NEURAL CLUSTERING: CONCATENATING LAYERS FOR BETTER PROJECTIONS

Sean Saito & Robby T. Tan
Yale-NUS College
Singapore, 138533
{sean.saito, robby.tan}@u.yale-nus.edu.sg

ABSTRACT

Effective clustering can be achieved by mapping the input to an embedded space rather than clustering on the raw data itself. However, there is limited focus on unsupervised transformation methods that improve clustering and classification accuracies. In this paper, we introduce Neural Clustering[1], a simple yet effective unsupervised model to project data onto an embedded space where intermediate layers of a deep autoencoder are concatenated to generate high-dimensional representations. Optimization of the autoencoder via reconstruction error allows the layers in the network to learn semantic representations of different classes of data. We then use the k-NN algorithm to classify the projected points. Our experimental results yield significant improvements on other models and a robustness across different kinds of datasets.

1 INTRODUCTION

Clustering is a fundamental approach to unsupervised learning and is also used extensively for data visualization and analysis (Aggarwal and Reddy (2013)). In many cases, a given input domain is projected onto some embedded space for more accurate classification. It is an extensively studied field, with various methods for grouping data points. The k-NN algorithm, for example, takes the k nearest neighbors of a query node based on Euclidean distance to assign a classification. Many of these methods depend on the ”quality” of the embedded space; that is, the input domain must be transformed to a target space that provides greater separation of different classes and closer projection of similar data. Clustering on the pixels themselves is naive and hence an effective transformation is necessary. Hence the question arises: what kind of transformations achieve this? What kind of models can learn complex representations of the data? Unfortunately, research that address these questions are limited.

Recent work in this field involves deep embedded clustering (DEC) which uses autoencoders as non-linear mappings to the embedded domain (Xie et al. (2016)). This clustering method minimizes Kullback-Leibler (KL) divergence and uses the k-means algorithm to approximate k centroids generated by the encoded layer of the autoencoder, where k is the number of categories. However, this model requires the end-user to specify the number of centroids, k, and the target distribution $p(x)$ that is used to calculate KL divergence. This constitutes a naive approach, for it is possible for the optimal embedded distribution of a dataset to exceed k clusters. Moreover, the ideal target distribution is likely to differ depending on the data and hence is difficult to be determined heuristically.

In this paper, we introduce Neural Clustering, a simple unsupervised method that yields better classification results. Rather than using just the central encoded layer of an autoencoder to generate embeddings, we concatenate the learned representations of all intermediate layers. The training of our model only consists of one stage - the optimization of the autoencoder itself. We then classify the embedded points using the k-NN algorithm. The main idea is to use the features learned by each neural network layer to generate a combined representation which can be used for effective cluster analysis and classification.

[1] Implementation will be made publicly available at https://github.com/seansaito

2 NEURAL CLUSTERING

Our clustering model consists of two stages: the training stage, which involves optimizing an autoencoder, and the representation stage, whereby we extract the features learned by layers of the network to generate a descriptor.

2.1 AUTOENCODER

An autoencoder is a type of deep neural network which task is to find mappings for reconstructing the input domain. It is extensively studied in the unsupervised learning domain and has a variety of applications, such as denoising images (Vincent et al. (2010)). The architecture consists of two main components, the *encoder* and the *decoder*. The encoder learns a deterministic non-linear mapping that transforms the input to some lower dimensional representation. The decoder aims to find the inverse mapping. The autoencoder is trained with backpropagation to minimize the distance between the input and the decoder output:

$$E(x, \theta) = \frac{1}{n} \sum_{n} (x_n - g_\theta(f_\theta(x_n)))^2$$

where f_θ and g_θ are the encoder and decoder transformations with learnable parameters θ. Moreover, for 2-dimensional data such as images, we employ convolutional filters rather than vanilla deep neural networks. This is inspired by recent work that have significantly improved the benchmark for various image classification tasks using convolutional neural networks (LeCun et al. (1998); Krizhevsky et al. (2012); Girshick et al. (2014)). Using convolutional filters allows the autoencoder to identify local patches of spatial features. Certain visual features produce different outputs through a convolutional filter; this allows us to construct combined representations that act as discriminating descriptors.

