Improving Auto-Encoders' self-supervised image classification using pseudo-labelling via data augmentation and the perceptual loss

Aymene Mohammed Bouayed, Karim Atif, Rachid Deriche, Abdelhakim Saim

▶ **To cite this version:**

Aymene Mohammed Bouayed, Karim Atif, Rachid Deriche, Abdelhakim Saim. Improving Auto-Encoders' self-supervised image classification using pseudo-labelling via data augmentation and the perceptual loss. 2020. hal-03046243

HAL Id: hal-03046243
https://hal.science/hal-03046243

Preprint submitted on 8 Dec 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Preprint Under Consideration at Computer Vision and Image Understanding Journal

Improving Auto-Encoders' self-supervised image classification using pseudo-labelling via data augmentation and the perceptual loss

Aymene Mohammed Bouayed[a,**], Karim Atif[a], Rachid Deriche[b], Abdelhakim Saim[c,d]

[a]*Computer Science Department, Faculty of Electronics and Computer Science, University of Sciences and Technology Houari Boumediene, BP. 32, Bab-Ezzouar, Algiers, Algeria.*
[b]*Athena Project-Team, INRIA Sophia-Antipolis-Méditérranée, 06902 Sophia Antipolis, France.*
[c]*IREENA - Institut de Recherche en Energie Electrique de Nantes Atlantiques Saint-Nazaire, France.*
[d]*Control and Instrumentation Department, Faculty of Electronics and Computer Science, University of Sciences and Technology Houari Boumediene, BP. 32, Bab-Ezzouar, Algiers, Algeria.*

ABSTRACT

In this paper, we introduce a novel method to pseudo-label unlabelled images and train an Auto-Encoder to classify them in a self-supervised manner that allows for a high accuracy and consistency across several datasets. The proposed method consists of first applying a randomly sampled set of data augmentation transformations to each training image. As a result, each initial image can be considered as a pseudo-label to its corresponding augmented ones. Then, an Auto-Encoder is used to learn the mapping between each set of the augmented images and its corresponding pseudo-label. Furthermore, the perceptual loss is employed to take into consideration the existing dependencies between the pixels in the same neighbourhood of an image. This combination encourages the encoder to output richer encodings that are highly informative of the input's class. Consequently, the Auto-Encoder's performance on unsupervised image classification is improved both in termes of stability and accuracy becoming more uniform and more consistent across all tested datasets. Previous state-of-the-art accuracy on the MNIST, CIFAR-10 and SVHN datasets is improved by 0.3%, 3.11% and 9.21% respectively.

1. Introduction

Classification is one of the most important tasks in deep learning. It consists of identifying a trait in the input and assigning a label to it. The input could be an image, a video, a simple vector of values or else.

Classification has many useful and valuable applications such as spam detection (Chetty et al., 2019), disease identification (Zilong et al., 2018), particle discovery (Bouayed and Atif, 2019) etc. Current deep learning technics are able to achieve outstanding performance on this task using supervised learning. However, the efficacy of these methods depends on the presence of labeled data which is very scarce. For this aim, the development of unsupervised and semi-supervised methods has seen an increasing interest.

In order to benefit from the sheer amount of available unlabelled data, a lot of work has been done to improve the performance of deep learning models in the unsupervised learning context. Among these methods, Radford et al. (2016) propose to concatenate the output of some of the convolutional layers of the GAN's discriminator (Generative Adverserial Network) and pass them through a linear classifier to infer the class of the input images. In the work of Kosiorek et al. (2019), a Stacked Capsule Auto-Encoder has been used to break down an image into multiple objects. Then, according to the present objects in a scene, the class of the image can be deduced.

Auto-Encoders are neural networks that are trained to attempt to reconstruct the input that is presented to them and construct an internal representation of it (Goodfellow et al., 2016). They have been used to different ends among them modelling (Wu et al., 2017), de-noising (Vincent et al., 2008) and more.

The work of Bouayed and Kennouche (2020) explored the use of Auto-Encoders to do self-supervised classification and optimise the architecture of the network using the Genetic Al-

**Corresponding author: Tel.: +213-698-727-691;
e-mail: bouayedaymene@gmail.com (Aymene Mohammed Bouayed)

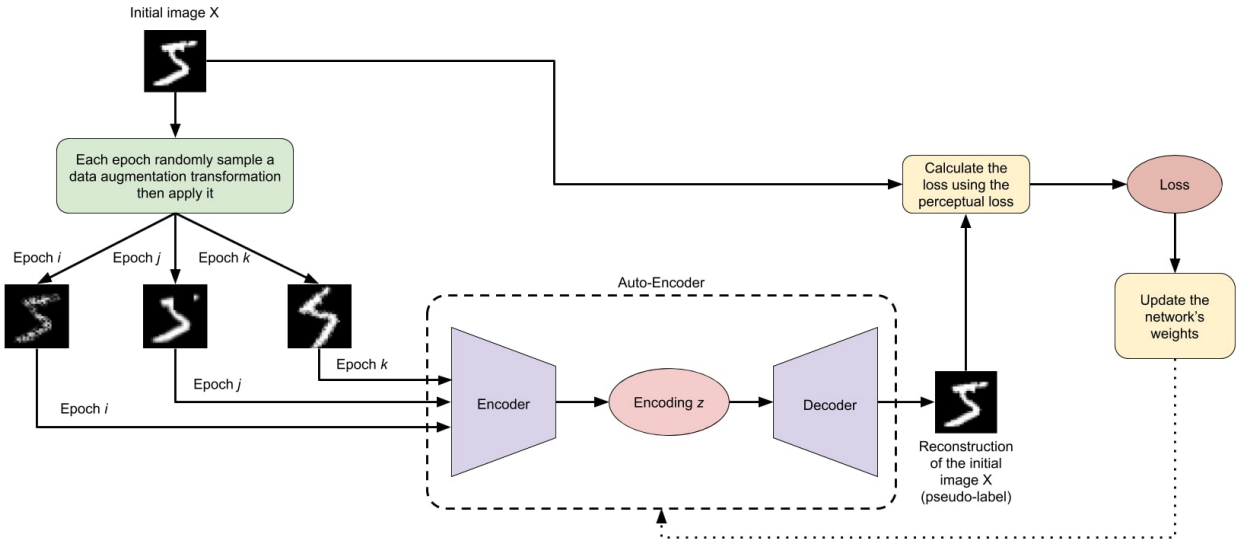

Fig. 1. Auto-Encoder with Pseudo-Labelling method proposed in this work. (2-column image).

gorithm. This work obtained satisfactory results on the MNIST dataset where 96.32% accuracy has been achieved with an optimised Auto-Encoder that has fully connected layers.

Conventionally, Auto-Encoders use a pixel-wise loss function to calculate the difference between the input and the output. Thus, the pixel-wise loss does not take into consideration the dependencies between the neighbouring pixels of an image. To solve this problem, the perceptual loss was introduced by Larsen et al. (2016) and its importance and impact have been evaluated for image reconstruction and generation where it demonstrated great potential. The perceptual loss uses a pre-trained model to capture the dependencies between the pixels of an image then applies the loss function to the output of this model.

