Spatially Sparse Mixture of Experts

Daniel Byrne[1] and John Santerre[1,2]

[1] Master of Science in Data Science, Southern Methodist University, Dallas TX 75275 USA {byrned,santeerej}@smu.edu
[2] Springer Heidelberg, Tiergartenstr. 17, 69121 Heidelberg, Germany
lncs@springer.com
http://www.springer.com/gp/computer-science/lncs

Abstract. The difficulty in training and usig deep neural networks begins and ends with the fact that the design principle on which many are founded, fullly connected layers (FFL), contributes to a quadratic increase in the number of calculable parameters as the number of input dimensions and parameters increases. Traditionally, researchers have simply increased the number of parameters to increase the capacity of the neural network to learn representations. However, research into sparsely connected networks suggests that this need not be the case. Furthermore, research into small-world networks and biological systems suggests that the spatially local expert clusters can combine in situationally uniquie ways with distant expert clusters through sparse long distance connections to form conditionally active subnetworks adept at solving complex problems in a computationally efficient way. This paper introduces a new neural network architecture that compresses traditional representations using a t-SNE gated Mixture of Sparse Experts to enforce expert cluster specializations.

1 Introduction

Deep learning's explosive growth in learning over the past few years has been fueled by the technique's success at embedding high dimensional data into multi-level encoded representations of increasing abstraction to solve complex functions and or to model natural processes such as Natural Language Processing (NLP) and Computer Vision, and Speech Recognition. In common feed forward Deep Neural Networks, DNNs, layers are typically fully connected. That is every node at any one layer is connected to every other node in adjacent layers. This fully connected nature supplies each successive layer with any and all embeddings of its preceding layer in any possible combination. It is then the task of the training procedure to effectively reduce this over abundance of connections by iteratively applying the delta rule to adjust the weights on the connections. These learned weights then act as a gating threshold which activates or deactivates the neurons they are connected. The net result is a collection of weights and neurons that when applied an input stimulus effectively recreates an alternative representation or classification of the input data at the output of the network.

However, training such models is time consuming compute resource intensive. The difficulty in training deep neural networks begins and ends with the fact that the design principle on which many are founded, full connectivity, contributes to a quadratic increase in calculable parameters as the number of input dimensions and parameters increases. In a typical DNN, there may be millions of parameters and similarly large set of labelled training examples.

In a multi-class neural neural network, competing classifications with similar and dissimilar features will inject noise relative to each competing target classification. Neural networks then have to be trained to discriminate the important overlapping features while at the same time filtering out the less important noise to achieve adequate pattern seperation properties.

For example the theoretical ideal set of weights that consistently identifies cats in images is most likely in conflict with set of all possible weights of connections tuned to correctly classify trees. There will be some overlap, and that overlapping region is nonlinear objective function the neural network is trying to optimize. However, arriving at one of the many possible number of near optimal solutions to this equation is not trivial.

There have been a number of solutions to address this problem in the literature. The techniques vary, but the end results are that interfering patterns are filtered, and unique signals are amplified.

Dropout is a method by which a tunable percentave of random hidden units and their connections are eliminated from the network with each training case. During test all available neurons are included, but the trained weights on their connections are multiplied by the dropout probability p. Dropout gives major improvements in training time and accuracy by encouraging each individual neurons to learn a feature by degrading or filtering the impact other hidden units have on its connection weights.

DropConnect is similar to Dropout in that it introduces dynamic sparsity within the model, but instead of dropping neurons and their connections, it drops only connections based again on a settable probability parameter.

Embedding auto-encoders have been successful in image noise reduction and modeling the time series relationships of stock trading data. Auto encoders work by compressing input data of n dimensions into a smaller p dimensional hidden layer which is then re-encoded into an n dimensional output layer through gradient descent training. The effect of this compression and decompresssion technique is to filter out nonessential, noisey, data points.

In Convolutional Neural Networks, pooling layers are often utilized to reduce the size of the feature maps coming out of a filter bank before feeding the resultant filtered feature map to successive layers. This has the effect of reducing the number of interfering connections, reducing the computational complexity, and low pass filtering noisy features.

While Dropout and DropConnect reduce the number of parameters during training, but reinstate them during testing, a true reduction in the network connections has not been achieved. Song Han et al. chose to learn both the weights on connections and the existence of connections during training. Their

method first trains a dense network, prunes the low weight connections, and then retrains the network to fine tune the remaining connections.

Deep Compression has achieved compression rations of 98 percent with no loss of accuracy following a similar process, but in the final step they used Huffman encoding to further reduce the size of the network.

Inspired by research on Rat Brain Tissue which found that synaptic formation for specific functional brain regions can be modeled as a random formation, StochasticNets modified this above approach by starting with a sparse randomly connected network and ending with a sparse network. Pruning is thus inherent in their design.

Decebal Constantin Mocanu et al. again taking inspiration from the structure of biological networks argue that DNN should never have fully connected layers and subsequently demonstrate the effectiveness of sparse representations on several diverse neural network architectures (RBMs, MLPs, and CNNs).

N. Alex Cayco-Gajic in their study on the pattern seperation abilities of the cerebellum determined that feedforward networks such as those in the brain and in neural networks can separate patterns by projecting them onto a larger population of neurons and spatially encoding the inputs to decorrelate their influences.

Furthermore, research on a broad cross-section of network architectures has observed this pattern of densely connected clusters with sparse connections to remote clusters occuring organically in biological, social, and technological systems thus reinforcing the notion that this type of architecture is an optimal solution in a number of diverse paradigms.

Thus that leads to the assumption that while sparse networks tend to out perform dense ones, perhaps the networks we should be trying to develop are composite networks comprised of a mixture of local dense expert clusters weakly connected distant, non-overlapping, clusters.

Such a technique has been implemented by Hinton et al. in Sparsely Gated Mixture of Experts model. This models using a separate gating network which permits access to segregated subnetworks of trained to be experts on subdomains of the overall target classification domain. However their approach did not spatially segregate the training inputs leading to overuse of certain experts thus increasing their correlations and thus increasing the noise.

In this paper we propose a modification of this gated MOE model in which the gating neurons are spatially isolated from their neighbors using a Parametric T-SNE loss function. Parametric T-Stochastic Neighbor Embedding (T-SNE) is a method by which the Mutual Information component of similar Gaussian distributions are converted into a Euclidian distances to reduce a high dimensional signal into a lower dimensional space while preserving the structure of the data.

We first build a parametric t-sne model from our training data which during inference classifies inputs into a 2D vector representation. Training cases are thed fed through this Pt-SNE model to generate the spatially aware gate thresholds for each training case.

The gated MoE model is further modified to add a stochasitcally sparse number of connections between expert clusters. This information sharing between clusters reduces the greedy nature of training algorithm to overuse some clusters while minimally using others.

Gated subnetworks are sparsely initialized using methods devised in StochasticNet, and owing to the the potential influence of far flung experts on local problems, minimal connections in-between expert subnetworks are permitted albeit at a smaller randomly assigned rate.

Parametric T-SNE is used to gate the MoE FF networks so that spatial clusters can form and specialize on a represenatation that can then be used referentially by weakly connected distant clusters traied as experts in other domains. Furthermore allowing t-SNE to drive the gates allows us embed similarities in training data samples on top of the labels already attached to the data thus effectively applying additional classification labels to our data without any added effort.

The result is a spatially aware, sparsely connected and initiated, mixture of local experts with sparsely connected remote experts which allows for contributions from far flung expert networks through minimal influence connections.

References