ± 1.19	-	95.79 ± 0.65
MNIST							
ResNet50	97.79 ± 0.37	97.2 ± 0.16	**98.84 ± 0.22**	98.38 ± 0.19	98.23 ± 0.12	98.09 ± 0.28	99.04 ± 0.36
VGG16	95.31 ± 0.37	96.59 ± 0.19	98.13 ± 0.296	97.69 ± 0.09	97.79 ± 0.26	96.98 ± 0.29	98.38 ± 0.26
VGG19	96.07 ± 0.47	96.95 ± 0.16	98.33 ± 0.14	97.47 ± 0.16	97.37 ± 0.14	97.58 ± 0.17	98.50 ± 0.29
Xception	90.91 ± 0.88	75.87 ± 4.36	91.33 ± 1.38	91.23 ± 0.46	89.66 ± 1.29	93.03 ± 0.37	96.17 ± 1.95
EMNIST							
ResNet50	83.48 ± 0.72	81.45 ± 0.37	82.37 ± 0.41	81.58 ± 0.63	81.45 ± 0.33	83.71 ± 0.62	88.78 ± 0.59
VGG16	83.91 ± 0.17	82.05 ± 0.41	82.73 ± 0.55	82.33 ± 0.42	82.21 ± 0.35	84.3 ± 0.47	88.78 ± 0.28
VGG19	83.85 ± 0.34	80.55 ± 0.22	82.27 ± 0.32	81.8 ± 0.26	80.57 ± 0.15	**84.44 ± 0.43**	89.22 ± 0.47
Xception	74.28 ± 0.64	63.77 ± 0.87	69.07 ± 0.69	69.1 ± 0.51	68.45 ± 0.47	75.31 ± 0.45	82.72 ± 0.46
MNIST-Fashion							
ResNet50	87.58 ± 0.16	87.40 ± 0.13	**87.80 ± 0.26**	87.11 ± 0.15	86.4 ± 0.22	87.68 ± 0.26	91.36 ± 0.36
VGG16	85.64 ± 0.07	85.23 ± 0.15	86.80 ± 0.17	86.08 ± 0.20	85.72 ± 0.55	86.85 ± 0.31	90.28 ± 0.30
VGG19	86.96 ± 0.25	83.49 ± 0.45	85.90 ± 0.32	84.73 ± 0.31	84.12 ± 0.64	86.6 ± 0.44	90.04 ± 0.24
Xception	81.12 ± 0.26	75.63 ± 0.81	81.79 ± 0.31	80.88 ± 0.28	79,65 ± 0.36	82.77 ± 0.36	87.32 ± 0.36

Fig. 7: Examples of MNIST-Fashion dataset [32].

sidering the algorithms LP, LGC, GFHF and OMNI, Image Transformation (IT) and HyperGAN.

Table I presents the average classification results (with standard deviation) for each DA technique and each dataset. We also show the results considering each CNN used as a feature extractor. As a baseline, we performed experiments with the complete original training set (i.e., no data augmentation; similar to executing the workflow of Figure 2 with only Phases 1 and 4). The baseline results are shown in the last column of Table I and it represents a possible upper bound for the classification results because in this case, we had access to the original labels of the training set.

We could observe that for some datasets, the use of DA techniques could achieve very close results to the baseline, indicating that the DA technique could generate or label good samples for training the classifier. For the *Brazilian Coffee Scenes* dataset, we noticed that image transformation (IT) performed better than the other DA techniques, for any of the CNNs used. For Dogs vs Cats dataset, GSSL techniques, especially LGC, performed better although other DA techniques presented similar accuracies. For MNIST, GSSL-LGC also showed higher accuracy values. For EMNIST, the HyperGAN presented slightly higher accuracy values, indicating that the generated handwritten letters were effective in enlarging the training set. For MNIST-Fashion, many DA techniques performed similarly. In general, one conclusion is that, although the differences among the DA techniques tested were not large, GSSL techniques, especially LGC, could better handle the task of enlarging a training dataset. GSSL obtained this effective augmentation by labeling unlabeled data with similar distribution to the original dataset. ITs, although very simple, also performed well in most of the cases.

