Checking Functional Modularity in DNN By Biclustering Task-specific Hidden Neurons

Anonymous Author(s)
Affiliation
Address
email

1 Introduction

While real brain networks exhibit functional modularity, we investigate whether functional modularity also exists in Deep Neural Networks (DNN) trained through back-propagation. Under the hypothesis that DNN are also organized in task-specific modules, in this paper we seek to dissect a hidden layer into disjoint groups of task-specific hidden neurons with the help of relatively well-studied neuron attribution methods. By saying task-specific, we mean the hidden neurons in the same group are functionally related for predicting a set of similar data samples, i.e. samples with similar feature patterns.

We argue that such groups of neurons which we call Functional Modules can serve as the basic functional unit in DNN. We propose a preliminary method to identify Functional Modules via biclustering attribution scores of hidden neurons.

We find that first, unsurprisingly, the functional neurons are highly sparse, i.e., only a small subset of neurons are important for predicting a small subset of data samples and, while we do not use any label supervision, samples corresponding to the same group (bicluster) show surprisingly coherent feature patterns. We also show that these Functional Modules perform a critical role in discriminating data samples through ablation experiment. Also, these modules learn rich representations and are able to detect certain feature patterns, including visual classification as well as a biological application where these modules are shown to have learned different variations of DNA motifs.

We suggest that, for AI-DNN side, more advanced interpretation for DNN can be obtained by taking a module-based perspective; and for neuroscience side, the technical usage of neuron attribution and biclustering should also give some inspirations on new neuroscience experiments.

2 Related Works

Hypothesis of modularity in DNN is not new. Hinton et al. [4] put forward *neuron co-adaption* that some hidden neurons co-adapt together as a module for prediction. Co-adaption is only discussed in thought experiments supported by some biological inspirations and we do not know how to identify such co-adaptive neurons. However this phenomenon itself inspires Dropout [4; 8] which is arguably the most robust regularization technique for DNN.

Identifying modularity in DNN remains difficult as community detection methods are generally not directly applicable on densely-connected acyclic graphs. We turn to the relatively well-studied neuron attribution methods and biclustering algorithm for help.

Attribution methods for hidden neurons. Neuron attribution is easier to do for artificial neural networks than real neural nets. Basically, the goal of neuron attribution is to assign a score a to a hidden neuron n that represents how much important this neuron is for predicting a sample x to a class y. In a sense, the attribution score measures the per-sample importance of a hidden neuron.

There is a variety of neuron attribution methods. To name a few, Shrikumar et al. [6] proposed DeepLIFT, originally designed to assign scores to input nodes and can be generalized to assign a

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

score of importance to a specific hidden neuron. Sundararajan et al. [9] proposed integrated gradients for attributing input neurons and later generalized it to total conductance [7; 2] for attribution hidden neurons. Leino et al. [5] proposed an influence-based attribution method. Such methods have been demonstrated to be capable of identifying important hidden neurons that are relevant to a specific prediction on a class y given a data sample x. However, these methods do not consider that hidden neurons may be functionally related.

Biclustering. A data mining technique that simultaneously clusters the rows and columns of a matrix, and is especially popular in bioinformatics. There are many biclustering algorithms for different purposes, however, we choose to use spectral co-clustering because it produces biclusters with no overlaps, which is the simplest case for us to start with.

3 Method

At a high-level, our approach for finding Functional Modules in hidden neurons goes in two phases: 1) we first construct a neuron-sample matrix where each entry is a attribution score of a hidden neuron (row) for a input sample (column) and then 2) based on this matrix, we apply spectral co-clustering [3], a biclustering algorithm to simultaneously group neurons and samples that have consistent high attribution values.

Given a dataset $X = \{x\}^N$ of data samples, a pre-trained DNN f parameterized by θ such that $\hat{y} = f_\theta(x)$ is the prediction of sample x, and an attribution function a which can be any of [2; 5; 6; 7], the attribution score $a(n, x, y)$ measures how much a hidden neuron n contributes to the prediction of sample x into class y.

