A Clustering-Based Deep Autoencoder for One-Class Image Classification

Matheus Gutoski[a,1], Manassés Ribeiro[a,b,2], Nelson Marcelo Romero Aquino[a,3]
André Eugênio Lazzaretti[a,4] and Heitor Silvério Lopes[a,5]

***Abstract*—Anomaly detection in images is a topic of great interest in Computer Vision. It can be defined as an One-Class problem, where the goal is to detect deviations from known patterns, which are defined as normal. Recently, Deep Learning methods became popular due to their performance on classification tasks. This works presents an image anomaly detection classifier based on a previously known method, the Deep Embedded Clustering, which is based on a Deep Autoencoder. We show the effectiveness of the method through three different experiments. Results suggest that the method improves classification performance when compared to a Stacked Denoising Autoencoder in the image anomaly detection context.**

I. INTRODUCTION

Anomaly detection, also known as novelty detection, is a well known problem in the Pattern Recognition area [1]. It can be defined as an One-class Classification (OCC) problem, where the normal or expected patterns are previously known, and the anomalous patterns to be recognized by the classifier are all patterns that are scarce or not present in the training data [2]. The classification of anomalous behavior in images has been a subject of great interest [3]. Particularly, in recent years, many efforts have been focused to detect anomalous events in images and in automated video surveillance [4], [5], [6].

In the OCC problem the known class (normal patterns) is referred as positive or target class and has a large number of instances instances (training set). The negative class is characterized by instances that do not form a statistically-representative sample [7] in the training set. In this sense, a novelty detection model can be approached as an OCC problem [8]. Consequently, we assume that the positive class of the OCC is the normal class (with known instances) and the negative class is the abnormal class.

Several OCC methods have been extensively used for novelty detection problems. For instance, the one-class Support Vector Machine was used by Xu and Ricci [5] to predict anomalous frames in video. Also, a similar one-class approach, named space-time Markov Random Field model, was devised by Kim and Grauman [9] to detect abnormal activities. In [10], a probability model is learned for OCC, and the classifier is unsupervised, without the need to label the training data. It is also possible to include some *a priori* knowledge about the application. In [11], an unsupervised Deep Belief Network (DBN) was trained to extract a set of features in a relatively low-dimensional space, and a one-class classifier was trained with the features learned by the DBN. In general, one-class classifiers can be inefficient for modeling decision surfaces in large and high-dimensional datasets. However, by combining the classifier with a DBN, it is possible to reduce redundant features and improve the performance for standard OCC datasets.

Deep Learning (DL) methods have been investigated for several problems, and turn out to be very effective for recognition tasks. In fact, DL methods have already achieved the state-of-the-art performance for object recognition in supervised classification of images and videos [12], [13]. A possible reason for such a high performance is that they can learn features automatically, and with superior discriminatory power for image representation, when compared to hand-crafted image descriptors [14]. However, in the context of OCC, DL methods are still in early stages of development, and this is one of the motivations of this work. Therefore, Stacked Denoising Auto-Encoders (SDAEs) [15] can be an interesting option for OCC problems, since they can be trained using only the positive class, as presented in [5], [16].

However, SDAEs may be inefficient when applied directly to OCC problems, since the feature space mapping of the bottleneck can be sparse, i.e. it does not guarantee a compact representation of the data in the bottleneck, which is an essential characteristic in OCC problems [8]. During the training process, the SDAE's bottleneck is mapped to minimize the reconstruction error of all instances, without attempting to achieve a dense representation in the feature space (bottleneck space). Thus, positive (normal) and negative (anomalous) classes can be, eventually, overlapped in the bottleneck representation. In fact, this issue hinders the delimitation of the positive class and compromises the classification performance, as illustrated in Fig. 1 (a).