In our experiments, the employed HyperGAN did not converge for *Brazilian Coffee Scenes* and *dog vs cat* datasets (see Figures 8 and 9). Coffee plantation recognition, in general, is difficult because it is usually cultivated in mountainous regions. This causes shadows and distortions in the spectral information, which makes the recognition difficult. Moreover, in a very specific scenario such as recognition of coffee regions in remote sensing, it is difficult to guarantee the generated images will be of the expected classes. Expert supervision may be required and this fact is not desirable when using data augmentation techniques. In the dataset of dogs and cats, one possible reason for HyperGAN to not converge is that more images may be necessary.

In addition, we applied and evaluated CNN as image feature extractors. Table II presents a comparison of CNN architecture details (network size, number of parameters and number of attributes it generates), as well as features file size generated

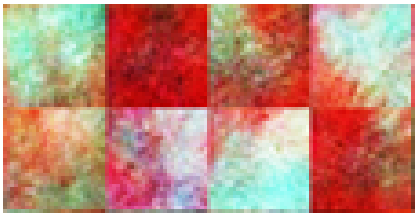

Fig. 8: Examples of images generated by HyperGAN for *Brazilian Coffee Scenes* dataset.

Fig. 9: Examples of images generated by HyperGAN for *Dog vs Cat* dataset.

and the time spent. In this case, the last two factors were evaluated with the MNIST-Fashion dataset, as it is the largest used in this work (30,000 examples). Xception is the network with a smaller size, smaller number of parameters and its features occupy less space in relation to the others. ResNet50, in turn, generates fewer attributes, took less time and has the 2nd smallest size and number of parameters. VGG16 and VGG19 are the networks with a larger size, larger number of parameters (almost 6x more), occupy more storage and spent more time.

The ResNet50, VGG16 and VGG19 networks did well for almost every experiment, with ResNet50 having the best results overall. Xception, in turn, got the worst results. Depending on the application of interest, one important decision factor for selecting a CNN is related to computational complexity. In our experiments, ResNet50 presented the highest accuracy in many cases and a good performance related to the other CNNs tested, both in terms of parameters, processing time, and feature dimensionality.

VI. Conclusion

In this work, we evaluated the effectiveness of different data augmentation techniques for enlarging training datasets for supervised learning scenarios. Additionally, to commonly used data augmentation techniques, we evaluated graph-based semi-supervised learning (GSSL) techniques and concluded that GSSL performed very well in our experiments. Our experimental scenario considered using only 10% of the labels of the original training set and applying the data augmentation techniques to label or to generate the remaining 90% of the training set. GSSL techniques presented high accuracy values and presented less variation in performance across the different datasets tested, showing good robustness to the different image properties. In general, LGC had a better performance since it uses a hyperparameter that controls the trade-off between considering the initial labels and the smooth concerning the graph.

As future work, we would like to evaluate other techniques to generate the graph in GSSL and other semi-supervised approaches. We also envision opportunities for evaluating other GANs. We also believe that GSSL can be more robust in scenarios in which the training set is enlarged by adding examples from other datasets and domains, because more data variability can be included in the training set, consequently making the classifier more robust to real-world situations.

VII. Acknowledgements

This study was financed in part by the National Council for Scientific and Technological Development (CNPq) and Sao Paulo Research Foundation (FAPESP) grant 2018/01722-3.

References

[1] O. Chapelle, B. Schlkopf, and A. Zien, *Semi-Supervised Learning*, 1st ed. The MIT Press, 2010.

[2] A. Fawzi, H. Samulowitz, D. Turaga, and P. Frossard, "Adaptive data augmentation for image classification," in *IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3688–3692.

[3] J. P. K. Catunda, A. T. d. Silva, and L. Berton, "Car plate character recognition via semi-supervised learning," in *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*, 2019, pp. 735–740.

[4] O. A. Penatti, K. Nogueira, and J. A. dos Santos, "Do deep features generalize from everyday objects to remote sensing and aerial scenes domains?" in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2015, pp. 44–51.

[5] J. Wang and L. Perez, "The effectiveness of data augmentation in image classification using deep learning," in *arXiv preprint*, 2017, p. arXiv:1712.04621v1.

[6] J. Shijie, W. Ping, J. Peiyi, and H. Siping, "Research on data augmentation for image classification based on convolution neural networks," in *Chinese Automation Congress (CAC)*, 2017, pp. 4165–4170.