Construct the neuron-sample attribution matrix. We first specify the neurons to be from a given layer l and construct the neuron-sample matrix $M_{N_l * N_{\text{data}}}$ where $N = |l|$ is the number of neurons in that layer and $N_{\text{data}} = |X|$ be the size of dataset $X = x_1, x_2, \ldots, x_N$. Each entry in the matrix represents $e_{ij} = e(n_i, x_j)$, where e is a embedding function measuring the contribution of neuron n_i for predicting data sample x_j. We compute $e(n_i, x_j) = e_{ij} = a(n_i, x_j, \hat{y}) - \frac{1}{|C|}\sum_c a(n_i, x_j, c)$ where $\hat{y} = f_\theta(x_j)$ is the predicted class of x_j and a can be any of the attritbution function among [2; 5; 6; 7]. We choose to use DeepLIFT [6] score, and in experiments we find these attribution methods do not differ much.

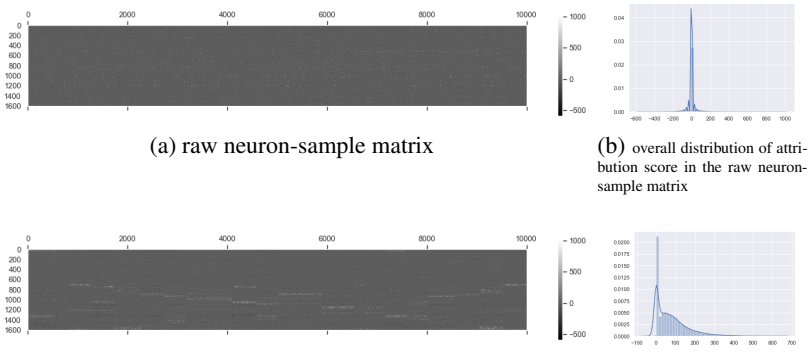
Spectral Co-clustering. Given the constructed matrix, we use spectral co-clustering to find biclusters with values higher than those in the corresponding other rows and columns. Given a predefined number k of biclusters, spectral coclustering algorithm treats the input data matrix as a bipartite graph and approximates the normalized cut of this graph to find heavy subgraphs. In the resulted biclusters, each row (neuron) and each column (sample) belongs to one bicluster with no overlaps.

4 Experiments

We first inspect a pre-trained DNN for visual classification on MNIST. The architecture is a 4 layer model where the first two layers are convolutional layer followed by two fully-connected layer. We use 10000 data samples and choose l to be the second convolutional layer of shape $(5, 5, 64)$. The total number of hidden neurons in l is 1600. We compute the DeepLIFT score [6] and construct the neuron-sample matrix and we set $k = 20$ in spectral co-clustering.

The neuron-sample matrix is highly sparse. The constructed neuron-sample matrix is very sparse, as only a few entries have attribution score above zero (see Figure 1a). We further report the distribution of attribution scores in Figures 1b and 1d, where the x-axis represents the attribute score and the y-axis represents the frequency. The overal distribution is centered around zero but inside one bicluster, the mass is concentrated on the positive side. Rearranging the display of the matrix by bicluster label, we can observe a checkerboard pattern in Figure 1c, indicating that certain groups of hidden neurons are indeed highly-correlated for predicting certain data samples.

Samples corresponding to the same bicluster show coherent feature patterns. We find that the samples in one bicluster, i.e., samples corresponding to the same Functional Module , show coherent and similar feature patterns. We pick a bicluster and randomly gather samples from that bicluster and find these samples show very similar patterns (Figures 2a and 2b) and belong to the same class. We also show the distribution of true labels in the total $k = 20$ biclusters (Figure 2c). Note that despite



(a) raw neuron-sample matrix

(b) overall distribution of attribution score in the raw neuron-sample matrix

(c) neuron-sample matrix rearranged with bicluster labels

(d) distribution of attribution score in the 11-th bicluster

Figure 1: *Visualization of the neuron-sample matrix and distribution of score.* (Better see in digital version) Each row corresponds to one of the 1600 neurons in the second convolutional layer ($5 * 5 * 64$). Each column represents one of 10000 data samples.

(a) samples from the 13-th bicluster (b) samples from the 14-th bicluster (c) Distribution of class labels in each bicluster (better see in digital version)

Figure 2: *Looking into biclusters*

not using any label supervision in our approach, we can achieve a very good job of unsupervised discriminating.

