Clustering for neuron networks using the activation vectors of data

Zhenge Zhao

May 9, 2019

1 Introduction

Recent technical developments in the machine learning field have led to huge accuracy increase in a lot of practical applications and research areas such as image classification and pattern recognition. Among these techniques, neural networks are often considered to be the best-performing model and achieve success in many scenarios. However, unlike the previous methods like support vector machine(SVM) or decision tree. The multi-layer structure and the automatic differentiation process turn the neural network into a "black box". Their results are hard to interpret and the training process is hard to debug.

In addition to that, the nature of the complexity of neural networks also makes comparing them a non-trivial task. Given two neuron networks of different layouts, making direct comparisons is problematic because the difficulties of aligning neurons of different sizes and matching different dimensions. Moreover, even these two neuron networks are of the same structure but just different initializations or permutations of training batches, the comparison is still nontrivial because high dimensional parameters(e.g. weight and bias) in each layer.

In this project, we propose a new way of clustering neurons networks. Instead of directly comparing neuron networks of different similar structures, we are focusing on the activation vectors produced by neural networks and trying to find neuron networks which treat the dataset in a similar way. In another word, we would like to measure the similarities of two neuron networks by measuring the similarities of data clusters they produce individually for the same dataset under the same clustering method. We believe this could give us a method of clustering different neuron networks while bypassing the perplexing structures of them.

2 One sentence description

We are trying to cluster neuron networks having the same behaviors by using the activation vectors of the data.

3 Audience for this project

The outcomes of this project could be very interesting to the machine learning community and model designers. it could help them understand different behaviors of the neuron networks and analyze the models they are constructing. Without adopting such a method, the neuron network will remain to be a "black box" while comparing

and summarizing the common features of them will be difficult.

4 Approach Pipeline and technical details

The core of our approach is generating similarity measurements between two neural network models. Directly comparing structures of those complicated model parameters like weight and bias of each neuron is hard because of the alignment issue and non-linearities within the hidden layers. The method we propose, however, is suitable for all kinds of model structures as long as we are using the same dataset for the same classification tasks. My method is based on the assumption that the more similar two neuron networks are, they will also cluster the data in a similar way. Moreover, the activation vectors generated by neural network model for each data point automatically provide a dimension reduction mapping the whole dataset to a feature space which data points are still comparable.

To this end, our approach follows the pipeline below:

- Collect the activation vectors for each layer for all instances for each model.
- Generate clusters of the dataset for each model by using activation vectors.
- Utilize the cluster structure as representation of the model to calculate the similarity between two models.

4.1 Data collection

PyTorch [5] is a well-known library for training neural networks. We use it to generate 6 models for *Fashion-MNIST* dataset. *Fashion-MNIST* is a dataset of Zalando's article imagesconsisting of a training set of $60,000$ examples and a test set of $10,000$ examples. Each example is a 28×28 grayscale image, associated with a label from 10 classes. We choose a sample of 1000 including all of the 10 classes. The 6 models come from two different structures and they differs in *learning rate, momentum or different initializations*. All the 6 models reach an accuracy over 90% and is trained over 100 epochs.

We stored activations of all the 1000 data points for all the hidden layers of this 6 models across every epoch in the local disk for future use. We think the activations generated by hidden layers could collaborate with more information comparing to the single prediction. An the activation vectors of instances are comparable within any epoch of any model since the dimension are automatically matched.

4.2 Cluster Generation

For a specific model state(i.e. a epoch for any model), to generate clusters for the 1000 samples, we could choose a layer of activations therefore each vector have the same dimension m and could represent the corresponding data point. we use K-means [4] as our clustering approach. Before clustering these vectors, we also apply a dimension reduction method to the vectors since K-means may struggle for high dimensional data. We choose PCA [2] other than TSNE [8] since it is much faster.

After reducing the dimension of the data, we applied the K-means algorithm with 10 initial clusters. We choose K-means since the number of the clusters can be set as the same as the number of the classes and it has been implemented in the *Python* library *scikit-learn*. Later, for each

cluster, we choose the majority of the true labels within the cluster as the label for this whole cluster.