The work of Pihlgren et al. (2020) explored the effect of the perceptual loss on the embeddings produced by the encoder in the task of object positioning and image classification. It has been found that the perceptual loss enriches the encodings produced w.r.t the semantic class of the input. Big improvements in terms of classification accuracy have been noticed over not using the perceptual loss.

Similarly, the use of data augmentation has been shown as a powerful technique to improve the generalisation capabilities on neural networks in a supervised learning context (Shorten and Khoshgoftaar, 2018).

Data augmentation is a technique where alterations are introduced into the training data in order to improve its diversity and allow the network to see the data from different perspectives during the training phase. In the recent work of Chen et al. (2020) data augmentation was one of the most important ideas that allowed obtaining state-of-the-art performance on multiple datasets, surpassing the accuracy of neural networks having done the learning phase in a supervised manner. In the work of Chen et al. (2020), data augmentation is applied to an image twice and then encoded using a large pre-trained model. The weights of the network are then refined so that the encodings of the image are the same regardless of the applied data augmentations.

Despite the effectiveness of data augmentation in the supervised and the semi-supervised contexts, this technique's effect on the embeddings produced by an Auto-Encoder in the unsupervised classification task has not been explored yet notably when used in conjunction with the perceptual loss.

In this work we found that incorporating data augmentation in a simple way where the Auto-Encoder reconstructs its input, which is an image with data augmentation, does not improve the embeddings produced. As a result, we introduce a novel and effective unsupervised method, the *Pseudo-Labelling* method. This method consists of creating a correspondance between data augmented images and the original image (see Figure 1). Combining the proposed method with an Auto-Encoder trained with the perceptual loss results in improvements in both the stability and the accuracy of the unsupervised classification. Consequently, the proposed method incorporates data augmentation in a way that forces the encoder to better understand the input, look past the alterations applied and create more concise encodings.

Subsequently, we summarise our contributions in this work as follow:

- We emphasise the improvements that the perceptual loss brings when compared to a simple Auto-encoder w.r.t the unsupervised images classification accuracy.
- We propose a novel method of incorporating data augmentation called *Pseudo-Labelling* that leads the encoder to improve the quality of the outputted embeddings.
- We demonstrate the superior performance of the method we propose in this work compared to 8 different methods including other Auto-Encoder and none Auto-Encoder based methods (Radford et al., 2016) (Kosiorek et al., 2019) (Pihlgren et al., 2020) (Haeusser et al., 2018) (Bengio et al., 2007) (Hu et al., 2017) (Ji et al., 2019).

The outline of this paper is as follows. In Section 2, we present the basic concepts of the Auto-Encoders then we detail the main idea of Auto-Encoders trained using the perceptual loss. In Section 3, we deal with the data augmentation method applied to images. Section 4 introduces two different ways of incorporating data augmentation in an Auto-Encoder that is trained using the perceptual loss. The first one uses data augmentation in a simple way, whereas the second one (the Pseudo-Labelling method) harnesses data augmentation in a beneficial way that urges the network to learn better encodings. In Section 5, we describe the experimental setup we used, our data augmentation transformations selection and the results of our experiments. Section 6 compares the two methods of incorporating data augmentation into an Auto-Encoder to variety of Auto-Encoder and none Auto-Encoder based methods in the task of unsupervised image classification. Our conclusions and perspectives for future work are presented in Section 7.

2. Auto-Encoder

2.1. Basic Concepts

An Auto-Encoder (B-AE: *Basic Auto-Encoder*) is a special kind of neural network architecture that is composed of two neural networks an encoder and a decoder. The encoder and the decoder are stacked one on top of the other and trained at the same time. The goal of the encoder is to produce a concise encoding z that keeps all the important information about the input X. The decoder's task is to use the encoding z and reconstruct the input the best way possible (Goodfellow et al., 2016).

$$z = encoder(X) \tag{1}$$

$$X' = decoder(z) \tag{2}$$

The whole network is trained using the back-propagation algorithm. While the loss is computed by comparing the input image X to its reconstruction X' using a loss function ℓ such as the MSE (Mean Squared Error) loss (See Figure 2 and Algorithm 1).

2.2. Auto-Encoder with Perceptual Loss

Convolutional layers first introduced in the work of LeCun et al. (1998), rely on the fact that pixels in an image are not independent entities but they depend on the neighbouring pixels. This idea allows to extract more information from the images by capturing these dependencies and harnessing them to perform better on the task at hand.

This concept is also used in the perceptual loss of Auto-Encoders (Larsen et al., 2016) (Pihlgren et al., 2020). Where in this context too the value of a pixel depends on the value of the neighbouring pixels. So, the loss function ℓ should not be applied pixel-wise between the input X and its reconstruction X' as done in the B-AE:

$$\mathcal{L} = \sum \ell(X', X) \tag{3}$$

Instead, X and X' are passed through a portion of a pre-trained model p that captures the relationships between the pixels in the same neighbourhood. Then, the loss function ℓ is applied to the output of the pre-trained network p. As a result, for the Auto-Encoder with perceptual loss (P-AE), equation (3) and the corresponding formula in line 9 of the Algorithm 1 should be replaced with the following formula:

$$\mathcal{L} = \sum \ell(p(X'), p(X)) \tag{4}$$

This allows to compare regions of the input image X with their correspondent regions of the reconstructed image X'. Figure 3 illustrates the perceptual loss module.

The impact of the P-AE on the quality of the embeddings produced by the encoder has been discussed in the work of Pihlgren et al. (2020). It has been found that the perceptual loss greatly enriches the produced latent vector z allowing a higher accuracy in self-supervised classification.

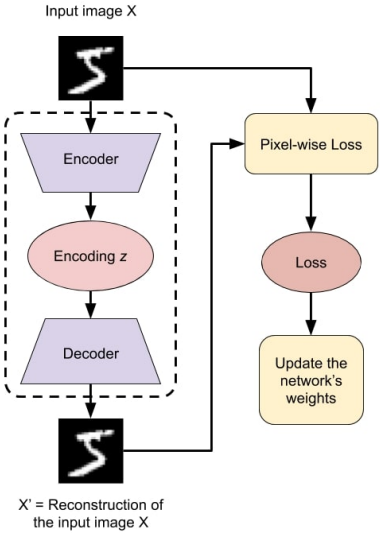

Fig. 2. A chart of a simple Auto-Encoder where the input image X is passed through the encoder to obtain an encoding z. Then the encoding z is passed through the decoder to generate X' which is a reconstruction of X. Finally, X and X' are compared using a pixel-wise loss function. (1-column image).

Algorithm 1: B-AE main learning algorithm.

