A DEEP CONVOLUTIONAL AUTO-ENCODER WITH EMBEDDED CLUSTERING

A. Alqahtani, X. Xie, J. Deng, M.W.Jones

Department of Computer Science, Swansea University, Swansea, UK

ABSTRACT

In this paper, we propose a clustering approach embedded in deep convolutional auto-encoder (DCAE). In contrast to conventional clustering approaches, our method simultaneously learns feature representation and cluster assignment through DCAE. DCAE have been found effective in image processing as it fully utilizes the properties of convolutional neural networks. Our method consists of clustering and reconstruction objective functions. All data points are assigned to their new corresponding cluster centers during the optimization, after that, clustering centers are iteratively updated to obtain a stable performance of clustering. The experimental results on MNIST and USPS datasets show that the proposed method substantially outperforms deep clustering models in term of clustering quality.

Index Terms— deep learning, deep convolutional auto-encoder, embedded clustering.

1. INTRODUCTION

Clustering as one of unsupervised machine learning approach, it aims to group a set of unlabelled data based on homogeneous patterns in the given feature space. Traditional clustering algorithms are limited to demonstrate satisfying performance as the dimensionality goes higher. Therefore, dealing with high-level representation provides beneficial components, benefiting the achievement of such a clustering task. As there is no supervision knowledge to provide information of categories labels, representative features with compact clusters are much more beneficial. Deep auto-encoder (DAE) and deep convolutional auto-encoder (DCAE) are unsupervised models for representation learning. They map inputs into new space representation, allowing to obtain useful features through encoding procedure. The data is projected into a set of feature spaces, using the encoding part, from which the decoding part reconstructs the original data. The training is performed in an unsupervised manner via minimizing the differences between original data and reconstructed data with distance metrics. The major difference between DAE and DCAE is that the former adopts fully-connected layers to reconstruct signal globally while the later utilizes local information to achieve the same objective. DCAE can benefit from such local model. These methods have been exploited for the purpose of clustering where features learned through deep networks (e.g. AE or CAE) provide an abstracted latent representation which is used for clustering analysis. Existing works can be classified into four categories summarized in Table 1. *Huang et al.* [2] and *Tian et. al* [1] used AE to learn

Method	Separated Clustering	Embedded Clustering
AE	*Tian* [1], *Huang* [2]	*Song* [3], *Xie* [4]
CAE	*Li et. al* [5]	*Guo et. al* [6]

Table 1. Deep Clustering Methods

a lower dimensional representation space, seeking to obtain effective features used for clustering, thereafter, k-means is applied to cluster the obtained features while Lia at el. [5] utilizes the CAE to learn representation, thereafter, the decoder part is neglected and a soft k-means model is added on top of the encoder to make a unified clustering model. Even though such scheme takes advantages of a deep neural network to map the original data into a representative feature space followed by clustering analysis, feature space learning and clustering are two separated steps procedures, objectives of which are not optimized jointly. *Song et al.* [3] and *Xie et al.* [4] embedded clustering objective into an AE framework, while *Guo et al.* [6] recently propose clustering with CAE. Developing embedded clustering approach in a deep network allows extracting latent features and clustering assignments simultaneously. This scheme usually leads to a more compact latent feature space.

In this paper, we present a clustering approach embedded into a DCAE framework which aims to learn feature representation and cluster assignment simultaneously. In contrast to conventional clustering approaches, our method makes use of representation learning with deep neural networks, which helps to find a compact and representative latent feature spaces for further recognition tasks. Most of existing methods fundamentally rely on pre-training the parameters, using different settings, while we train our model in an end-to-end way in fixed settings without any pre-training or fine-tuning procedures, enabling faster training process. In contrast to Guo work, our proposed method differs in several key respects. First, for clustering approach, instead of clustering with KL divergence, we apply an objective function that restricts the distance between learned feature representations, in a latent space, and their identical centroids, producing a stable representation, which is appropriate for clustering process. Accordingly, the centroids are iteratively updated. Second, our work particularly differs in the term of architecture, cost functions, and optimization. Finally, our results show that our model yield substantially better for both reconstruction and clustering quality. We evaluate our proposed methods model on MNIST and USPS datasets and compare our method with three baselines, showing that our method substantially outperforms others in both reconstruction and clustering quality.