Recently, some approaches were proposed to circumvent the above-mentioned problem, so as to modify the optimization procedure used in SDAEs, by simultaneously minimizing data reconstruction and imposing compactness in the bottleneck representation [17], [18]. Such methods were initially developed for clustering problems, where data representation and cluster centers are updated jointly, from which a stable perfor-

[a]Graduate Program in Electrical and Computer Engineering, Federal University of Technology – Paraná (UTFPR), Av Sete de Setembro 3165, Curitiba (PR), 80230-901, Brazil.
[b]Catarinense Federal Institute of Education, Science and Technology – (IFC), Rod. SC 135 km 125, Videira (SC) 89560-000, Brazil.
[1]E-mail: matheusgutoski at gmail.com
[2]E-mail: manasses at ifc-videira.edu.br
[3]E-mail: nmarceloromero at gmail.com
[4]E-mail: lazzaretti at utfpr.edu.br
[5]E-mail: hslopes at utfpr.edu.br

978-1-5386-3734-0/17/$31.00

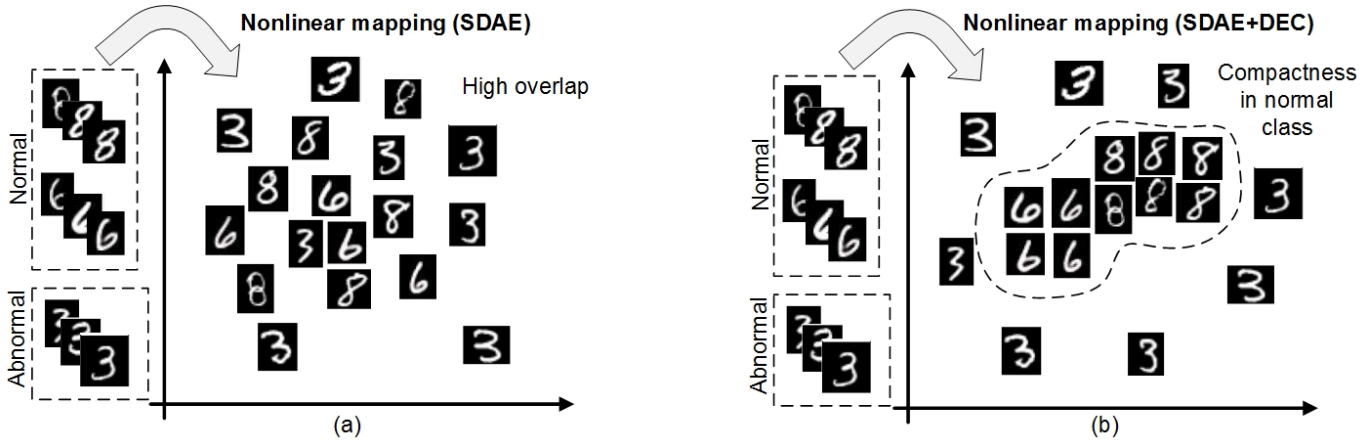


Fig. 1. General Overview of the Proposed Method. (a) With the SDAE, the normal class is overlapped by the anomaly class, making the discrimination between these classes more complex. (b) By including the DEC in the SDAE, it is possible to include compactness in the representation of normal class, improving the discrimination between normal and abnormal classes. The presented classes and mapping are merely illustrative.

mance of clustering is achieved, making the new representation more compact with respect to the cluster centers. Particularly, the method proposed by [18] consists in performing the clustering procedure using data mapped by the bottleneck of a SDAE. The proposed algorithm – defined as Deep Embedded Clustering (DEC) – clusters data by simultaneously learning a set of cluster centers in the feature space and the parameters (weights) of the SDAE that maps data points into the bottleneck with compact representation, as illustrated in Figure 1(b).