1 **input:** batch size N, number of epochs *epochs*
2 epoch $\leftarrow 1$
3 **while** *epoch* $\leq$ *epochs* **do**
4 **for *each*** *sampled minibatch* $\{x_i\}_{k=1}^{N}$ **do**
5 **for *all*** $k \in \{1, \ldots, N\}$ **do**
6 $z_k \leftarrow encoder(x_k)$
7 $x'_k \leftarrow decoder(z_k)$
8 **end**
9 $\mathcal{L} = \sum_{k=1}^{N} \ell(\boldsymbol{x'_k}, \boldsymbol{x_k})$
10 Update the *encoder*'s and the *decoder*'s weights to minimise $\mathcal{L}$
11 **end**
12 epoch $\leftarrow$ epoch + 1
13 **end**
14 **return** *encoder* and *decoder*

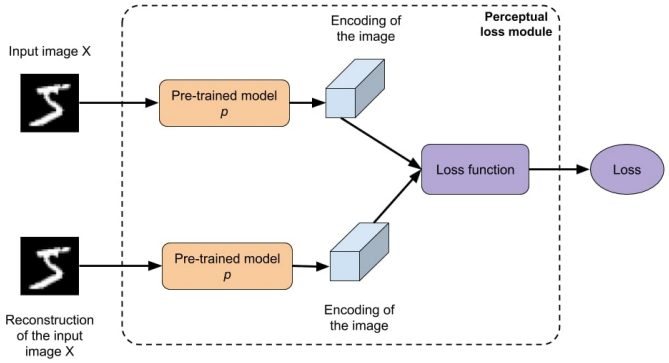

Fig. 3. Content of the perceptual loss module. In an Auto-Encoder that uses the perceptual loss, the pixel-wise loss module of Figure 2 is replaced by this module where first the original image X and its reconstruction X' are passed through a pre-trained model p. This allow to obtain encodings which contain the important information and the extracted dependencies between the pixels. Then, a loss function such as the MSE Loss is used to compare the encodings pixel-wise. (1-column image).

3. Data Augmentation

Data augmentation is a technique that is heavily used in the field of computer vision. It allows the neural network to build an internal representation of the data that is more robust to small changes in the input and better generalise to new data thus obtaining a higher accuracy (Shorten and Khoshgoftaar, 2018).

Data augmentation works through the introduction of new images into the training set by transforming the existing ones. The transformations that could be applied include but not limited to horizontally or vertically flipping an image, hiding parts of the image (cutout), adding gaussian noise to the image and so on (see Figure 4).

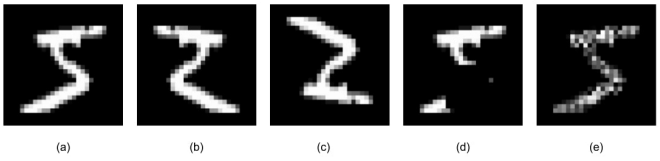

Fig. 4. Examples of data augmentation transformations on an image from the MNIST dataset. (a) represents the original image, (b) and (c) represent horizontally and vertically flipping the original image, (d) represents the cutout transformation and (e) represents the original image with added gaussian noise. (1-column image).

4. Proposed Approach

4.1. Auto-Encoder with Simple Data Augmentation

In order to further improve the quality of the encodings produced by the P-AE, data augmentation could be incorporated into the Auto-Encoder.

An Auto-Encoder with Simple Data Augmentation (SDA-AE) can be used where a data augmentation transformation t is applied to the input image X to obtain a modified image denoted F_S.

$$F_S = t(X) \tag{5}$$

Then, the modified image F_S is used as input and target of the P-AE (see algorithm 2).

Algorithm 2: SDA-AE main learning algorithm.

input: batch size N, number of epochs *epochs*, a portion of a pre-trained model p
epoch $\leftarrow 1$
while *epoch* $\leq$ *epochs* **do**
  for ***each*** *sampled minibatch* $\{x_i\}_{k=1}^{N}$ **do**
    for ***all*** $k \in \{1, \ldots, N\}$ **do**
      $t \leftarrow$ a data augmentation transformation
      $f_{S,k} \leftarrow t(x_k)$
      $z_k \leftarrow encoder(f_{S,k})$
      $x'_k \leftarrow decoder(z_k)$
    end
    # Using the perceptual loss and applying it between the image with data augmentation and its reconstruction.
    $\mathcal{L} = \sum_{k=1}^{N} \ell(p(\boldsymbol{x'_k}), p(\boldsymbol{f_{S,k}}))$
    Update the *encoder*'s and the *decoder*'s weights to minimise $\mathcal{L}$.
  end
  epoch $\leftarrow$ epoch + 1
end
return *encoder* and *decoder*

Even though the SDA-AE represents a viable solution to integrate data augmentation into the P-AE, it might be unstable during the training phase. This is due to the fact that there isn't a fixed target for each input. As a consequence, the quality of the important information stored in the encodings might fluctuate and not result in a better accuracy compared to the P-AE. For this aim we propose a novel method to incorporate data augmentation in the PL-AE presented in the next section.

4.2. Auto-Encoder with Pseudo Labelling

The Auto-Encoder with Pseudo Labelling (PL-AE) proposed in this work is an unsupervised method that imitates the mapping **input** $\rightarrow$ **label** used in the supervised learning context.

To do so, various data augmentation transformations are applied to generate variations of the training images. This results in multiple data augmented images that correspond to each single initial training image. We can then consider the initial images as *pseudo labels* to the sets of their corresponding data augmented images. After that we use an Auto-Encoder to learn the mapping **data augmented images** $\rightarrow$ **original image (pseudo-label)** (see Figure 5). Furthermore, the PL-AE makes use of the perceptual loss as it allows to construct richer encodings that provide accurate information w.r.t the image's class.

Since in the PL-AE the input to the Auto-Encoder is richer, this encourages the encoder part to construct a deeper understanding of the images regardless of the applied transformations, and then encode them in a concise way. Also, PL-AE provides a stable target that is not affected by any data augmentation. This allows for a much more stable learning as opposed to the SDA-AE.

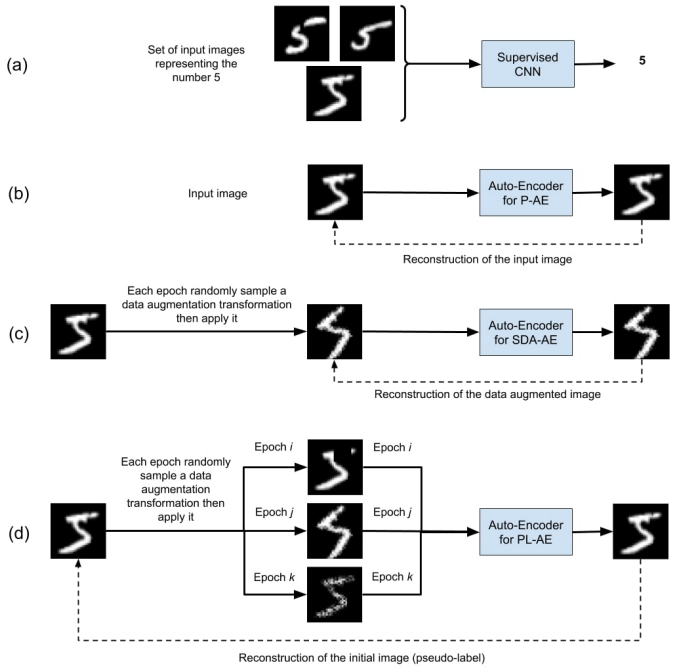

Fig. 5. PL-AE vs different Auto-Encoder approches and a supervised CNN: The supervised CNN (a) maps the different images where 5 is drawn to the same label 5. The P-AE (b) and the SDA-AE (c) reconstruct their input images. Whereas the PL-AE (d) maps the different data augmented input images to the same initial image (pseudo-label). (1-column image).

