Decorrelating Neural Network Activations

Jonathan Simon
Department of Computer Science
Cornell University
Ithaca, NY 14850
js3268@cornell.edu

Abstract

The strength of deep neural networks lies in their ability to represent a diverse array of complex features through their hidden unit activations. While some amount of redundancy between these feature representations is necessary for robust learning, too much redundancy can hinder a network's ability to encode information. We explore the role of dropout in reducing these redundancies between hidden unit activations, and propose a regularization technique for explicitly reducing these redundancies by penalizing strongly-correlated activations.

1 Introduction

1.1 Motivation

Over the past several years deep neural networks have become the dominant supervised learning technique across a wide range of fields [1]-[4]. The success of these algorithms is due to their ability to learn hierarchical feature representations, with each network layer encoding information at a different level of abstraction. However within each layer, it is not always possible to discern what specific features individual units are representing [5]. This problem is due in part to the fact that for a given feature, there may be several different units in the network which respond weakly to it, with no single unit representing that feature precisely. This phenomenon of partially-redundant feature representations also has the unwanted side-effect of decreasing the expressive power of the network. In the most extreme case where two hidden units in the same layer respond identically to all inputs, the behavior of the network could be exactly replicated by removing one of these units and rescaling the weights on the remaining unit. A similar intuition is used by Srivastava et al. to motivate the dropout heuristic, which is thought to work by breaking up undesired coadaptations between units. However rather than adding random noise to the network to eliminate these coadaptations as in dropout, we propose adding a term to the loss function which penalizes strong correlations between hidden unit activations.

1.2 Related Work

It has long been known that decorrelating neural network activations can aid in the learning process. One of the simplest examples of this is whitening the input data before it is fed into the network [9][10]. Whitening the activations in the hidden layers poses a greater challenge, as the network may tune its weights in a way which reverses the whitening, leading to a parameter explosion [11]-[13]. This instability problem has since been solved by the Batch Normalization algorithm, which z-scores the hidden layer activations rather than whitening them, and propogates the gradient through the z-score calculation [7]. An alternative approach to decorrelating activations is to introduce new regularization terms to the loss function which seek to minimize quantities related to correlation, such as covariance and cross-covariance [6][15]-[17]. Our proposed method distinguishes itself by discarding magnitude information in order to penalization correlations directly, as these are likely to be a better reflection of the feature information encoded by the units [18][19].

1.3 Network Architecture

In order to ensure a fair comparison between the different training methods used throughout the paper, we employ a single family of network architectures for all experiments. In all cases the network is fully-connected with 3 hidden layers where layer $n+1$ contains half as many units as layer n, and each layer uses a leaky ReLU nonlinearity. The 5 network sizes which we consider are shown in Table 1. All networks are trained to minimize a negative log-likelihood loss using stochastic gradient descent for 100 epochs with minibatch size $= 50$, learning rate $= 0.05$, weight decay $= 1\text{e}{-4}$, and momentum $= 0.8$. Because we are interested in the behavior of the layer activation statistics over the course of training, the base network ("VANILLANET") uses neither dropout nor batch normalization, as these would skew the statistics of interest. Two variants of this network are considered, one which employs dropout with drop probability $= 0.3$ at each layer ("DROPNET"), and one which employs our correlation regularization at each layer ("CORRNET"). This regularization is described in detail in Section 3. To reduce verbosity, specific network architectures are referred to throughout this paper using the syntax "[SIZE]-[TYPE]", e.g. "LARGE-DROPNET".

Table 1: Network Sizes

Size Name	Layer 1	Layer 2	Layer 3
TINY	10	20	40
SMALL	30	60	120
MEDIUM	50	100	200
LARGE	70	140	280
HUGE	90	180	360

2 Analysis of Dropout

There are two commonly-proposed explanations for the success of dropout:

1. By randomly dropping nodes from the network, the training process is implicitly ensembling over $2^{\#nodes}$ many possible network architectures
2. By introducing noise to the network, the hidden units are less likely to form strong co-adaptations with one another

The first explanation can be made rigorous by specifying how the ensemble of networks evolves over the course of training. However in its original formulation, the second explanation was described in the informal language of evolutionary biology [8]. In an effort to make this more concrete, we use hidden unit correlation as a proxy for the less well-defined concept of "co-adaptation", since two strongly co-adapted units are likely to exhibit strong correlations between their activations.