Inspired by the method presented in [18], in this work we propose the use of the DEC as an OCC in the novelty detection context, mainly due to its compactness characteristic, as summarized in Fig. 1. The working hypothesis is that DEC is able to learn normal patterns (positive class) and, therefore, we hypothesize that the bottleneck can be used for discriminating anomalous pattern (negative class). Differently from OCC methods using SDAEs [5], the feature extraction in the bottleneck provides compact representations with the DEC procedure, improving the final classification performance – which is the main contribution of this work. We present the advantages of the compactness in the bottleneck by comparing the classification performance of the standard SDAE and the DEC using three image datasets (STL-10, MNIST and NOTMNIST). Additionally, some relevant steps of the DEC procedure are analyzed and discussed.

This paper is organized as follows. Section II addresses the fundamental topics related to OCCs, SDAEs and DEC. Section III describes in detail the proposed method. Section IV presents how the experiments were done, their results and a short discussion. Finally, Section V reports the general conclusions drawn, and suggests future research directions.

II. Theoretical Aspects

A. One-Class Classification Methods

There are three different approaches to use one-class classifiers in novelty detection problems [8]. The first is based on models that estimate the probability density function of input patterns (density methods). From the probability density function, it is possible to establish if a given input pattern is a novelty or not, based on its probability value. In this first group, parametric estimators based on Gaussian Mixture Models and nonparametric estimators based on Parzen Windows can be highlighted [8].

The second group comprises models with imposed boundaries upon the training dataset, assuming an unknown distribution. Therefore, a boundary optimization problem is solved in order to represent the data. In this group, techniques based on Nearest Neighbor and Support Vector Data Description (One-Class Support Vector Machine) can be highlighted.

Finally, the third group concerns reconstruction methods. In this group, *k-means* clustering and autoencoder (AE) are the most used techniques [8]. From a *k-means* representation, it is possible to define if a given input pattern is a novelty or not, based on the distance from the input pattern to previously defined clusters. As for the autoencoder, the reconstruction error can be used as a metric to define the novelty degree. In the following subsections, the autoencoder for OCC is detailed.

B. AE and Stacked Denoising Auto-Encoders

The autoencoder model was introduced by Rumelhart et al. [19] and, later, popularized by Vincent et al. [15] with the SDAEs. AE is regarded as an unsupervised fully connected one-hidden-layer neural network to learn from unlabeled datasets. The idea is that the AE is trained to reconstruct the input pattern at the output of the network. An AE takes an input $\mathbf{x} \in \mathbb{R}^d$ and first maps it to the latent representation (hidden layer) $\mathbf{h} \in \mathbb{R}^{d'}$ using the mapping function $\mathbf{h} = f_\Theta = \sigma(\mathbf{W}\mathbf{x} + b)$ with parameters $\Theta = \{\mathbf{W}, b\}$. For reconstructing the input, a reverse mapping of $f : \mathbf{y} = f_{\Theta'}(h) = \sigma(\mathbf{W'}\mathbf{h}+b')$ with $\Theta' = \{\mathbf{W'}, b'\}$ is used. The parameters $\mathbf{W}$ learnt from the input layer to the hidden layer compose the encoder and the parameters $\mathbf{W'}$ learnt from the hidden layer to the output

layer define the decoder. The decoder parameters are normally related to the parameters in the encoder by $\mathbf{W'} = \mathbf{W}^T$ [20].

Training an AE does not require label information of the input data. It uses the back propagation algorithm to minimize the reconstruction error e between each input $\mathbf{x}_i$ and the corresponding output $\mathbf{y}_i$, by adjusting the parameters of the encoder $\mathbf{W}$ and the decoder $\mathbf{W'}$, as shown in Equation 1:

$$e(\mathbf{x}, \mathbf{y}) = \frac{1}{2N} \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{y}_i\|_2^2. \tag{1}$$

The drawback of the above formulation is that, without additional constraints, the final mapping is the identity. There are different approaches proposed in the literature to circumvent such limitation. Denoising AE (DAE) [21] are one of the most used models. A DAE reconstructs the input by using its partially corrupted version, where the corrupted version is obtained by adding some amount of noise, distributed according to the characteristics of the input vector. Deep architectures can be built stacking several DAEs as a SDAE. The SDAE training process is performed in two steps. First, a pretraining is performed layer by layer, where they are learned as a greedy-wise training. Later, a fine-tuning process is used to adjust weights for a better quality reconstruction.