To apply the proposed method (PL-AE), firstly a data augmentation transformation t is applied to the input image X of the encoder. It should be noted that from epoch to epoch, the data augmentation transformation t changes. This allows the creation of different variations of the original image and learn a better mapping between them and the original one.

$$F_{PL} = t(X) \tag{6}$$

$$z = encoder(F_{PL}) \tag{7}$$

Then, the decoder takes the encoding z and reconstructs the input X (the one without any data augmentation) as opposed to the SDA-AE which reconstructs F_S.

$$X' = decoder(z) \tag{8}$$

Where X' is the reconstruction of the original image X.

Finally, the perceptual loss is used instead of the pixel-wise loss to compare X with X' as it allows for a richer encoding z as has been proven by Pihlgren et al. (2020).

As a result the training algorithm for the PL-AE is similar to the one of the SDA-AE with all instances of $f_{S,k}$ in Algorithm 2 replaced with $f_{PL,k}$, except for the loss function in line 12 which is replaced with the following formula:

$$\mathcal{L} = \sum_{k=1}^{N} \ell(p(x'_k), p(\boldsymbol{x_k})) \tag{9}$$

The complete PL-AE system is illustrated in Figure 1.

5. Experiments and Results

In this section we present our experiments to validate the PL-AE approach proposed in this work.

5.1. Datasets

This work makes use of three different classification datasets which are MNIST (LeCun et al., 1998), CIFAR-10 (Krizhevsky, 2009) and SVHN (Netzer et al., 2011).

5.1.1. MNIST

The MNIST dataset is a collection of grey scale images of hand written digits. Each image has a size of 28×28 and only one channel. The dataset is divided into 60,000 images for the training set and 10,000 images for the test set. The goal in this dataset is to classify the images according to the digit that is drawn (see Figure 6) (LeCun et al., 1998).

In our work, before using the images of this dataset, we rescale them up to a size of 32×32, then we duplicate the image three times along the channels axis to have the same image but with three channels.

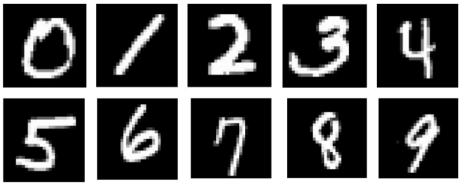

Fig. 6. Sample images from the MNIST dataset. (1-column image).

5.1.2. CIFAR-10

The CIFAR-10 dataset is a collection of 32×32 coloured images of objects and animals. It englobes 50,000 images for the training set and 10,000 images for the test set. The goal in this dataset is to identify the object or the animal in the image (see Figure 7) (Krizhevsky, 2009).

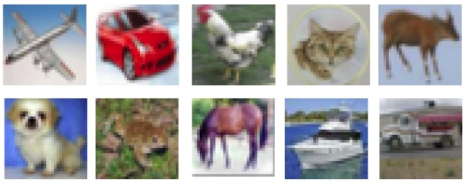

Fig. 7. Sample images from the CIFAR-10 dataset. (1-column image).

5.1.3. SVHN

The SVHN (Street View House Numbers) dataset is composed of 73,257 training images and 26,032 test images. Each image has a size of 32x32 and represents a digit from a house number, and the goal is to identify this digit (see Figure 8) (Netzer et al., 2011).

Finally, for each image of all the three datasets we duplicate it into a 2×2 grid to get an image of 64×64 pixels as it has been done by Pihlgren et al. (2020).

Fig. 8. Sample images from the SVHN dataset. (1-column image).

5.2. Training Environment

To implement the proposed method several libraries have been used namely:

- **The Pytorch library** (Paszke et al., 2019) which provides implementations of different neural network layers, optimisers and pre-trained models. It has been used for the implementation of the perceptual loss and the different Auto-Encoders in this work.
- **The Torchvision library** (Marcel and Rodriguez, 2010) which provides an easy access to several benchmark datasets and a multitude of data augmentation transformations. It has been used to get the datasets that have been tested in this work and apply data augmentation to them.
- **The Sci-kit learn library** (Buitinck et al., 2013) which provides an implementation of linear classifiers (Fan et al., 2008) and the t-SNE algorithm (van der Maaten and Hinton, 2008). Where the latter has been used to cast the encodings from a high dimensional space to 2D to be plotted, and the linear classifier has been used to calculate the accuracy of the encodings generated by the Auto-Encoders for the different tested datasets.
- **The Matplotlib library** (Hunter, 2007) which is a general purpose plotting library on Python. It has been used to draw the scatter plots of the obtained encodings.

Nvidia Tesla GPUs provided on the Google Colaboratory website which include T4, P4, P100 and K80 GPUs with 8Go of VRAM were used to train the different models. It takes for each epoch around 2, 3 or 4 minutes to train and evaluate the accuracy of an Auto-Encoder for the MNIST, CIFAR-10 and SVHN datasets respectively.

5.3. Training Parameters

All the Auto-Encoder training methods (B-AE, P-AE, SDA-AE and PL-AE) have been implemented using the same backbone architecture, similar to the one illustrated in Figure 9 and used by Pihlgren et al. (2020).

Each Auto-Encoder has been trained for 90 epochs using the MSE loss function and Adam optimiser with a learning rate of lr=0.001, β_1=0.9 and β_2=0.999 (Kingma and Ba, 2015) and a batch size of 100.

For the models making use of the perceptual loss, the pre-trained model selected, denoted as p in Algorithm 2, is the portion of the Alexnet model (Krizhevsky et al., 2012) showcased in Figure 10 which consists of the first three layers of this network. This portion is the same as the one used by Dosovitskiy and Brox (2016) and Pihlgren et al. (2020). The whole Alexnet network is publicly available on the Pytorch library and any portion of it could be taken.

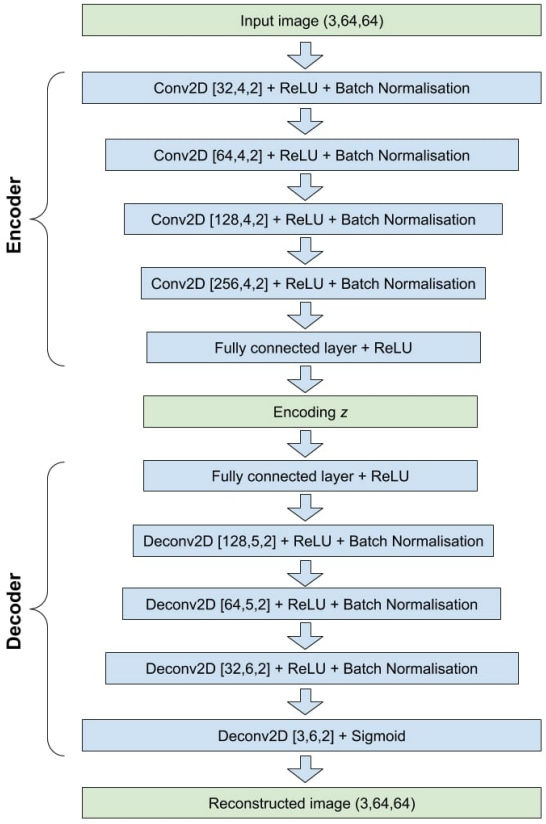

Fig. 9. Backbone architecture of the Auto-Encoder used in this work. Conv2D stands for the 2D convolution operation with [number of kernels, size of kernels, stride] parameters, ReLU for Rectified Linear Unit activation function, Deconv2D for the 2D deconvolution operation with [number of kernels, size of kernels, stride] parameters, and Sigmoid for the sigmoid activation function. (1-column image).