2. METHOD

The proposed approach embeds K-means clustering algorithm into a DCAE framework which is trained in a fully unsupervised manner. The architecture is shown in Fig. 1. It consists of two objective functions, one minimizes the distance between feature representations and their identical cluster centers, and another minimizes the reconstruction error. Both two objectives are simultaneously optimized.

2.1. Deep Convolutional Auto-encoder (DCAE)

In contrast to DAE model, DCAE [7] uses convolutional and deconvolutional layers instead of fully connected layers. DCAE can be better appropriate in image-processing tasks because it takes advantage of the convolutional neural networks (CNN) properties [8]. Local connections and parameter sharing distinguish CNN to have a property in translation latent features [9]. In the encoding part, convolutional layers are used, as feature extractors, to learn features through mapping the data into an internal layer. A latent representation of the n^{th} feature map of the existing layer is given by the following form:

$$h^n = \sigma(x * W^n + b^n) \tag{1}$$

where W identifies the filters and b is the corresponding bias of the n^{th} feature map, σ is the activation function (e.g. sigmoid, ReLU), and $*$ denotes the 2D convolution operation.

In contrast, the deconvolutional layers invert this process and reconstruct the latent representation back to its original shape, so this process maps the obtained features into pixels [10] by using the following form:

$$y = \sigma(\sum_{n \in H} h^n * \tilde{W}^n + c) \tag{2}$$

where H denotes the group of latent feature maps, $\tilde{W}$ is the flip operation over both dimensions of the weights, c is the corresponding a bias, σ is the activation function, and $*$ denotes the 2D convolution operation.

DCAE allows extracting latent representation through its internal layer by minimizing the reconstruction error. We use the cross-entropy (logistic) loss via Eq.(3) because experiments have shown that the euclidean (L2) loss function is not robust to convolutional neural networks designed with deconvolutional layers, and networks trained with perceptual loss tend to produce much better results [11–13]. In a like manner of standard networks, the backpropagation method computes the gradient of the error with respect to all parameters.

$$E_1 = -\frac{1}{N} \sum_{n=1}^{N} (y_n \log \widehat{y}_n + (1 - y_n) \log(1 - \widehat{y}_n)) \tag{3}$$

Where $\widehat{y}$ is the pixel value of the reconstructed image, and y is the pixel value of the target image. For further detail of CAE readers can refer to [7].

2.2. Clustering Embedded on Deep Convolutional Auto-encoder

Using DCAE model described in Section 2.1, we now utilize the strength of such model as training procedure for feature transformations. The goal of our clustering model is to learn feature representation and cluster assignment simultaneously. Using DCAE, as features extractor, supports the achievement of such a clustering process. This idea allows clustering method to deal with learned features instead of raw data. We follow [3] to develop deep clustering model, instead of classic AE, we apply DCAE for clustering task. Although DCAE provides an effective representation in a new latent space, it does not internally impose compact representation constraints using clustering. Therefore, we add a clustering objective function to the DCAE framework, which minimizes the distance between data samples and assigned centroids in latent space follows [3]:

$$E_2 = \lambda \cdot \frac{1}{2N} \sum_{n=1}^{N} \parallel h^t(x_n) - c_n^* \parallel^2 \tag{4}$$

where N denotes the number of samples, λ is clustering weight-parameter that control the contribution percentage of clustering cost function in the overall cost function Eq.(5), $h^t(*)$ is the internal representation obtained by the encoder mapping at the t^{th} iteration, (x_n) is the n^{th} sample in the dataset, and c_n^* is the assigned cluster center to the n^{th} sample. The overall cost function is a combination of two parts: the first part is essentially cross-entropy loss minimizing the reconstruction error, while the second part is clustering objective function minimizing the distance between data representations in a latent space and their corresponding cluster centers.