4.3 Model Comparison

We can choose a pair of models for further comparisons. For each model, we can represent each sample using the representative label of cluster it belongs. We can then compare two different clusterings by Rand index [1], which is regular measure of the similarity between two data clusterings in statistics. The Rand index has a value between 0 and 1, with 0 indicating that the two data clusterings do not agree on any pair of points and 1 indicating that the data clusterings are exactly the same. Such a method can represent a model state using the dataset while bypass aligning different dimensions of activation vectors directly.

5 Visual Analysis

For the models we generate, *NN0, NN1, NN4, NN5* has exactly the same structure but different initializations while *NN2* and *NN3* share the same structure. What's more, *NN1* and *NN3* have different learning rates and momentums. Having this similarity measurement, we start with looking at the model similarity at different epochs. More specifically, we generate similarity matrix for two convolution layers, two fully connected layers. As shown in Fig. 1, for all the 4 layers, from epoch to epoch, the *NN0*, *NN4* and *NN5* become more and more similar to each other. We can also see that *NN2* and *NN3* become more similar to each other but different from the other 4 neural networks in Fig. 1a and Fig. 1b.

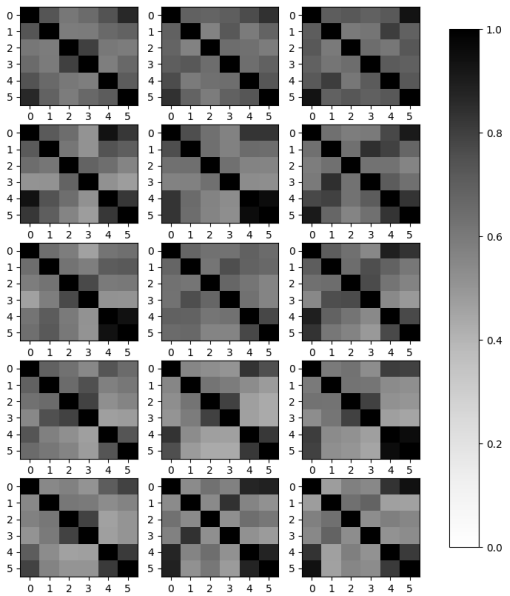
In addition to model comparison, we could could also use our similarity algorithm for comparing different layers. Shown in Fig. 2, we find out that two convolution layers tend to treat these samples in a similar way as well as two fully connected layers. We guess the reason is that convolution layers may capture different features than linear layers.

6 Conclusion

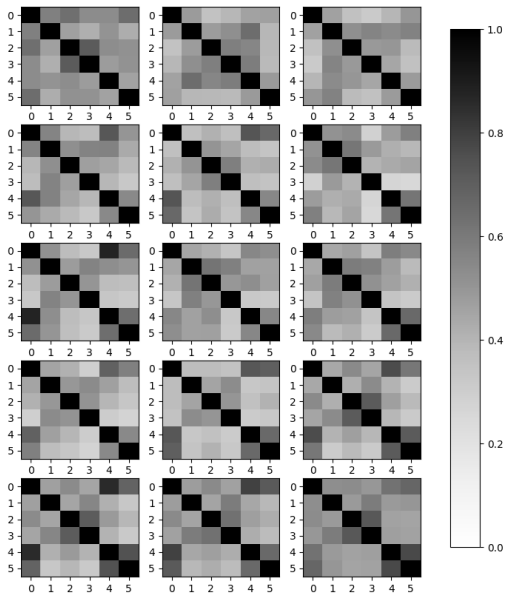
In this project, we generate models from Fashion-MNIST datasert, collect the activation vectors of all the instances for all the models and propose a new way of comparing different neural networks. we could expect our clustering method could cluster the models such that within one cluster two models recognize the instances more similarly than two models from two different clusters. And based on the distribution of the instances, we could summarize the behavior of the models in the same cluster. We could also see the application of the method when comparing different layers. The core of our method based on the fact that similar models or models should have more similar activation responses to the dataset.