Also, since the pre-trained Alexnet model takes as input images of size 224×224 and the images of the datasets used in this work are of size 64×64, we pad the images with zeros all around until we get the desired size.

5.4. Accuracy Evaluation

The performance of the Auto-Encoders w.r.t the classification accuracy is computed by training a linear classifier on the training images encodings and their corresponding labels. Then, the linear classifier is evaluated on the test images encodings and their corresponding labels. Finally, the obtained accuracy is reported as done in (Kosiorek et al., 2019). No data augmentation is used during this phase (see Algorithm 3).

5.5. Data Augmentation Transformations Selection

In order to find the data augmentation transformations that yield the best unsupervised classification accuracy, we studied different ones for each dataset using the proposed PL-AE method. Also, we made sure to consider the transformations

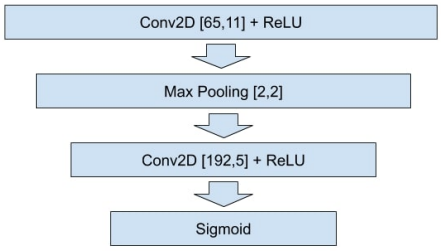

Fig. 10. The used portion of the Alexnet network to calculate the perceptual loss. (1-column image).

Algorithm 3: Accuracy evaluation algorithm.

input: training images and their labels (*train_img*, *train_labels*), test images and their labels (*test_img*, *test_labels*), The encoder part from the Auto-Encoder to be tested *encoder*, linear classifier *lin*.
train_encoding ← *encoder*(*train_img*)
test_encoding ← *encoder*(*test_img*)
lin.fit(train_encoding, *train_labels*)
accuracy ← *lin*.score(test_encoding, *test_labels*)
return accuracy

that are semantically correct to apply then tested them. This is to ensure that the distribution of the obtained images is not far from the one of the train and test images. For example we did not use vertical and horizontal flips for the datasets containing numbers (MNIST, SVHN) as their combination for the number 6 will give the number 9. Table 1 summarises the appropriate transformations that have been evaluated for each dataset. The parameters of each transformation can be found in Appendix A.

Transformation	MNIST	CIFAR-10	SVHN
Random rotation	✓	✓	✓
Affine transformation	✓	✓	✓
Crop	✓	✓	✓
Cutout	✓	✓	✓
Gaussian noise	✓	✓	✓
Colour jitter	-	✓	✓
Gray scale	-	✓	✓
Horizontal flip	-	✓	-
Vertical flip	-	✓	-

Table 1. Data augmentation transformations that have been tested for each dataset.

In addition, applying a combination of two different data augmentation transformations to the input images has been explored. This choice is motivated by the findings of Chen et al. (2020). In (Chen et al., 2020) it has been found that even though applying two different data augmentation transformations makes the learning process harder, it yields to richer encodings. The results obtained [1] are reported in the heat maps of Figure 11, 12 and 13 which correspond to the MNIST, CIFAR-10 and SVHN datasets respectively. The diagonal represents when only one data augmentation is applied whereas the rest represents the use of two different data augmentations.

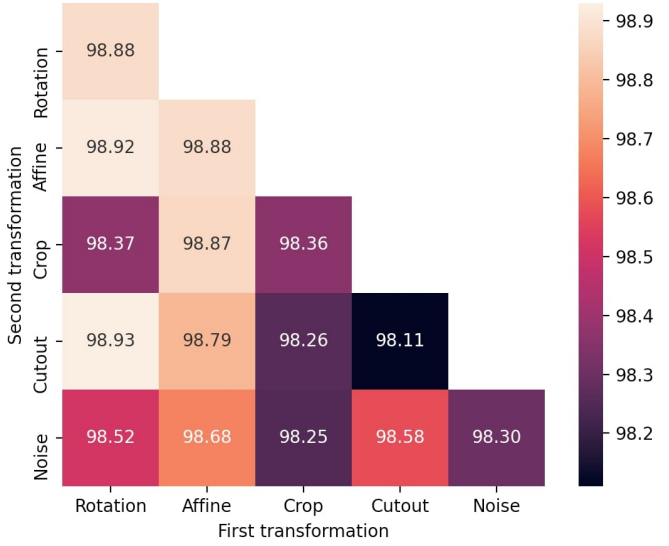

Fig. 11. Accuracies obtained for different data augmentations on the MNIST dataset. (1-column image).

From figure 11 it can be seen that the accuracies obtained for the different data augmentations are close. This means that none of them has the potential to hinder the performance. As a result, sampling randomly from the list of the explored transformations for the MNIST dataset is possible. That is what we will be doing in our following experiments for the MNIST dataset.

Figure 12 and 13 show that there are several transformations that did not obtain high accuracies such as the gaussian noise for the CIFAR-10 dataset and the crop transformation for the SVHN dataset. So, if we sample randomly from a list of all possible transformations as done on the MNIST dataset there will be a performance degradation. To counter this problem and maximise the unsupervised classification accuracy, for our following experiments we sample from a list containing only the top-10 transformations for each dataset whenever data augmentation is used.

5.6. Results

After training different Auto-Encoders with different training methods (B-AE, P-AE, SDA-AE, PL-AE) we report on the column B-AE, P-AE, SDA-AE, PL-AE of the Tables 2, 3 and 4 the best accuracies obtained.

- B-AE column represents a basic Auto-Encoder which uses a pixel-wise loss and has as input and target the images X without any data augmentation.

[1] The training parameters for the exploration of the different data augmentation transformations are the same as previously stated except for the number of epochs which was reduced to 30 epochs.

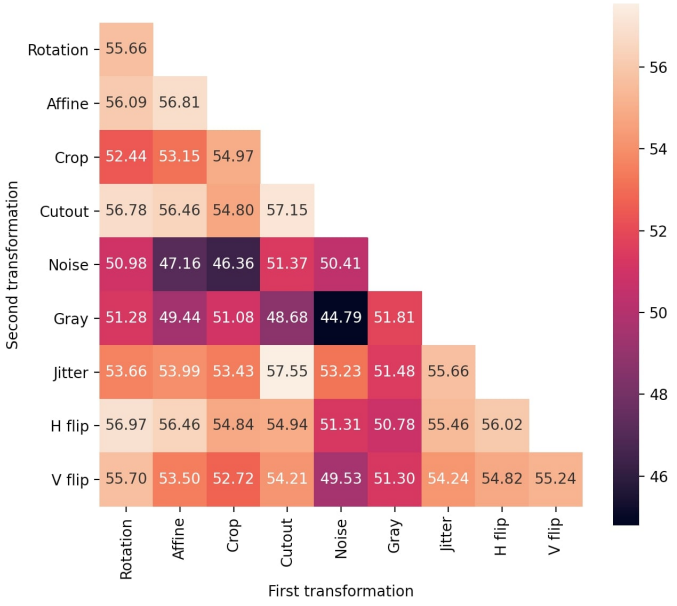

Fig. 12. Accuracies obtained for different data augmentations on the CIFAR-10 dataset. (1-column image).