$$\min_{W,b} E_1 + E_2 \tag{5}$$

2.3. Optimization

At each epoch, our model optimizes two components using stochastic gradient descent and backpropagation: (1) CAE parameters as well as mapping function h, and (2) cluster centers c. At each epoch, the model optimizes the mapping function h, while keeps the cluster centers fixed at c. Thereafter, each obtained new internal representation is assigned to the closest centroid, following [3], this is defined as:

$$c_n^* = arg \min_{c_m^{t-1}} \parallel h^t(x_n) - c_m^{t-1} \parallel^2 \tag{6}$$

where c_m^{t-1} denotes the cluster centers computed at the previous epoch. After each internal representation is assigned to the closest cluster center, the cluster center is updated using the sample assignment computed in the previous epoch via the following equation as [3]:

$$c_m^t = \frac{\sum_{x_n \in c_m^{t-1}} h^t(x_n)}{\sum c_m^{t-1}} \tag{7}$$

where c_m^{t-1} is all samples that belong to the m^{th} cluster at the previous epoch, and $\sum c_m^{t-1}$ is the number of samples that belong to the m^{th} cluster.

2.4. Architecture

We utilize the architecture of classic DCAE. Our contributions to the DCAE architecture are the following. First, we exploit the learned features via the internal layer and feed it to clustering loss which minimizing the distance between data points and their assigned cluster centers, embedding clustering techniques in a DCAE framework. Second, instead of optimizing CAE to reach optimal reconstruction, we sequentially optimize the mapping function h and cluster centers to obtain efficient clustering results.

For MNIST, we adopt the base architecture proposed in [14], instead of two-loss function to minimize the reconstruction error, we only use cross-entropy loss as previous studies have shown that euclidean loss function is not robust to convolutional neural networks designed with deconvolutional layers [11–13]. Also with only the cross-entropy loss, our experiments have shown that only cross-entropy reconstruction loss can provide good training convergence. The network architecture consists of two convolutional layers with filter sizes of 9×9 with 8 kernels in the first convolutional layer and 4 kernels in the second convolutional layer. This followed by two fully-connected layers, which have 250 neurons and 10 neurons respectively, in the encoding part. In the decoding part, a single fully-connected layer of 250 neurons followed by two deconvolutional layers. The first deconvolutional layer consists of 4 kernels with the size of 12×12, and the second deconvolutional layer consists of 4 kernels with the size of 17×17. The final architecture of our deep clustering model embedded in DCAE is presented in Fig. 1.

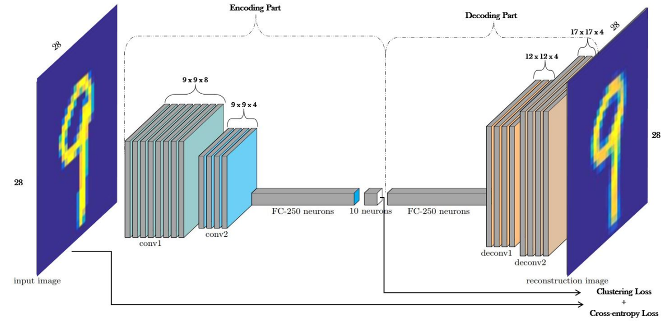

Fig. 1. The architecture of our proposed model For MNIST.

3. EXPERIMENT AND DISCUSSION

The proposed method was implemented using MatConvNet [15] and evaluated on *MINST* and *USPS* datasets. The model was trained end-to-end in an unsupervised manner. There is no pre-training and fine-tuning procedures involved. All weights were initialized using *Xavier* method [16] and biases were set to 0, and the cluster centres is initialized randomly. Stochastic gradient descent with mini-batch was used where each batch contains 100 random shuffled images. We set λ, the clustering weight-parameter that controls the loss contribution percentage of clustering error to 0.2.