7 Future Work

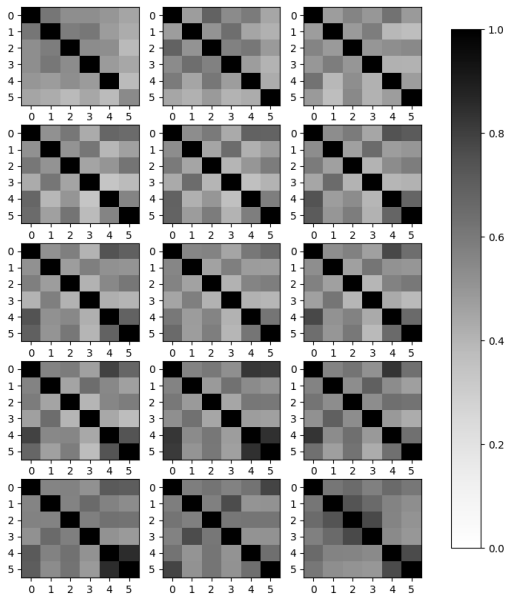
Currently, our work only look at a epoch each time. The next step could be how to combine these vectors when comparing a pair of models. This could give us a more comprehensive presentation of the data clusters. Moreover, we would like to apply our method to more samples and more models of more different structures to prove the scalability of our method. What's more, a better clustering method could be used other than K-means, For example, we could train a neural network to classify these activation vectors or using traditional method like multi-class