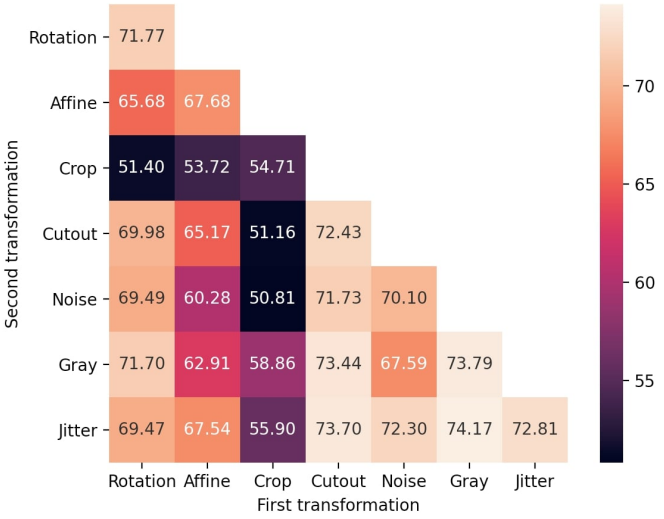

Fig. 13. Accuracies obtained for different data augmentations on the SVHN dataset. (1-column image).

- P-AE column represents an Auto-Encoder which uses the perceptual loss and has as input and target the images X without any data augmentation.
- SDA-AE column represents an Auto-Encoder which uses the perceptual loss and has as input and target the images with data augmentation F_S.
- PL-AE column represents an Auto-Encoder which uses the perceptual loss and the Pseudo-Labelling method.
- CNN column represents the performance of a supervised CNN (Convolutional Neural Network) [2]. This column is used as reference to compare the unsupervised methods (B-AE, P-AE, SDA-AE, PL-AE) with the supervised CNN. It should be noted that for the CNN we apply the same data augmentation policy as applied for the SDA-AE and PL-AE.

5.6.1. MNIST Results

After training multiple Auto-Encoders with different methods and embedding sizes on the MNIST dataset, We report the best obtained accuracies in the Table 2 below.

When looking at Table 2 from the effect of the embedding size perspective, it can be seen that most of the methods' accuracy reaches its peak at an embedding size of 300. Also, the PL-AE approach stores the most information in this embedding and obtains an accuracy that is almost the same as the equivalent CNN network.

[2]The CNN was trained in a supervised manner using the cross entropy loss function. Its architecture is composed of the architecture of the used encoder (showcased in Figure 9) combined with an MLP (Multi-Layer Perceptron) classification head.

Embedding size	MNIST				
	B-AE	P-AE	SDA-AE	PL-AE	CNN
250	95.91%	97.78%	98.12%	**98.7%**	**99.38%**
300	95.96%	98.37%	98.04%	**99.3%**	**99.44%**
350	96.67%	98.49%	98.37%	**99.27%**	**99.43%**

Table 2. Best performances obtained on the test set of the MNIST dataset for different embedding sizes. The supervised CNN column is provided as reference to the other unsupervised Auto-Encoder based methods. The best unsupervised and the CNN accuracies obtained are highlighted in bold.

From an overall accuracy perspective, It can be appreciated that the P-AE greatly improves on the accuracy of the B-AE by around 2% whereas the contribution of the SDA-AE method compared to the P-AE is negligible. The proposed PL-AE method allows an enhancement in the performance of the unsupervised classification with 0.81% over the P-AE and falls short by only 0.14% when compared to the CNN.

From Figure 14 it can be seen that the B-AE, P-AE and the SDA-AE reach a peak in performance within the first 15 epochs then their accuracies drop. This implies that there is a loss of important information that is helpful to the classification within the generated encodings. Also, these approaches suffer from a lot of fluctuations in unsupervised classification accuracy especially after 40 epochs. As for the proposed PL-AE method, it achieves a very high and sustained performance with a maximum accuracy of 99.3%. In addition, it can be appreciated that the PL-AE's curve is very close to the one of the CNN.

Figure 15 represents the embeddings of the test images of the MNIST dataset produced by the PL-AE and plotted in 2D using the t-SNE algorithm (van der Maaten and Hinton, 2008). It can be seen that well-formed clusters for each class are constructed which confirms the high accuracy obtained on this dataset.

5.6.2. CIFAR-10 Results

Table 3 showcases the best accuracies we obtained on the CIFAR-10 dataset for the different Auto-Encoder training meth-

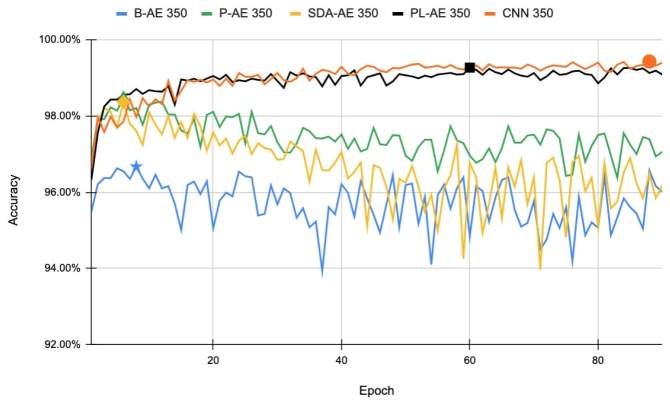

Fig. 14. Classification accuracy vs number of epoch on the MNIST dataset. The best accuracies obtained by the B-AE, P-AE, SDA-AE, PL-AE and the supervised CNN are marked with a blue ★, a green ▲, a yellow ♦, a black ■ and an orange • respectively. (This Figure uses colours, 1-column image).

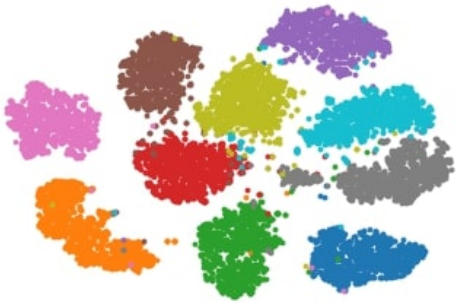

Fig. 15. Encodings of MNIST test images produced by the PL-AE plotted in 2D using the t-SNE algorithm. Each colour represents one class from 0 to 9. (This Figure uses colours, 1-column image).

ods.

Embedding size	CIFAR-10				
	B-AE	P-AE	SDA-AE	PL-AE	CNN
250	36.27%	55.22%	56.21%	**60.66%**	**80.77%**
300	37.67%	56.33%	56.86%	**60.60%**	**81.14%**
350	39.15%	56.46%	57.42%	**60.71%**	**81.73%**

Table 3. Best performances obtained on the test set of the CIFAR-10 dataset for different embedding sizes. The supervised CNN column is provided as reference to the other unsupervised Auto-Encoder based methods. The best unsupervised and the CNN accuracies obtained are highlighted in bold.

