Data Augmentation via Variational Auto-Encoders

Unai Garay-Maestre[1], Antonio-Javier Gallego[1](✉), and Jorge Calvo-Zaragoza[2]

[1] Department of Software and Computing Systems,
University of Alicante, Alicante, Spain
ugm2@alu.ua.es, jgallego@dlsi.ua.es

[2] PRHLT Research Centre, Universitat Politècnica de València, Valencia, Spain
jcalvo@prhlt.upv.es

Abstract. Data augmentation is a widely considered technique to improve the performance of Convolutional Neural Networks during training. This step consists in synthetically generate new labeled data by perturbing the samples of the training set, which is expected to provide more robustness to the learning process. The problem is that the augmentation procedure has to be adjusted manually because the perturbations considered must make sense for the task at issue. In this paper we propose the use of Variational Auto-Encoders (VAEs) to generate new synthetic samples, instead of resorting to heuristic strategies. VAEs are powerful generative models that learn a parametric latent space of the input domain from which new samples can be generated. In our experiments over the well-known MNIST dataset, the data augmentation by VAEs improves the base results, yet to a lesser extent of that obtained by a well-adjusted conventional data augmentation. However, the combination of both conventional and VAE-guided data augmentations outperforms all the results, thereby demonstrating the goodness of our proposal.

Keywords: Data augmentation · Variational auto-encoders · Convolutional Neural Networks · MNIST dataset

1 Introduction

Supervised learning is the most considered approach for addressing automatic classification tasks. It is based on learning from a series of correct input-output pairs, from which a model is built with the aim of generalizing to correctly classify unseen inputs.

Convolutional Neural Networks (CNNs) have been one of the biggest breakthroughs of supervised classification [5], especially in the fields of computer vision and image processing. These networks allow learning a hierarchy of features suitable for the recognition task by means of a series of stacked convolutional layers. Although these networks were initially proposed decades ago, several factors have contributed to their eventual success [1].

R. Vera-Rodriguez et al. (Eds.): CIARP 2018, LNCS 11401, pp. 29–37, 2019.
https://doi.org/10.1007/978-3-030-13469-3_4

Within these factors, data augmentation has become a *de facto* standard to improve the learning process [4,6]. It is a step focused on generating a set of synthetic samples out of those in the training set. The intention of this process is twofold: (i) since these neural networks need to be trained on a large set of data, data augmentation might boost the performance by increasing the size of the original training set, (ii) if the augmentation procedure creates examples that mimic expected distortions, the CNN might be more robust to variations at test stage. There are several ways to do data augmentation, especially for images (rotation, color variation, random occlusions, etc.), although the goodness of each one is strongly dependent on the task at issue. Many augmentations can be combined to produce a higher number of new images.

Instead of resorting to hand-crafted procedures, this work proposes a learning-driven approach for the data augmentation stage by means of Variational Auto-Encoders (VAE) [3]. VAEs are powerful generative models that estimate a parametric distribution of the input domain from data. This allows us to generate synthetic samples that fit such distribution. Data augmentation needs to be adjusted manually to select a set of specific augmentations that are suitable to predict variations at the test stage. Nevertheless, a VAE is expected to learn these variations among input samples by itself, thereby offering a greater generalization to any type of classification task. Our experiments demonstrate the goodness of this approach on the MNIST dataset, improving the results obtained with the original training set and demonstrating its complementarity with conventional data augmentation techniques.

The rest of the paper is organized as follows: the proposed approach is elaborated in Sect. 2, our experimental results are presented in Sect. 3, and the main conclusions of our work are summarized in Sect. 4.

2 Method

2.1 Variational Auto-Encoders

Auto-Encoders (AE) are neural networks with an encoder-decoder structure [2,8]. Traditionally, the encoder takes the input and converts it into a smaller, dense representation, from which the decoder converts the input back. Depending on the size of the intermediate representation, the encoder has to learn to preserve as much of the relevant information as possible in the limited space, and intelligently discard irrelevant parts. The space in which the encoding projects the input is usually called *latent space*. Typically, the latent space of a conventional AE does not follow any constraint, and therefore it is difficult to interpret.

Variational Auto-Encoders (VAEs) follow the same topology of that of an AE, but the latent space they consider is forced to fit a parametric distribution [7], allowing easy random sampling and interpolation. Typically, this is achieved by forcing the latent space to behave as a normal distribution. Therefore, the encoder must yield two representations, instead of one: a vector of means, μ, and another vector of standard deviations, σ.