C. Deep Embedded Clustering

The deep embedded clustering used in this work applies the method presented in [18]. Basically, it performs the clustering using data mapped by the bottleneck (feature space) of a deep autoencoder network. The proposed algorithm clusters data by simultaneously learning a set of K cluster centers $\{\boldsymbol{\mu}_k\}_{k=1}^{K}$ in the feature space and the parameters (weights) of the deep autoencoder that maps data points into bottleneck.

Given the initial mapping provided by the autoencoder and the initial K cluster centers $\{\boldsymbol{\mu}_k\}_{k=1}^{K}$, the idea of the algorithm proposed by [18] is to iteratively alternate between two main steps: compute a soft assignment between the embedded points and the cluster centroids; update the deep mapping (weights of the autoencoder) and refine the cluster centroids ($\{\boldsymbol{\mu}_k\}_{k=1}^{K}$) by learning from current high confidence assignments using an auxiliary target distribution.

To do so, the optimization is performed by minimizing the Kullback-Leibler (KL) divergence loss between soft assignments q_{ij} and auxiliary distribution p_{ij}, as proposed by [22], [23]:

$$L = KL(P \| Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{2}$$

The soft assignment is defined as the probability of assigning a mapped sample $\mathbf{z}_i$ (in the bottleneck) to cluster $\boldsymbol{\mu}_k$, i.e.:

$$q_{ij} = \frac{\sum_k \left(1 + \|\mathbf{z}_i - \boldsymbol{\mu}_k\|^2\right)}{1 + \|\mathbf{z}_i - \boldsymbol{\mu}_j\|^2}. \tag{3}$$

On the other hand, the auxiliary distribution is calculated using soft assignments with the following relationship:

$$p_{ij} = \frac{q_{ij}^2 / \sum_m q_{mj}^2}{\sum_k \left(q_{ik}^2 / \sum_m q_{mk}^2\right)}. \tag{4}$$

Using the proposed auxiliary distribution, it is possible to improve cluster purity, put more emphasis on data points assigned with high confidence, and normalize loss contribution of each centroid to prevent large clusters from distorting the hidden feature space [18].

The cluster centers are then optimized jointly to the autoencoder parameters using Stochastic Gradient Descent with momentum and the standard backpropagation to compute parameter's gradient. The gradients are defined as:

$$\frac{\partial L}{\partial \mathbf{z}_i} = 2 \sum_k \frac{(p_{ij} - p_{ij})(\mathbf{z}_i - \boldsymbol{\mu}_k)}{\left(1 + \|\mathbf{z}_i - \boldsymbol{\mu}_k\|^2\right)}, \tag{5}$$

$$\frac{\partial L}{\partial \boldsymbol{\mu}_k} = -2 \sum_i \frac{(p_{ij} - p_{ij})(\mathbf{z}_i - \boldsymbol{\mu}_k)}{\left(1 + \|\mathbf{z}_i - \boldsymbol{\mu}_k\|^2\right)}. \tag{6}$$

III. Proposed Method

A. Overview

This work studies the effects of the Deep Embedded Clustering [18] method applied to the one-class image classification problem. Figure 2 shows an overview of the proposed method. Each step of the method is explained in the following Sections.