CAE is trained to transform the data into latent representation and then reconstruct the original input or obtain an optimal approximation of the underlying data representation by minimizing the reconstruction error. Some examples of original inputs and reconstruction images obtained by our model are demonstrated in Fig. 3, allowing to visually differentiate and evaluate the reconstruction quality of the proposed model. In Fig. 3, the reconstructed images (bottom row) look qualitatively identical to original ones (top row) with certain levels of blurring, which helps to capture common patterns instead of subtle details at local regions for reconstruction. It is noteworthy that due to such smoothness the digit 5 has similar structure with digit 6. One reasonable explanation is that the proposed method is designed for unsupervised representation learning with signal reconstruction objective, where such supervision information *i.e.* differentiating 5 and 6 are no available to aid forming discriminative features.

To evaluate the cluster quality, two evaluation metrics, accuracy (ACC) and normalized mutual information (NMI) were computed. We compared our method with three baseline methods, *DEC* [4], AEC [3] and DCEC [6], the results are summarized in Table. 2. Our proposed method outperforms the baseline methods by a significant margin on both ACC and NMI metrics, where 84.97% and 92.14% were achieved respectively. Especially, proposed method substantially outperforms the second place by 6.85% which also uses CAE approach with jointed clustering loss. Fig. 4 shows that both the changes of ACC and NMI with the number of training cy-

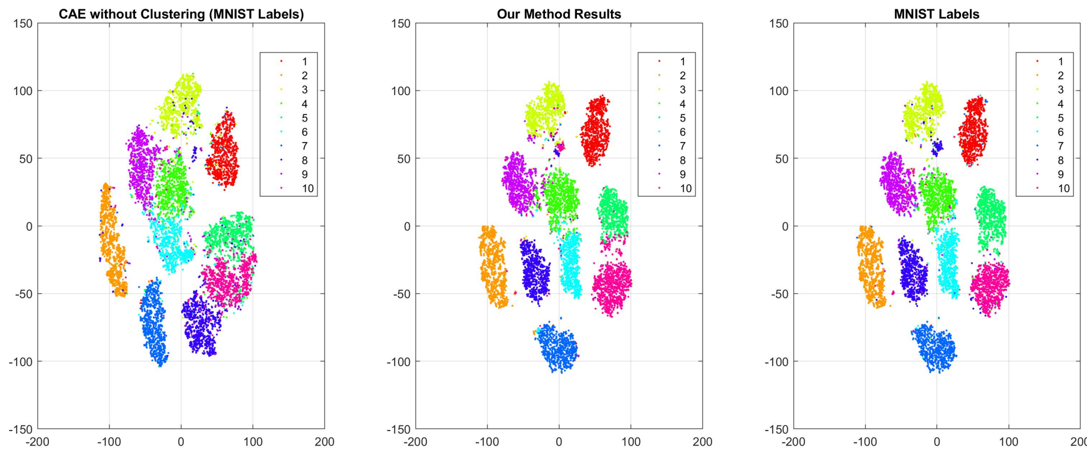

Fig. 2. Visualization of Latent Representation on MNIST Testing Set. Left: CAE without clustering loss; Middle: CAE with clustering; Right: Ground-Truth.

Fig. 3. Visualization of input and reconstruction images with respect to digits 0, 3, 5 and 9

cles, which clearly indicates that clustering stably converges using iterative training scheme.

	MNIST		*USPS*	
	NMI	*ACC*	*NMI*	*ACC*
Xie et. al [4], DEC	-	84.30%	-	-
Song et. al [3], AEC	66.90%	76.00%	65.10%	71.50%
Guo et. al [6], DCEC	-	85.29%	-	79.00%
Proposed	**84.97%**	**92.14%**	**79.89%**	**89.03%**

Table 2. Comparison of clustering quality with baselines on MNIST and USPS datasets

In addition, we carried out visual assessment where the t-SNE visualization method [17] was applied to evaluate clustering results of the proposed method. Fig. 2 shows a 2D projection of latent representation of our proposed method, where the clustering results are visualized with color coding using ground truth label. It shows that with jointed clustering loss, the learned latent representation space has more compact structures forming significant clusters which has better matching with true labels. Especially, with jointed clustering loss (Fig. 2, Right), the learned features has larger inter-cluster distances and tighter structures (see clusters labelled with green, magenta and dark blue colors) compared to the method using no clustering constraint (Fig. 2, Left).