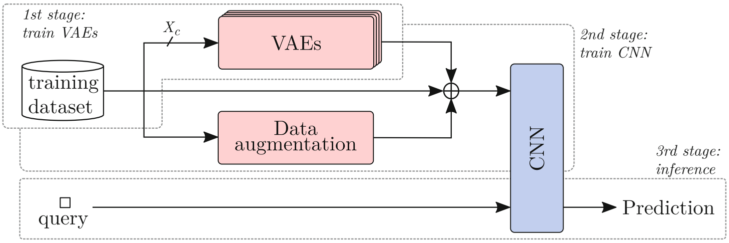

Fig. 1. General outline of the proposed methodology.

Two additional considerations are necessary for training a VAE. On the one hand, the loss function includes the minimization of a divergence between the distribution defined by μ and σ and the chosen distribution for the latent space. On the other hand, the decoder does not operate over the latent space itself, but its parameters are used to generate a random vector that follows the defined distribution. Therefore, the decoder must learn to reconstruct the inputs from sampled values of the distribution estimated by the encoder. This is known as the "re-parameterization trick".

As the latent space samples are somehow generated from the distribution defined by μ and σ, the decoder learns to not just decode single, specific points of the latent space, but the distribution itself. Once trained, decoding sampled vectors from the learned distribution should generate new images that fit within the distribution of the input domain, thus behaving as a generator of samples.

In this work we will train a different VAE per class, and so ensuring that each VAE generates samples that belong to the class that it has been provided during its training. Therefore, the generated samples can be reliably labeled for the classification task.

2.2 Methodology

Figure 1 shows an outline of the methodology proposed in this work. The process consists of three stages: first, different VAEs are trained for every class on the dataset in order to independently model the variations of each class. Once trained, new samples of each class can be created by sampling the latent space distribution. In the second stage, a CNN is trained with the samples generated by the VAEs and/or conventional data augmentation. In the last stage, the trained CNN is able to make predictions about the test samples.

3 Experiments

This section describes the experiments carried out to measure the goodness of the proposed approach.[1]

[1] For the sake of reproducible research, the code of the experiments is available at http://github.com/ugm2/DataAugmentation_VAE.

3.1 MNIST Dataset

The experimentation has been carried out using the MNIST dataset of handwritten digits (10 classes). Originally, this dataset is split into two parts: 60,000 samples of training data and 10,000 samples of test data. The training partition is used both to train the VAEs and the CNN. In order to measure the impact of our proposal, we consider reduced training sets. In particular, we consider training set of sizes 50, 100, 250, 500, and 1,000. Each of these sizes represent the total images, i.e. for the size of 50 only 5 samples per digit will be used. For the case of the VAEs, as there is one for every class of the dataset, a tenth of the amounts are used to train every class-wise VAE. From the training partition, 85% is used to train the VAEs, while the remaining 15% is used as validation to know when to stop. The evaluation part is performed with 700 images of each class (7,000 in total).

3.2 Architectures

Table 1 shows the architecture used for the VAEs and the CNN. The hidden layer of the VAE (marked with (*)) refers to two separated fully connected layers of the size of the latent space: one representing the mean vector (μ) and the other the standard deviation vector (σ). The lambda (λ) layer of the VAE (marked with (**)) is used to sample a vector with the dimensionality of the latent space, following the actual values of μ and σ. The dimensionality of the latent space will be studied empirically.

Table 1. VAE and CNN architectures. Notation: Conv(f, $w \times h$) stands for a layer with f convolutional operators of size $w \times h$; ConvT(f, $w \times h$) stands for a layer with f transposed convolutional operators of size $w \times h$; MaxPool($w \times h$) stands for the Max-Pooling operator with a $w \times h$ kernel; Drop(d) refers to Dropout with ratio d; FC(n) is a Fully-Connected layer with n neurons; LS denotes the dimensionality of the latent space.