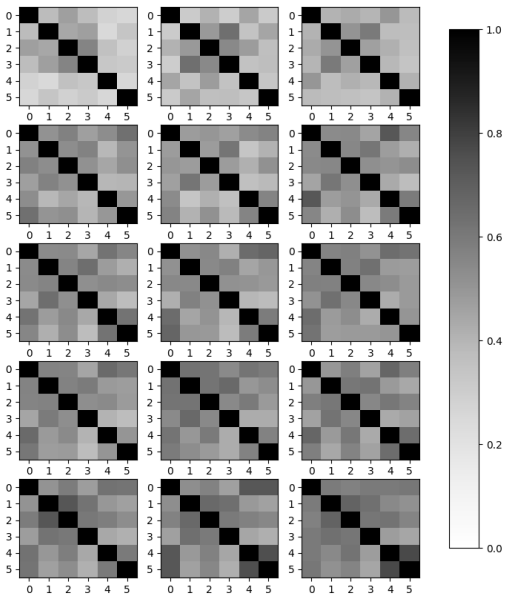
(a) First Convolution Layer



(b) First Convolution Layer

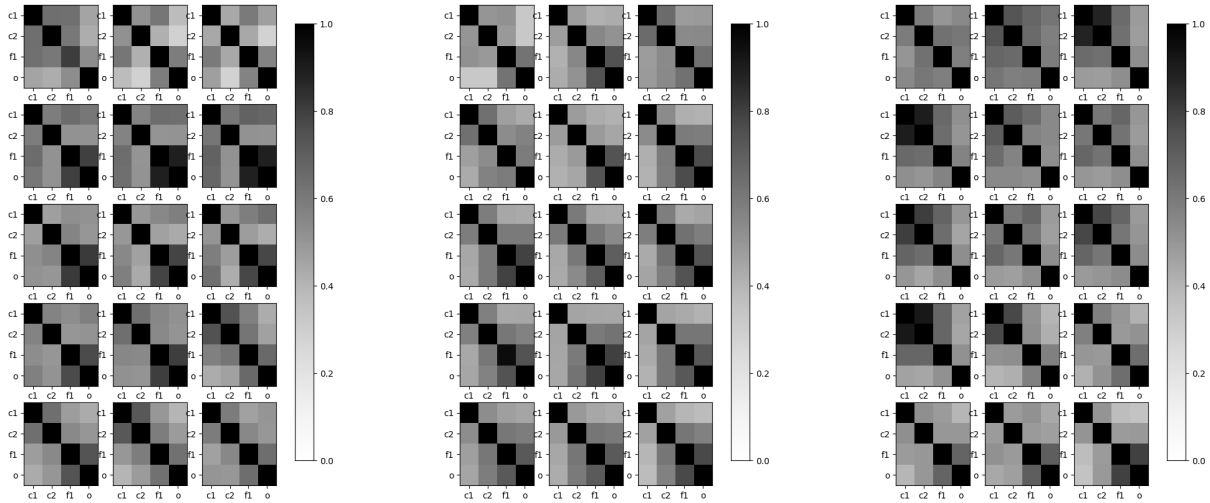


(c) Second Fully Connected Layer

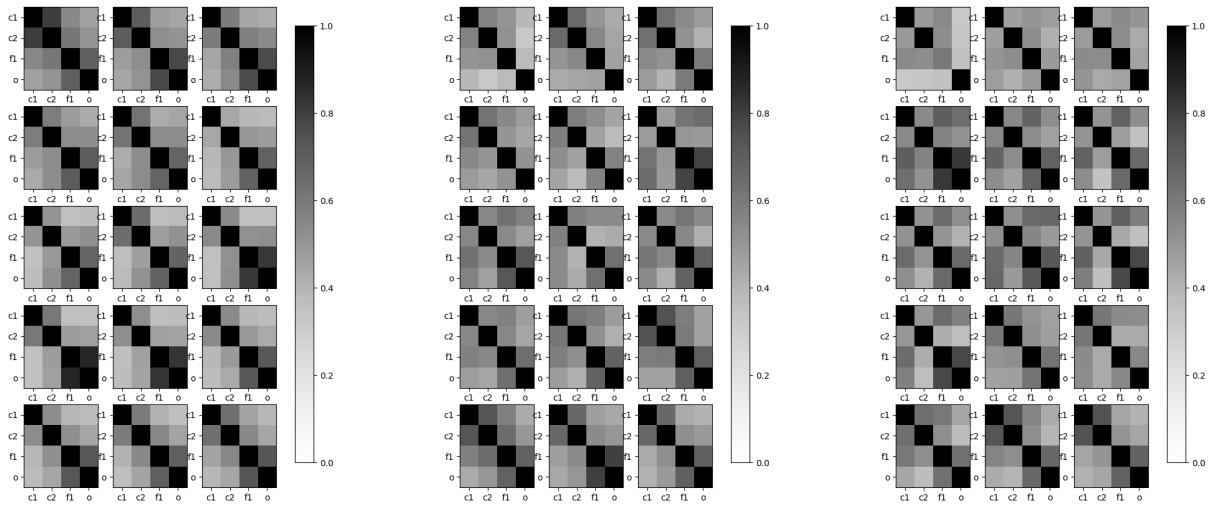


(d) Second Fully Connected Layer

Figure 1: The similarity matrices of 6 models for Epoch $1 - 10{,}20, 30, 40, 50, 100$



(a) NN0 (b) NN1 (c) NN2



(d) NN3 (e) NN4 (f) NN5

Figure 2: The similarity matrices of 4 layers for Epoch $1 - 10$,$20, 30, 40, 50, 100$

SVM. We could also consider other dimension reduction method like TSNE or UMAP for better results. Finally, other measures of similarity of data clusterings could be included since the measurement we choose will have a direct effect on the cluster of models we generate.

8 List 5 major publications that are most relevant to this project, and how they are related.

- This survey conducted by Kriegel et. al. [3] give a summary of several clustering methods for the high-dimension dataset.
- t-distributed stochastic neighbor embedding(TSNE) [8] proposed by Maaten et. al. is currently widely for dimension reduction.
- Principal component analysis(PCA) [2] is another widely used dimension reduction method.
- Kernel density estimation [6] is a non-parametric way to estimate the probability density function of a random variable. We can make use it to estimate the distribution of instances for each model.
- Recent work [7, 9] also propose different algorithms to do clustering for high-dimension and low sample size data.

References

[1] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, Dec 1985.

[2] I. T. Jolliffe. *Principal component analysis*. Springer, New York, 1986.

[3] Hans-Peter Kriegel, Peer Kröger, and Arthur Zimek. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Trans. Knowl. Discov. Data*, 3(1):1:1–1:58, March 2009.

[4] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press.

[5] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[6] Murray Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27, 09 1956.

[7] Yoshikazu Terada. Clustering for high dimension, low sample size data using distance vectors. 12 2013.

[8] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[9] Kazuyoshi Yata and Makoto Aoshima. Effective pca for high-dimension, low-sample-size data with noise reduction via geometric representations. *J. Multivariate Analysis*, 105:193–215, 02 2012.