2.2 COMBINED REPRESENTATION

Suppose an autoencoder with n layers of neurons between the input and the encoded layer. After the training phase of the autoencoder described above, we generate a descriptor for each input by concatenating the intermediate outputs of each layer in the network. Hence $d(x)$, the function for generating the descriptor, can be defined as:

$$d(x) = (l_1(x), l_2(x), \ldots, l_n(x))$$

where l_k is the transformation applied at encoder layer k:

$$l_k(x) = \sigma_k(W_k l_{k-1}(x) + b_k)$$

Using all intermediate layers rather than the encoding alone allows the embedded space to represent more complex semantic representations. For convolutional autoencoders, we exclude subsampling layers to avoid coarse representation of specific visual features. This idea is borrowed from work on fully convolutional networks that have produced promising results in semantic segmentation (Long et al. (2015)). Thus the intuitive idea behind the combined representation is to generate high-dimensional representations of the data that increases separates data of different classes and hence increases clustering accuracy.

As shown in Figure 1 in the Appendix, the Euclidean distance of the combined representations is lower on average for data of the same class (the diagonals). This helps a distance-based classification algorithm such as k-NN make better classifications. We also observe that certain digits that look alike, such as 4's and 9's, have relatively lower distances, while those that are clearly dissimilar have higher distances. Figure 2 shows examples of how layers in the network react to different classes of data.

3 Experiments

In the testing stage, we use the k-NN algorithm to evaluate the classification accuracy of our model. We conduct experiments on three datasets, namely MNIST, CIFAR-10, and 20newsgroups (14 classes). For MNIST and CIFAR-10, we use deep convolutional autoencoders, while for 20newsgroups we use vanilla deep neural networks and vectorize the input using the TF-IDF transformation. We compare our model performance with those of recent state-of-the-art techniques and other standard models. For the k-means algorithm, we set k as the number of classes.

3.1 Results

In Table 1, we report the best performance of each algorithm. Note our method outperforms all other models and produces state-of-the-art results. We observe significant differences for more complex datasets with higher dimensions; this indicates a robustness to varying levels of complexity. Table 2 compares variations to our model. Given the descriptors generated from the autoencoder, we apply t-SNE to reduce the dimension to either 2 or 3. This transformation yields faster inference, yet results show that classifying the raw combined representations produces the best results.

Table 1: Comparison of classification accuracies

Model	MNIST	CIFAR-10	20newsgroups
Deep Embedded Clustering	84.7%	18.6%	11.5%
t-SNE-2 + k-NN $(k = 3)$	92.4%	36.6%	36.6%
k-means $(k = \#\text{classes})$	53.5%	20.6%	19.4%
k-NN $(k = 3)$	95.8%	50.5%	37.2%
neural-clustering (ours)	**96.6%**	**61.1%**	**82.8%**

Table 2: Comparison of variations to our model

Model	MNIST
neural-clustering-tsne-2	92.4%
neural-clustering-tsne-3	95.4%
neural-clustering	**96.6%**

4 Conclusion and Discussion

This paper proposes Neural Clustering, an unsupervised method for generating embedded representations of data that enable effective distance-based classification. Our method does not depend on heuristics such as the number of desired centroids or the target distribution of the embedded space. Rather, optimization of an autoencoder allows each layer to learn a semantic representation of the data. Combining these representations can generate descriptors which can help distinguish certain categories from another.

Our results strongly support the effectiveness of this method. Not only does it produce state-of-the-art performances, it also demonstrates robustness across different types of datasets.

We also would like to raise certain questions regarding this model. There lacks firm theoretical grounding on why it outperforms others; there remains the question of how an autoencoder is able to learn features that help with clustering even if it does not directly optimize clustering error. Future endeavors would attempt to address these issues as well as observe its performance across different tasks and datasets.