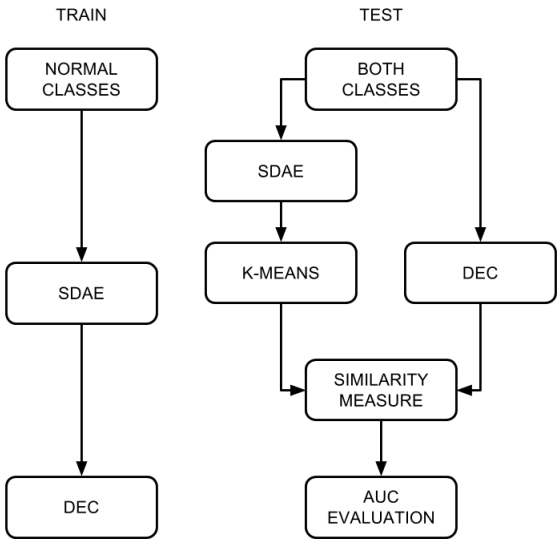


Fig. 2. Overview of the proposed method. In the train phase, only the known class, i.e., the normal class, is used to train the SDAE. The DEC network is initialized using the weights learned by the SDAE encoder and is further trained to simultaneously learn cluster centers and feature representations. The test phase consists of forwarding the test image set through both networks and use the bottleneck features to evaluate them separately. We measure the similarity between feature vectors and cluster centers by using Equation 3. Evaluation is done by using the area under the ROC curve.

First, data from different datasets are re-labeled as normal or abnormal according to different criteria, discussed in Sections IV-C, IV-D and IV-E. Then, a SDAE is trained for each experiment using the re-labeled normal class. Next, the SDAE is further trained using the DEC method. Last, the bottleneck data is used to classify both normal and anomalous images according to a similarity measure to the nearest cluster centroid. The network was implemented using the Caffe Deep Learning Framework [24], as provided by [18].

B. Training the Stacked Denoising Autoencoder

The SDAE chitecture proposed by [18] was trained layer by layer followed by a fine-tuning step. The encoder contains three hidden layers, with 500-500-2,000 neurons in each layer, respectively, followed by a bottleneck layer, which has a variable amount of neurons according to the experiment. The decoder follows the reverse order of neurons presented in the encoder. Inputs are corrupted by using dropout layers [25] with a probability of 20%. Weights are initialized using a Gaussian distribution, and the Rectified Linear Unit (ReLU) is used as the activation function. The Stochastic Gradient Descent method is used to perform the optimization.

C. DEC optimization

In this step, we use DEC to simultaneously learn cluster centers and map the features into a compact representation. This is done by using the weights learned during the SDAE training to initialize a Deep Neural Network with the same architecture of the SDAE, discarding the decoder part [18].

In this phase, the network is trained for 150 epochs. However, we evaluate the model after every epoch to study the effects of the method throughout the whole training phase. We only report the results obtained in the best number of epochs for each experiment. The number of groups to cluster the data was set according to the experiment being performed, as explained in Section IV.

D. Classification and Evaluation

The classification task has been done using a similarity score between the bottleneck features and the learned centroids (as shown in Equation 3). For each test image, we compute the similarity score between the bottleneck data and every cluster center. This score provides means to classify whether an instance belongs to a group, i.e., is close enough to a centroid. Only the centroid with the highest score is used to classify each sample. The Area under the Receiver Operating Characteristic Curve (AUC) is used to measure the quality of the classifier. This allows the evaluation of a classifier regardless of the classification threshold.

To evaluate the standard SDAE, we cluster the bottleneck data using the *k-means++* method [26], initialized 100 times. The number of clusters parameter was set according to each experiment.

IV. Experiments and Results

A. Datasets

The STL-10 dataset [27] originally contains 96x96 pixels color image data divided into 10 classes: airplane, bird, car, cat, deer, dog, horse, monkey, ship, truck. The dataset is divided as follows: unsupervised train, containing $100,000$ unlabeled instances from all 10 classes and some variations; supervised train, containing 500 labeled images per class; test, containing 800 images per class. In this work, only the supervised train set and the test set were used.

MNIST is a well known handwritten digit recognition dataset introduced in [28]. The handwritten digits range from 0 to 9 (10 classes). The train set contains 60,000 images, whereas the test set contains 10,000 images. Images are in gray scale and have a dimension of 28×28 pixels.