It can be observed from Table 3 that for the CIFAR-10 dataset, the PL-AE results in better accuracy compared to the P-AE and SDA-AE by 4.25% and 3.29% respectively, and achieves the best unsupervised classification accuracy with 60.71%. Whereas when comparing the SDA-AE with the P-AE it can be seen that a maximum of almost 1% improvement have been made. On the other hand, the P-AE drastically improves the classification accuracy compared to the basic Auto-Encoder (B-AE) with up to 17.66%. Finally, comparing all unsupervised methods to the CNN it is clear that the CNN still has the edge with a margin of 21.02%.

From an embedding size perspective, it can be seen from Table 3 that all the methods stored the maximum information in the embeddings sizes used except for the B-AE. Thus, among all the Auto-Encoders tested the PL-AE stores the most useful information to the classification task.

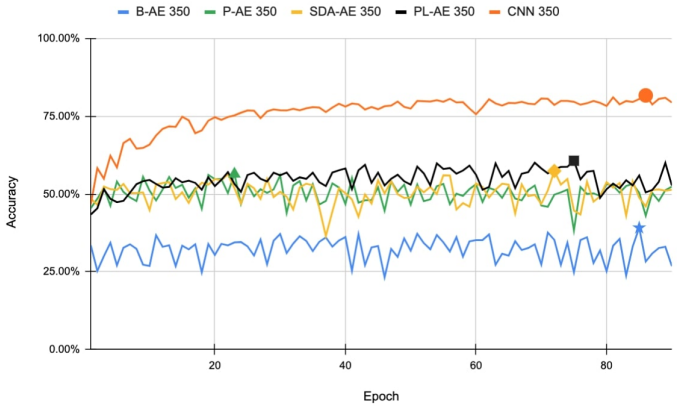

Fig. 16. Classification accuracy vs number of epoch on the CIFAR-10 dataset. The best accuracies obtained by the B-AE, P-AE, SDA-AE, PL-AE and the supervised CNN are marked with a blue ★, a green ▲, a yellow ♦, a black ■ and an orange • respectively. (This Figure uses colours, 1-column image).

Figure 16 shows that the accuracy of the B-AE, P-AE and the SDA-AE does not improve over the epochs and has a lot of fluctuations. Whereas the accuracy of the PL-AE improves over the epochs with far less fluctuations and surpasses all the other Auto-Encoder based methods.

5.6.3. SVHN Results

Table 4 presents the best accuracies we obtained on the SVHN dataset for the different Auto-Encoder training methods.

Embedding size	SVHN				
	B-AE	P-AE	SDA-AE	PL-AE	CNN
250	32.99%	73.01%	72.07%	**75.59%**	**92.53%**
300	35.12%	74.67%	72.92%	**76.21%**	**92.57%**
350	36.48%	74.76%	74.17%	**76.48%**	**92.57%**

Table 4. Best performances obtained on the test set of the SVHN dataset for different embedding sizes. The supervised CNN column is provided as reference to the other unsupervised Auto-Encoder based methods. The best unsupervised and the CNN accuracies obtained are highlighted in bold.

It can be observed from Table 4 that for the SVHN dataset in terms of classification accuracy the P-AE obtains an accuracy of 74.76% immensely improving upon the accuracy of the B-AE by a margin of 38.28% which is more than double. The P-AE also achieves an accuracy that is higher than the one of the SDA-AE by just 0.59%. Thus, it falls short compared to the proposed PL-AE which improves on its accuracy by 1.72%. Still the supervised method CNN achieves the best accuracy with at least 16.36% better accuracy.

Table 4 also shows that all the Auto-Encoder based methods benefited from increasing the embedding size. The P-AE, PL-AE and CNN reach their maximum accuracy at an embedding size of 300 and increasing it to 350 does not improve the accuracy much if any. Whereas the accuracy of the B-AE and the

SDA-AE seem to still improve further and benefit from a larger embedding size.

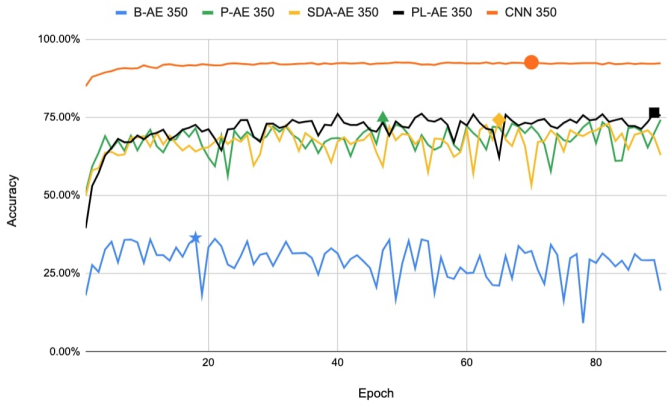

Fig. 17. Classification accuracy vs number of epoch on the SVHN dataset. The best accuracies obtained by the B-AE, P-AE, SDA-AE, PL-AE and the supervised CNN are marked with a blue ★, a green ▲, a yellow ♦, a black ■ and an orange • respectively. (This Figure uses colours, 1-column image).

From Figure 17, it can be seen that there is no improvement in the accuracy of the P-AE and SDA-AE over the training epochs and a lot of fluctuations are present. Whereas, the PL-AE achieves a competitive performance and surpasses the other Auto-Encoder based methods after few epochs, learning better encodings with a much smoother accuracy curve.

6. Comparison with State of the Art

Method	MNIST	CIFAR-10	SVHN	Reference
K-means	53.49%	20.8%	12.5%	Haeusser et al. (2018)
AE	81.2%	31.4%	-	Bengio et al. (2007)
GAN	82.8%	31.5%	-	Radford et al. (2016)
IMSAT	98.4%	45.6%	57.3%	Hu et al. (2017)
IIC	98.4%	**57.6%**	-	Ji et al. (2019)
ADC	98.7%	29.3%	38.6%	Haeusser et al. (2018)
SCAE	**99.0%**	33.48%	**67.27%**	Kosiorek et al. (2019)
B-AE	96.67%	39.15%	36.48%	-
P-AE	98.49%	57.46%	74.76%	Pihlgren et al. (2020)
SDA-AE	98.37%	57.42%	74.17%	Ours
PL-AE	**99.30%**	**60.71%**	**76.48%**	Ours
CNN	99.44%	81.73%	92.57%	-

Table 5. Unsupervised classification accuracy results in %. The supervised CNN line is provided as reference to the other unsupervised methods. Previous state-of-the art accuracies and the best accuracies we obtained are highlighted in bold.

Table 5 compares the performance of the Pseudo-Labelling proposed method in this work with other methods proposed in the literature. The performances of the other methods are reported from (Kosiorek et al., 2019).

It can be noticed that the B-AE out-performs the K-means (Haeusser et al., 2018), AE (Bengio et al., 2007) and GAN (Radford et al., 2016) methods as it extracts encodings from data then uses it for classification as opposed to directly clustering the data. That's what allows it to have an edge over the K-means algorithm. As for the AE and the GAN they are outperformed because the B-AE uses convolutional layers vs dense layers for the AE and is more specialised than the GAN. Thus, the B-AE falls short compared to the more advanced methods like IMSAT (Hu et al., 2017).