References

C. C. Aggarwal and C. K. Reddy. *Data clustering: algorithms and applications*. Chapman and Hall/CRC, 2013.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

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. *Journal of Machine Learning Research*, 11(Dec):3371–3408, 2010.

J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. In *International Conference on Machine Learning (ICML)*, 2016.

5 Appendix

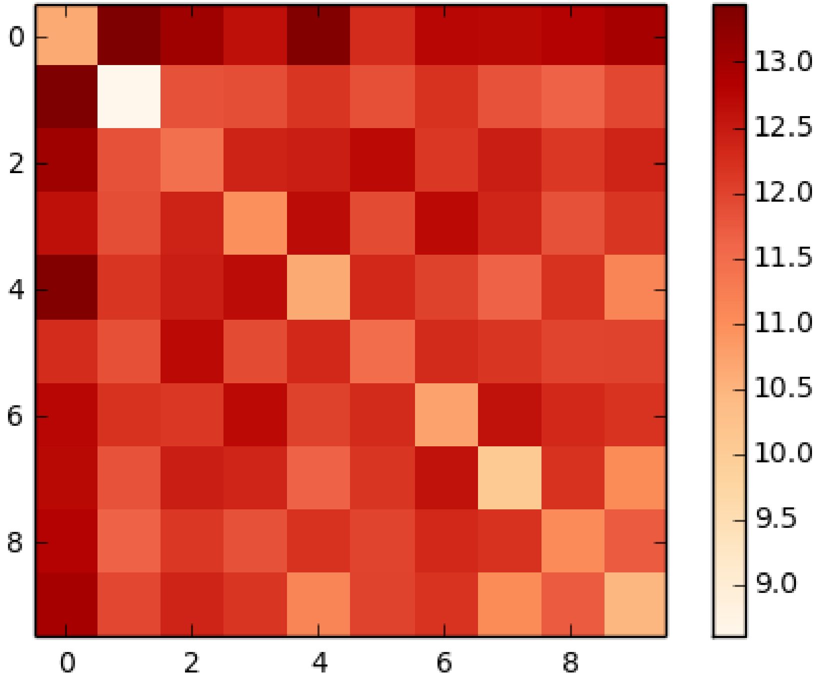

Figure 1: Heatmap representing the average Euclidean distances of the combined representations by class using MNIST data.

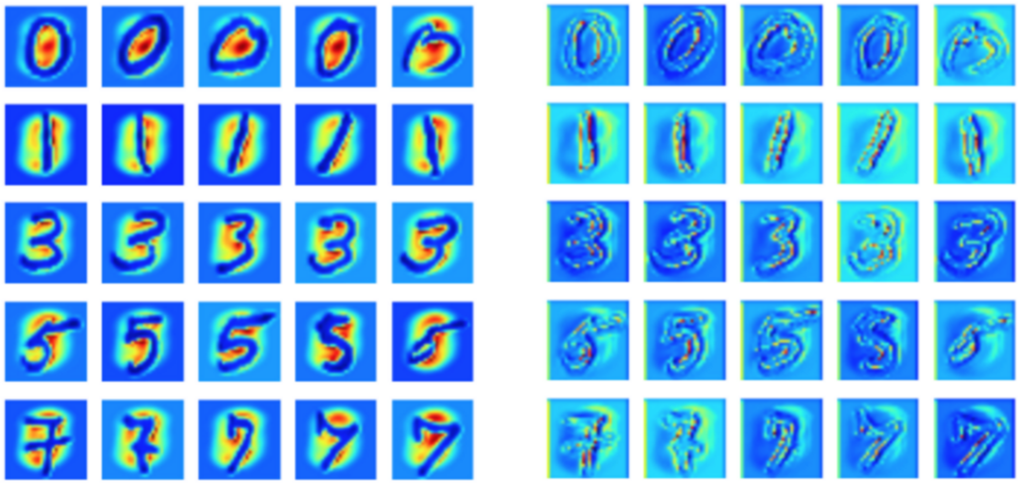

Figure 2: Example outputs of different filters from a particular layer in the autoencoder.