NOTMNIST is a printed character dataset containing 10 classes of letters from A to J. It contains 200,000 images in the training set, 10,000 images in the validation set and 10,000 images in the test set. Images are in grayscale and have a dimension of 28×28 pixels. Only the test set was used in this work.

B. Experiments

All three experiments reported below were modeled as an OCC problem. The SDAE is trained using only the known class, i.e., the normal class. Class labels were used only during the evaluation phase.

C. Experiment #1: MNIST 8-0

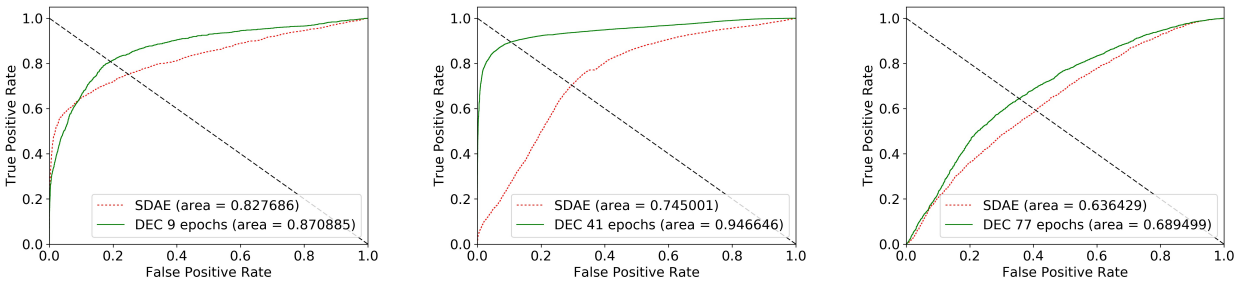
In this experiment we divided the MNIST dataset as follows: digits 0 and 8 were considered the normal class, and the remaining data were considered anomalous. Therefore, final train and test sets had 11,774 (normal) and 10,000 images (20% normal and 80% anomalies), respectively. The number of neurons in the SDAE bottleneck layer was set to 10 and the number of groups to 2.

Figure 3(a) shows the ROC curves resulting from the SDAE before and after applying the DEC method. The best result was obtained in the 9^{th} refinement epoch. Results show a small improvement over the standard SDAE (AUC increased 0.0432).

D. Experiment #2: MNIST – NOTMNIST

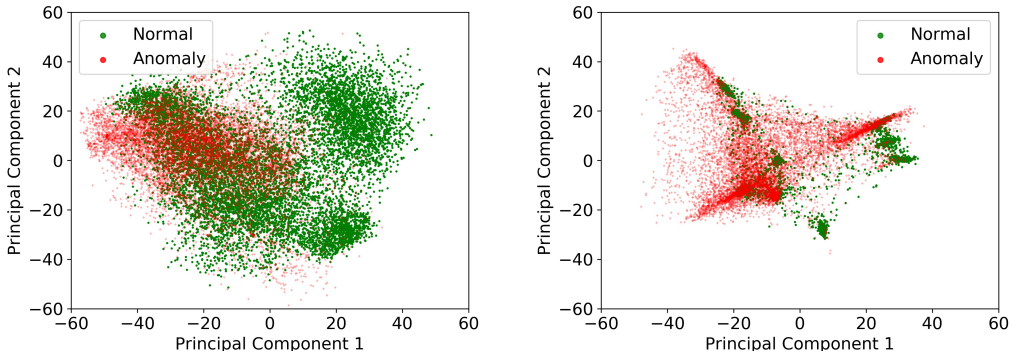
In this experiment we merged the MNIST and NOTMNIST datasets. Classes that contain numbers (MNIST) were considered normal, whereas the printed characters (NOTMNIST) were considered anomalies. Both datasets were normalized in the same range before applying the method. The final train and test sets had 60,000 images (normal) and 20,000 images (50% normal and 50% anomalies), respectively. In this experiment, the number of neurons in the SDAE bottleneck layer was set to 10 and the number of groups to 10.