Network	Part	Configuration			
VAE	Encoder	Conv(1, 2 × 2)	Conv(64, 2 × 2)	Conv(64, 3 × 3)	Conv(64, 3 × 3)
	Hidden	Flatten()	FC(128)	FC(LS[μ, σ])*	λ(sampling([μ, σ])**

	Decoder	FC(128)
		FC(12544)
		Reshape(14 × 14 × 64)
			ConvT(64, 3 × 3)
			ConvT(64, 3 × 3)
			ConvT(64, 3 × 3)
				Conv(1, 2 × 2)	

CNN	–	Conv(64, 3 × 3)
		Conv(128, 3 × 3)
		Ma × Poo(2 × 2)
		Drop(0.5)
			Conv(128, 3 × 3)
			Conv(128, 3 × 3)
			Ma × Pool(2 × 2)
			Drop(0.5)
				Flatten()
				FC(128)
				Drop(0.5)
				FC(10)	

3.3 Training

3.3.1 VAE

For the training of the VAEs it has been employed the RMSprop optimizer, which uses the magnitude of recent gradients to normalize the gradients. The loss function consists of two terms: the binary cross-entropy and the Kullback-Leibler (KL) divergence. The first one evaluates "how wrong" the output of the decoder (y) matches the input of the encoder ($\hat{y}$). It is calculated as:

$$-\frac{1}{N}\sum_{i=1}^{n} y_i \cdot \log(\hat{y_i}) + (1 - y_i) \cdot \log(1 - \hat{y_i}) \tag{1}$$

The KL divergence measures the difference between $\mathcal{N}(0,1)$ and $\mathcal{N}(\mu,\sigma)$. It is computed as:

$$\sum_{i=1}^{n} \sigma_i^2 + \mu_i^2 - \log(\sigma_i) - 1 \tag{2}$$

The number of epochs used for training the VAEs has been adjusted manually according to the size of the initial training set.

3.3.2 CNN

For the training of the CNN, the Adam gradient descent optimization algorithm has been employed with a categorical cross entropy loss function. The training process was monitored using early stopping, which stops the training process if the validation loss of the training does not decrease after 10 epochs. Once the training process is stopped, the model of the epoch with the best validation loss is chosen.

For the use of conventional data augmentation during the training of the CNN, the following transformations of the data were applied: rotation range of 20°, width shift range of 20%, and height shift range of 20%.

3.4 Results

In this section, we both analyze the generative power of the VAEs and the results of the proposed methodology. The classification performance metric considered in this work is the F_1 score. This metric is defined as the harmonic mean of the precision and the recall, and it properly summarizes the classification performance.

First, we show in Fig. 2 some examples of the digits that have been generated by the VAEs trained with 50 images each, and with varying sizes of the latent space. It seems that the digits generated when considering a latent space of 3 dimensions are the most realistic ones.

Figure 3 shows the effect of applying different types of transformations during the data augmentation process. The types of transformations applied go gradually from a possible lack of expert supervision (applying all the transformations possible) to suitable changes for the MNIST dataset. It has been used different

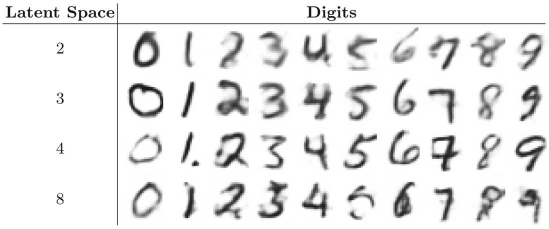

Fig. 2. Generated digits using VAEs with different latent space sizes.

levels of data augmentation adjustment to observe that in order to improve over the CNN without data augmentation (red line), it needs expert knowledge about which perturbations to do on the dataset at issue, as it could worsen the results otherwise.

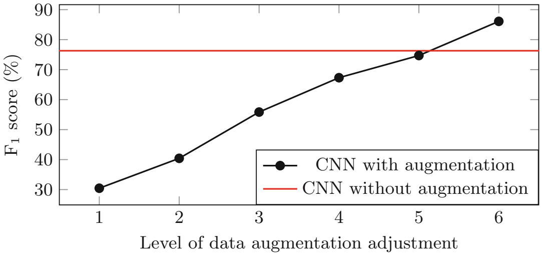

Fig. 3. Comparison of the improvement obtained by gradually adjusting the transformations applied in the data augmentation process from inexpert hands to suitable changes for the corresponding dataset.

The final classification experiments are shown in Table 2, including the CNN without any augmentation method (CNN), using standard data augmentation (AUG), using the generated digits from VAEs (VAE), and using both standard data augmentation along with the digits of the VAEs (AUG + VAE).