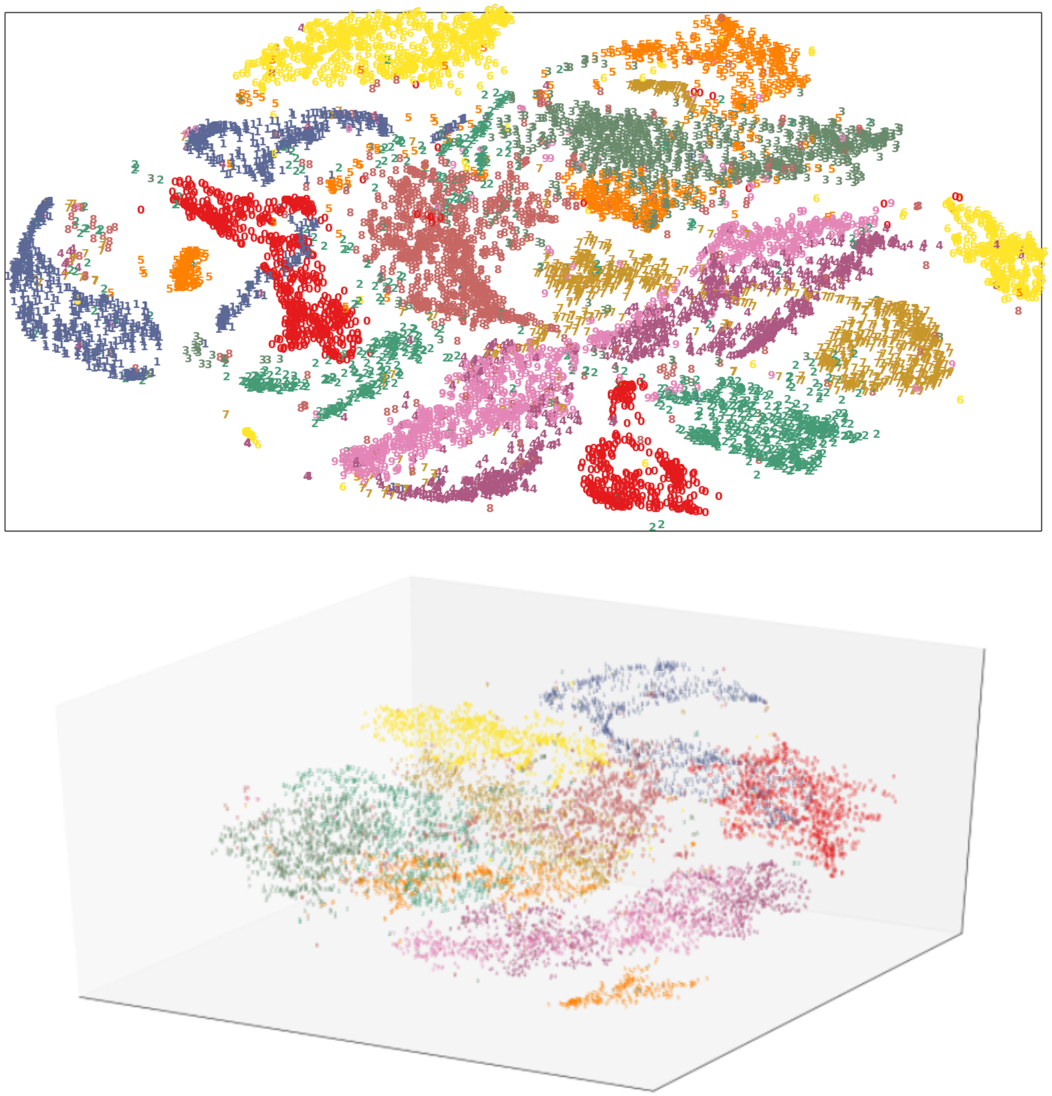

Figure 3: Projection of MNIST points to 2-D and 3-D spaces. These embeddings are generated by transforming the combined representations using the t-SNE algorithm. As discussed earlier, the embedding produces more than 10 clusters.