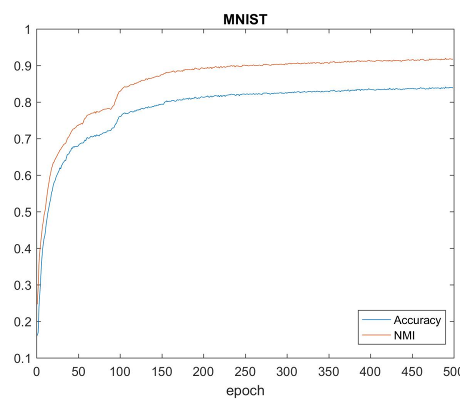

Fig. 4. Changing of accuracy and NMI during training on MNIST

4. CONCLUSION

In this paper, we introduce an unsupervised deep clustering method where a non-linear latent representation and compact clusters are learned jointly. The experimental results have demonstrated the effectiveness of our proposed method to cluster data into their appropriate group. Our potential future work is to experiment more difficult datasets and improve the accuracy of such deep clustering model.

5. REFERENCES

[1] Fei Tian, Bin Gao, Qing Cui, Enhong Chen, and Tie-Yan Liu, "Learning deep representations for graph clustering.," in *AAAI*, 2014, pp. 1293–1299.

[2] Peihao Huang, Yan Huang, Wei Wang, and Liang Wang, "Deep embedding network for clustering," in *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE, 2014, pp. 1532–1537.

[3] Chunfeng Song, Feng Liu, Yongzhen Huang, Liang Wang, and Tieniu Tan, "Auto-encoder based data clustering," in *Iberoamerican Congress on Pattern Recognition*. Springer, 2013, pp. 117–124.

[4] Junyuan Xie, Ross Girshick, and Ali Farhadi, "Unsupervised deep embedding for clustering analysis," in *International Conference on Machine Learning*, 2016, pp. 478–487.

[5] Fengfu Li, Hong Qiao, Bo Zhang, and Xuanyang Xi, "Discriminatively boosted image clustering with fully convolutional auto-encoders," *arXiv preprint arXiv:1703.07980*, 2017.

[6] Xifeng Guo, Xinwang Liu, En Zhu, and Jianping Yin, "Deep clustering with convolutional autoencoders," in *International Conference on Neural Information Processing*. Springer, 2017, pp. 373–382.

[7] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," *Artificial Neural Networks and Machine Learning–ICANN 2011*, pp. 52–59, 2011.

[8] Yueqing Wang, Zhige Xie, Kai Xu, Yong Dou, and Yuanwu Lei, "An efficient and effective convolutional auto-encoder extreme learning machine network for 3d feature learning," *Neurocomputing*, vol. 174, pp. 988–998, 2016.

[9] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep learning*, MIT press, 2016.

[10] Matthew D Zeiler, Graham W Taylor, and Rob Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2018–2025.

[11] Vladimir A Knyaz, Oleg Vygolov, Vladimir V Kniaz, Yury Vizilter, Vladimir Gorbatsevich, Thomas Luhmann, Niklas Conen, Wolfgang Forstner, Kourosh Khoshelham, Siddharth Mahendran, et al., "Deep learning of convolutional auto-encoder for image matching and 3d object reconstruction in the infrared range," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2155–2164.

[12] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2536–2544.

[13] Michael T McCann, Kyong Hwan Jin, and Michael Unser, "A review of convolutional neural networks for inverse problems in imaging," *arXiv preprint arXiv:1710.04011*, 2017.

[14] V. Turchenko and A. Luczak, "Creation of a deep convolutional auto-encoder in caffe," in *2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, Sept 2017, vol. 2, pp. 651–659.

[15] Andrea Vedaldi and Karel Lenc, "Matconvnet: Convolutional neural networks for matlab," in *Proceedings of the 23rd ACM International Conference on Multimedia*, New York, NY, USA, 2015, MM '15, pp. 689–692, ACM.

[16] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[17] Laurens van der Maaten and Geoffrey Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.