At first sight, it turns out that the results with the VAE-generated data remarkably improves the training with the original data; however, the data augmentation process boosts the performance even more, as it has been manually adjusted to the MNIST dataset. Furthermore, considering both data augmentation and the generated samples from the VAEs, as well as the original dataset, the best figures are generally attained, improving the results of just considering data augmentation in most of the cases.

It is important to emphasize that our approach does work with limited training data. For instance, starting from 50 images as initial training set, the result of data augmentation combined with VAE-generated data from a latent space of 3 dimensions, achieves the outstanding result of almost 91% of F_1 score, which

Table 2. Results of the experiments performed: no augmentation method (CNN), standard data augmentation (AUG), digits generated from VAEs (VAE), and using both standard data augmentation and digits generated from VAEs (AUG+VAE)

Latent Space	Training Size	CNN	VAE	AUG	AUG+VAE
2	50	76.30	84.89	86.10	**89.05**
	100	84.38	90.90	**94.85**	94.50
	250	93.03	94.84	97.12	**98.00**
	500	94.28	95.65	98.11	**98.22**
	1000	96.54	97.24	98.36	**98.87**
3	50	76.30	85.40	86.10	**90.86**
	100	84.38	91.98	94.85	**95.15**
	250	93.03	95.93	97.12	**97.97**
	500	94.28	96.38	98.11	**98.26**
	1000	96.54	97.74	98.36	**98.87**
4	50	76.30	83.77	86.10	**89.67**
	100	84.38	91.75	94.85	**94.73**
	250	93.03	95.16	97.12	**97.86**
	500	94.28	96.28	98.11	**98.30**
	1000	96.54	97.38	98.36	**98.90**
8	50	76.30	84.46	86.10	**89.32**
	100	84.38	91.28	**94.85**	94.56
	250	93.03	95.24	97.12	**97.86**
	500	94.28	95.95	98.11	**98.41**
	1000	96.54	97.50	98.36	**98.79**

Table 3. Results obtained for the statistical significance tests comparing our approach with the other methods evaluated. Symbols ✓ and ✗ state that results achieved by elements in the rows significantly improve or decrease, respectively, to the results by the elements in the columns. Significance has been set to $p < 0.01$.

	CNN	VAE	AUG
VAE	✓	–	✗
AUG	✓	✓	–
AUG + VAE	✓	✓	✓

increases the result of the original dataset by 14.56% and the result of the conventional data augmentation by 4.76%.

The dimensionality of the latent space set to 3 seems to give the best results overall, being settled down as the sweet spot for this dataset in concrete. This confirms what was already observed, visually, in Table 2.

In order to draw more robust conclusions from the results obtained, statistical significance tests are performed between the different configurations, taking into account the results for the different sizes of the training set. Specifically, Wilcoxon signed-rank tests are considered, which compare the different approaches by pairs. Table 3 reports the outcomes of these tests. It can be observed that the statistical significance is directly related to the average results obtained, and therefore the conclusions drawn from Table 2 have a proper statistical significance.

4 Conclusions

A learning-driven approach for data augmentation has been proposed. It considers Variational Auto-Encoders (VAEs), which can be used to generate new samples after being trained to model the input domain of a specific class of the classification task.

Our experiments with the MNIST dataset has reported very promising results. It has been shown that including the samples generated by the VAEs in the training set leads to a better performance compared to that of just using the initial training set. Although using conventional data augmentation improves the actual accuracy even more, it should be noted that our approach does not need to be manually adjusted. In addition, the combination of traditional data augmentation with the samples generated by the VAEs provides the best overall results.

This work has been restricted to the MNIST dataset, and so the first avenue to explore is to study this approach in other, more challenging tasks. We are especially interested in checking the performance of our approach in those datasets for which traditional data augmentation is not advisable.

Acknowledgements. This work was supported by the Spanish Ministerio de Ciencia, Innovación y Universidades through HISPAMUS project (Ref. TIN2017-86576-R, partially funded by UE FEDER funds).

References

1. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
2. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
3. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. Computing Research Repository abs/1312.6114 (2013)
4. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: 26th Annual Conference on Neural Information Processing Systems, pp. 1106–1114 (2012)
5. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)
6. Lv, J.J., Cheng, C., Tian, G.D., Zhou, X.D., Zhou, X.: Landmark perturbation-based data augmentation for unconstrained face recognition. Signal Process. Image Commun. **47**, 465–475 (2016)

7. Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. In: 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21–26 June 2014, pp. 1278–1286 (2014)
8. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, pp. 318–362. MIT Press, Cambridge (1986)