Ablation Study: Functional Modules are critical for discriminating samples. We check the performance of DNN by removing those hidden neurons by bicluster. In this experiment, we split the 10000 images X further into two sets where the first 5000 samples as validation set are processed by our approach and the rest 5000 samples are held-out for testing. We compare the accuracy when hidden neurons in layer l are gradually ablated on the test set with several baselines for ablating neurons, such as ablation of random neurons, ablation of top-important neurons greedily, and ablation of neurons by module (bicluster) (ours). Ablation of a neuron is implemented by setting the activation value of this neuron to zero, as convention in previous studies [2; 7]. As shown in Figure 3, ablation by module achieves the most significant accuracy drop compared with greedy ablation of top-k neurons (all use DeepLIFT score) and random ablation. It demonstrates that these Functional Module s found by spectral coclustering do play an important role in discriminating data samples.

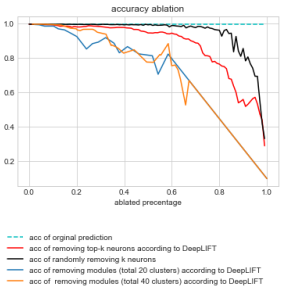
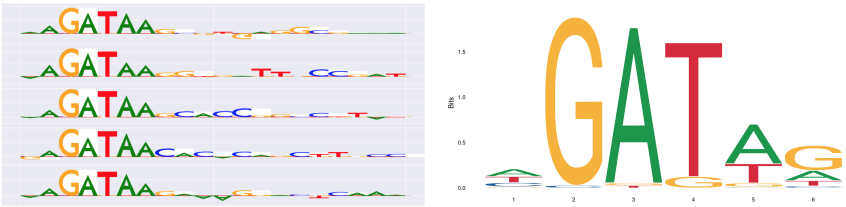


Figure 3: Accuracy decreases when neurons are ablated.

Inspecting what are learned by a DNN. We demonstrated how the Functional Modules can be used to discover what feature patterns are learned by a DNN trained for transcription factor binding. Recently there is a trend of applying convolutional neural networks for computational biology. DeepBind [1], for example, successfully applied deep learning to modeling the sequence specificity of protein binding with a performance superior to the best existing conventional learning methods. We inspect the DeepBind model to see if functional modules of neurons really learn to detect the specific motifs, even if DeepBind is trained through back-propogation without these prior knowl-

edge, and we compare the motifs to the ones that are manually discovered by scientists. We applied our approach on the DeepBind model and visualize the samples in one bicluster we find. We see that our finding is consistent with a motif that is known to be related in the binding process. This finding confirms that DeepBind model did learn something from the raw data.



(a) (b) known motif from Jaspar

Figure 4: *Different modules find different feature patterns* The same procedure is applied to a DeepBind model where the known motif for this transcription factor from Jaspar (a database of manually curated transcription factor binding profiles) is GATA. Each sample gene sequence is visualized using deep-lift score (note that here we use deepLIFT not only for constructing the neuron-sample matrix, but also to visualize the importance of each input base-pair).

5 Conclusion

We develop an approach to parcellate a hidden layer into functionally related groups which we call Functional Module , by applying spectral coclustering on the attribution scores of hidden neurons. We find the Functional Modules play an important role in discriminating data samples and we also show how Functional Module could be used to inspect what feature pattern are learned by the DNN.

References

[1] Babak Alipanahi, Andrew Delong, Matthew T Weirauch, and Brendan J Frey. 2015. Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nature Biotechnology* 33, 8 (jul 2015), 831–838.

[2] Kedar Dhamdhere, Mukund Sundararajan, and Qiqi Yan. 2018. How Important Is a Neuron? *arXiv.org* (May 2018). arXiv:1805.12233v1

[3] Inderjit S Dhillon. 2001. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 269–274.

[4] G E Hinton, N Srivastava, and A Krizhevsky arXiv. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arxiv.org* (2012).

[5] Klas Leino, Linyi Li, Shayak Sen, Anupam Datta, and Matt Fredrikson. 2018. Influence-Directed Explanations for Deep Convolutional Networks. *CoRR* (2018).

[6] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning Important Features Through Propagating Activation Differences. *arXiv.org* (April 2017). arXiv:1704.02685v1

[7] Avanti Shrikumar, Jocelin Su, and Anshul Kundaje. 2018. Computationally Efficient Measures of Internal Neuron Importance. *arXiv.org* (July 2018). arXiv:1807.09946v1

[8] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout - a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* (2014).

[9] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. *CoRR* abs/1703.01365 (2017). arXiv:1703.01365