The P-AE continues where the B-AE left-off where it can be seen that it out-performs the accuracy obtained by the IMSAT and the IIC (Ji et al., 2019) on the MNIST and CIFAR-10 datasets and achieves the second best performance on the SVHN dataset. This gain in performance comes from harnessing the dependencies between the pixels in computing the loss to improve the quality of the generated embeddings.

The SDA-AE brings no noticeable performance improvement over the P-AE even though it is based on it and incorporates data augmentation thus it seems to have a trival impact here.

Besides, it can be appreciated that the PL-AE improves by 0.3%, 3.11% and 9.21% over the previous state-of-the-art on the MNIST, CIFAR-10 and SVHN datasets respectively. This showcases that the Pseudo-Labelling method proposed in this work incorporates data augmentation in a beneficial way which allows obtaining very high accuracies that are stable across the different datasets that have been explored.

7. Conclusion

In this work, we proposed a novel method to pseudo-label images using data augmentation and we combined it with Auto-Encoders and the perceptual loss to do self-supervised image classification. The proposed method encourages the encoder to look past the data augmentation that has been applied and create rich encodings that are highly informative of the input's class. We show that when data augmentation is incorporated in a simple way it does not contribute much in improving the classification accuracy. Thus, when using the Pseudo-Labelling method improvement are seen on multiple datasets both in terms of accuracy and stability. State-of-the-art accuracy is obtained on the MNIST, CIFAR-10 and SVHN datasets with 99.3%, 60.71% and 76.48% respectively. Also, the accuracy obtained on the MNIST dataset is lower only by 0.14% than an equivalent supervised CNN.

A future extension to this work may include the exploration of different pre-trained models in the perceptual loss or the creation of data augmentation transformations that are specific and tailored to each dataset.

Acknowledgments

This work received partial funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (ERC Advanced Grant agreement No 694665 : CoBCoM - Computational Brain Connectivity Mapping) and from the French government, through the 3IA Côte d'Azur Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002.

Appendix A. Data Augmentation Tranformation Parameters

The used data augmentation transformations in this work have the following parameters :

- **Random rotation :** Rotating the image by up to 45 degrees clock-wise or counter clock-wise.
- **Affine transformation :** Composed of a rotation of up to 45 degrees, scaling of the image up or down by up to 1.5× or 0.5× respectively.
- **Crop :** Taking a random portion of the image that is of size 20×20.
- **Cutout :** Hiding one area of size 10×10 from the image.
- **Colour jitter** Changing the colours of the image with the following parameters brightness=0.8; contrast=0.8; saturation=0.8; hue=0.2

References

Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., 2007. Greedy layer-wise training of deep networks, in: Advances in Neural Information Processing Systems.

Bouayed, A.M., Atif, K., 2019. Un modèle de réseaux de neurones efficace pour la classification de données simulées pour l'identification du signal du Higgs au collisionneur hadronique du CERN. Technical Report. Computer Science Department, USTHB. [Unpublished results], https://github.com/aymene98/technical-report-DI-097-2019/blob/main/thesis.pdf.

Bouayed, A.M., Kennouche, H., 2020. Auto-Encoder with Optimised Architecture for Unsupervised Classification. Technical Report. Computer Science Department, USTHB. [Unpublished results], https://github.com/aymene98/technical-report-DI-AARN-2020/blob/main/report.pdf.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., Varoquaux, G., 2013. API design for machine learning software: experiences from the scikit-learn project, in: ECML PKDD Workshop: Languages for Data Mining and Machine Learning, pp. 108 –122.

Chen, T., Kornblith, S., Norouzi, M., Hinton, G., 2020. A simple framework for contrastive learning of visual representations, in: International Conference on Machine Learning.

Chetty, G., Bui, H., White, M., 2019. Deep learning based spam detection system, in: International Conference on Machine Learning and Data Engineering.

Dosovitskiy, A., Brox, T., 2016. Generating images with perceptual similarity metrics based on deep networks, in: Advances in Neural Information Processing Systems.

Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J., 2008. Liblinear: A library for large linear classification. Journal of Machine Learning Research .

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press. http://www.deeplearningbook.org.

Haeusser, P., Plapp, J., Golkov, V., Aljalbout, E., Cremers, D., 2018. Associative deep clustering: Training a classification network with no labels, in: German Conference on Pattern Recognition.

Hu, W., Miyato, T., Tokui, S., Matsumoto, E., Sugiyama, M., 2017. Learning discrete representations via information maximizing self-augmented training, in: International Conference on Machine Learning.

Hunter, J.D., 2007. Matplotlib: A 2d graphics environment. Computing in Science & Engineering 9, 90–95. doi:10.1109/MCSE.2007.55.

Ji, X., Henriques, J.F., Vedaldi, A., 2019. Invariant information clustering for unsupervised image classification and segmentation, in: International Conference on Computer Vision.

Kingma, D.P., Ba, J., 2015. Adam: A method for stochastic optimization, in: International Conference on Learning Representations.

Kosiorek, A.R., Sabour, S., Teh, Y.W., Hinton, G.E., 2019. Stacked capsule autoencoders, in: Advances in Neural Information Processing Systems.

[Dataset] Krizhevsky, A., 2009. Learning multiple layers of features from tiny images [Unpublished results], https://www.cs.toronto.edu/~kriz/cifar.html.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems.

Larsen, A.B.L., Sonderby, S.K., Hugo Larochelle, O.W., 2016. Autoencoding beyond pixels using a learned similarity metric, in: International Conference on Machine Learning.

[Dataset] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. IEEE .

van der Maaten, L., Hinton, G., 2008. Visualizing data using t-sne. Journal of Machine Learning Research .

Marcel, S., Rodriguez, Y., 2010. Torchvision the machine-vision package of torch, pp. 1485 – 1488. doi:https://doi.org/10.1145/1873951.1874254. https://pytorch.org/docs/stable/torchvision/index.html.

[Dataset] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y., 2011. Reading digits in natural images with unsupervised feature learning .

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S., 2019. Pytorch: An imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 8024–8035. URL: https://pytorch.org/docs/stable/index.html.

Pihlgren, G.G., Sandin, F., Liwicki, M., 2020. Improving image autoencoder embeddings with perceptual loss, in: IEEE World Congress on Computational Intelligence.

Radford, A., Metz, L., Chintala, S., 2016. Unsupervised representation learning with deep convolutional generative adversarial networks, in: International Conference on Learning Representations.

Shorten, C., Khoshgoftaar, T.M., 2018. A survey on image data augmentation for deep learning. Journal of Big Data .

Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A., 2008. Extracting and composing robust features with denoising autoencoders, in: International Conference on Machine Learning.

Wu, Y., Burda, Y., Salakhutdinov, R., Grosse, R., 2017. On the quantitative analysis of decoder-based generative models, in: International Conference on Learning Representations.

Zilong, H., Jinshan, T., Ziming, W., Kai, Z., Ling, Z., Qingling, S., 2018. Deep learning for image-based cancer detection and diagnosis - a survey. Pattern Recognition .