Figure 3(b) shows the ROC curves before and after applying the DEC method to the SDAE. The best result was obtained after 41 epochs. A significant improvement has been achieved (AUC increased 0.2016).



(a) (b) (c)

Fig. 3. ROC curves using SDAE and DEC (best epoch) for each experiment. Figure 3(a) shows the ROC curves for the experiment #1. Figures 3(b) and 3(c) show the ROC curves for the experiments #2 and #3, respectively.



(a) (b)

Fig. 4. Two-component PCA representation of the test dataset used in experiment #2. Each dot represents an image from the dataset. Figure 4(a) shows the normal and anomalous bottleneck features using the SDAE. Figure 4(b) shows the features after 41 epochs of training using the DEC method. Some regions may appear overlapped due to the distortion caused by the two-dimensional PCA representation.

In order to visualize results, we used Principal Component Analysis (PCA) in the data, and the first two components were plotted in Figure 4. The feature representation of the default SDAE is shown in Figure 4(a). Figure 4(b) shows the bottleneck features in epoch 41. The figure shows that after refinement using DEC, the normal class (green) forms more compact clusters, making the anomalies (red) more discernible from the normal patterns.

E. Experiment #3: STL10

This experiment uses the STL-10 train and test sets. The data was divided as follows: classes containing animals were considered normal (6 classes), while the remaining classes were considered as anomalies (4 classes). The final train and test sets had 3,000 (normal) and 8,000 images (60% normal and 40% anomalies) images, respectively. In this experiment, the number of neurons in the SDAE bottleneck layer was set to 100 and the number of groups to 6. We choose 100 neurons instead of 10 to compensate for the complexity of the images in this dataset.

Figure 3(c) shows the ROC curves with and without the DEC method. The best result was obtained after 77 epochs. An improvement was observed in terms of AUC (AUC increased 0.053).

The overall results for all experiments are shown in Table I. The DEC method leads to improvements for all experiments performed in this work. The most significant result was that of the experiment #2.

TABLE I
EXPERIMENTAL RESULTS (AUC) FOR THE SDAE AND DEC IN THREE DIFFERENT EXPERIMENTS.

Experiment	SDAE	DEC
#1 - MNIST	0.8276	0.8708
#2 - NOTMNIST	0.7450	0.9466
#3 - STL10	0.6364	0.6894

V. CONCLUSIONS

The anomaly detection problem in images has been a subject of growing interest in recent years. Anomaly detection can be defined as an OCC problem, where the objective is to detect deviations from the normal or expected patterns. This work presented a study of the applicability of the Deep Embedded Clustering [18] method in the image OCC problem.

The working hypothesis is that by having compact representations in the bottleneck, anomalies become more distinguishable from normal patterns. In this work, we have studied the importance of having compact representations when solving an OCC problem, and found that it can significantly improve the classification performance.

One way to achieve compactness in the representations is by using DEC. The method automatically learns cluster centers and compact feature representations in the normal class, allowing the classifier to discriminate anomalous instances more efficiently. By exploring this compactness characteristic, our experiments have shown a better separability between normal and anomalous patterns as compactness increases.

We compare the method with a Stacked Denoising Autoencoder in three different experiments using different public image datasets. In the first experiment, we re-labeled the MNIST dataset such as 0 and 8 digits belong to the normal class and the remaining of the digits belong to the anomaly class. In the second experiment, we merged the MNIST and NOTMNIST datasets such as that numbers belong to the normal class, whereas printed characters belong to the anomaly class. The third experiment uses a re-labeled STL-10 dataset, such that classes containing animals belong to the normal class, and non-animals belong to the anomaly class.