[7] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell, "Learning to classify text from labeled and unlabeled documents," in *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, ser. AAAI, 1998, pp. 792–799.

[8] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS. MIT Press, 2014, p. 2672–2680.

[9] M. Paulin, J. Revaud, Z. Harchaoui, F. Perronnin, and C. Schmid, "Transformation pursuit for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 3646–3653.

[10] S. Jeon, J.-W. Shin, Y.-J. Lee, W.-H. Kim, Y. Kwon, and H.-Y. Yang, "Empirical study of drone sound detection in real-life environment with deep neural networks," p. arXiv:1701.05779, 2017.

[11] C. C. Charalambous and A. A. Bharath, "A data augmentation methodology for training machine/deep learning gait recognition algorithms," in *arXiv preprint*, 2016, p. arXiv:1610.07570.

[12] G. Rogez and C. Schmid, "Mocap-guided data augmentation for 3d pose estimation in the wild," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS, 2016, pp. 3116–3124.

[13] E. Okafor, R. Smit, L. Schomaker, and M. Wiering, "Operational data augmentation in classifying single aerial images of animals," in *IEEE International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, 2017, pp. 354–360.

[14] S. Liu and S. Ostadabbas, "A semi-supervised data augmentation approach using 3d graphical engines," in *In: Leal-Taixé L., Roth S. (eds) Computer Vision – ECCV 2018 Workshops*, 2019, p. 11130.

[15] Q. Xie, Z. Dai, E. H. Hovy, M. Luong, and Q. V. Le, "Unsupervised data augmentation," *CoRR*, vol. abs/1904.12848, 2019.

TABLE II: CNN parameters comparison.

	ResNet50	VGG16	VGG19	Xception
Network size (MB)	98	528	549	**88**
Parameters	25,636,712	138,357,544	143,667,240	**22,910,480**
Attributes	**2048**	4096	4096	4096
Storage (GB)	0.977247	1.255253	1.204083	**0.509851**
Time (minutes)	**75**	122	124	111

[16] L. Taylor and G. Nitschke, "Improving deep learning with generic data augmentation," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2018, pp. 1542–1547.

[17] L. Berton, T. de Paulo Faleiros, A. Valejo, J. Valverde-Rebaza, and A. de Andrade Lopes, "Rgcli: Robust graph that considers labeled instances for semi-supervised learning," *Neurocomputing*, vol. 226, pp. 238–248, 2017.

[18] B. K. de Aquino Afonso and L. Berton, "Identifying noisy labels with a transductive semi-supervised leave-one-out filter," *Pattern Recognition Letters*, 2020.

[19] X. Zhu, "Semi-supervised learning literature survey," University of Wisconsin-Madison, Tech. Rep. 1530, 2005, computer Sciences.

[20] L. Berton, A. de Andrade Lopes, and D. A. Vega-Oliveros, "A comparison of graph construction methods for semi-supervised learning," in *2018 International Joint Conference on Neural Networks (IJCNN)*, July 2018, pp. 1–8.

[21] D. A. Vega-Oliveros, L. Berton, A. M. Eberle, A. de Andrade Lopes, and L. Zhao, "Regular graph construction for semi-supervised learning," *Journal of Physics: Conference Series*, vol. 490, no. 1, p. 012022, 2014.

[22] L. Berton and A. d. A. Lopes, "Graph construction based on labeled instances for semi-supervised learning," in *2014 22nd International Conference on Pattern Recognition*, 2014, pp. 2477–2482.

[23] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," Citeseer, Tech. Rep., 2002.

[24] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *Advances in neural information processing systems*, 2004, pp. 321–328.

[25] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of the 20th International conference on Machine learning (ICML)*, 2003, pp. 912–919.

[26] Y. Yamaguchi, C. Faloutsos, and H. Kitagawa, "Omni-prop: Seamless node classification on arbitrary label correlation," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[27] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, Jul 2019.

[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, 2009, pp. 248–255.

[29] Kaggle. (2015) Dogs vs. cats. [Online]. Available: https://www.kaggle.com/c/dogs-vs-cats/data

[30] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[31] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "Emnist: an extension of mnist to handwritten letters," *arXiv preprint arXiv:1702.05373*, 2017.

[32] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.