The results obtained in this work suggest that the Deep Embedded Clustering method is promising anomaly detection context. Future work comprises refinements of the method and its application to real-world anomaly detection datasets. Also, a Hybrid Autoencoder architecture containing convolutional layers should also be explored in future works.

ACKNOWLEDGEMENTS

M. Gutoski would like to thank CAPES for the scholarship. M. Ribeiro would like to thank the Catarinense Federal Institute of Education, Science and Technology and IFC/CAPES/Prodoutoral for the scholarship. N. Aquino would like to thank the Organization of the American States, the Coimbra Group of Brazilian Universities and the Pan American Health Organization. H.S.Lopes would like to thank to CNPq for the research grant number 440977/2015-0. All authors would like to thank to NVIDIA for the donation of a GPU used in this work.

REFERENCES

[1] A. A. Sodemann, M. P. Ross, and B. J. Borghetti, "A review of anomaly detection in automated surveillance," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1257–1272, 2012.

[2] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[3] A. A. Sodemann, M. P. Ross, and B. J. Borghetti, "A review of anomaly detection in automated surveillance," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 42, no. 6, pp. 1257–1272, 2012.

[4] Y. Yuan, J. Fang, and Q. Wang, "Online anomaly detection in crowd scenes via structure analysis," *IEEE Transactions on Cybernetics*, vol. 45, no. 3, pp. 562–575, 2015.

[5] D. Xu and E. Ricci, "Learning deep representations of appearance and motion for anomalous event detection," in *Proc. British Machine Vision Conference*, 2015, pp. 1–12.

[6] C. Chen, Y. Shao, and X. Bi, "Detection of anomalous crowd behavior based on the acceleration feature," *IEEE Sensors Journal*, vol. 15, no. 12, pp. 7252–7261, 2015.

[7] D. M. Tax and R. P. Duin, "Uniform object generation for optimizing one-class classifiers," *Journal of machine learning research*, vol. 2, no. Dec, pp. 155–173, 2001.

[8] D. M. J. Tax, "One-class classification," Ph.D. dissertation, Technische Universiteit Delft, 2001.

[9] J. Kim and K. Grauman, "Observe locally, infer globally: A space-time mrf for detecting abnormal activities with incremental updates," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2921–2928.

[10] T. Xiao, C. Zhang, and H. Zha, "Learning to detect anomalies in surveillance video," *IEEE Signal Processing Letters*, vol. 22, no. 9, pp. 1477–1481, 2015.

[11] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning," *Pattern Recognition*, vol. 58, pp. 121–134, 2016.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., 2012, vol. 25, pp. 1097–1105.

[13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, June 2015.

[14] H. A. Perlin and H. S. Lopes, "Extracting human attributes using a convolutional neural network approach," *Pattern Recognition Letters*, vol. 68, no. 2, pp. 250–259, 2015.

[15] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010.

[16] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 733–742.

[17] C. Song, F. Liu, Y. Huang, L. Wang, and T. Tan, "Auto-encoder based data clustering." in *CIARP (1)*, ser. Lecture Notes in Computer Science, J. Ruiz-Shulcloper and G. S. di Baja, Eds., vol. 8258. Springer, 2013, pp. 117–124.

[18] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, pp. 478–487.

[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986.

[20] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Proc. 21^{th} International Conference on Artificial Neural Networks*, vol. I, 2011, pp. 52–59.

[21] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25^{th} International Conference on Machine Learning*, 2008, pp. 1096–1103.

[22] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[23] L. van der Maaten, "Learning a parametric embedding by preserving local structure," *RBM*, vol. 500, no. 500, p. 26, 2009.

[24] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. of 22^{nd} ACM International Conference on Multimedia*, 2014, pp. 675–678.

[25] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[26] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[27] A. Coates, H. Lee, and A. Y. Ng, "An analysis of single-layer networks in unsupervised feature learning," in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 215–223.

[28] 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, 1998.