2.1 Expected Correlation

Let $\mathcal{B} = \{x_1 ... x_m\}$ be a minibatch of samples. Let $a_i^{(k)}$ be the activation of the i^{th} hidden unit on the k^{th} sample. Suppose that for each hidden unit a_i, the activations have been z-scored to have mean 0 and variance 1 over the minibatch, for example by using BatchNorm. Then for two specific hidden units a_i and a_j, the expected value of the unbiased estimator of the correlation between these units over the minibatch of samples is:

$$\mathrm{E}[\hat{\rho}_{ij}] = \mathrm{E}\left[\frac{1}{m-1}\sum_{k=1}^{m} a_i^{(k)} a_j^{(k)}\right] = \frac{1}{m-1}\sum_{k=1}^{m} \mathrm{E}[a_i^{(k)} a_j^{(k)}] = \frac{m}{m-1}\,\mathrm{E}[a_i^{(1)} a_j^{(1)}]$$

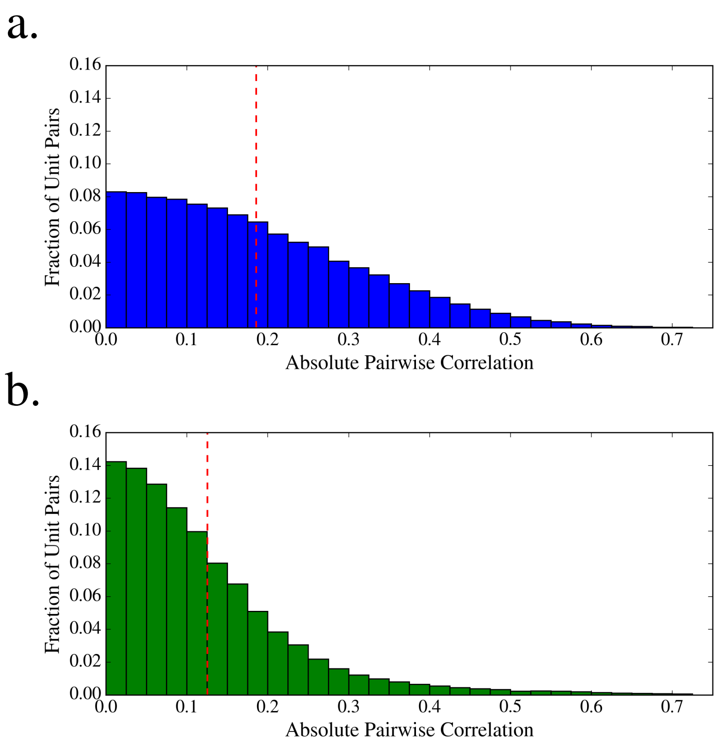

Figure 1: Absolute activation correlation distributions for Layer 1 of (a) HUGE-VANILLANET, and (b) HUGE-DROPNET. The red line indicates the mean correlation value.

Suppose for each sample we dropout a given unit with probability $1 - p$. Then whether or not the i^{th} unit is kept can be modeled as a Bernoulli random variable r_i having probability p. Therefore when using dropout the expected correlation becomes:

$$\mathrm{E}[\hat{\rho}'_{ij}] = \mathrm{E}\left[\frac{1}{m-1}\sum_{k=1}^{m}\left(r_i a_i^{(k)}\right)\left(r_j a_j^{(k)}\right)\right] = \frac{\mathrm{E}[r_i]\,\mathrm{E}[r_j]}{m-1}\sum_{k=1}^{m}\mathrm{E}[a_i^{(k)} a_j^{(k)}] = \frac{p^2 m}{m-1}\,\mathrm{E}[a_i^{(1)} a_j^{(1)}]$$

$$\therefore\ \mathrm{E}[\hat{\rho}'_{ij}] = p^2\,\mathrm{E}[\hat{\rho}_{ij}]$$

For the commonly-used value of $p = .5$, this corresponds to a 4-fold decrease in the expected correlation between the activations during training. And due to the network learning to model the statistics of the layer activations during training, it is reasonable to assume that this decrease in correlation will manifest during testing as well.

2.2 Observed Correlation

To test the validity of this derivation, we compare the distributions of pairwise correlations between hidden unit activations at test time between HUGE-VANILLANET and HUGE-DROPNET, as shown in Figure 1. The predicted trend is seen to hold for the correlations within the first layer of each network, with the mean correlations considerably lower in HUGE-DROPNET than in HUGE-VANILLANET. The activations in this first layer are approximately z-scored due to the preprocessing that is done on the network inputs, satisfying the assumptions of the derivation.

3 Correlation Regularization

3.1 Regularization Equation

In order to break up correlations between hidden unit activations, we define a layer-wise regularization term P which is equal to the normalized sum of squares of the pairwise correlations among all units in the layer computed across a minibatch. For a minibatch of size m and a layer of size d, the regularization term is computed as:

$$\mathrm{P} = \frac{1}{2d}\sum_{i=1}^{d}\sum_{j=1}^{d}\rho_{ij}^2 = \frac{1}{2d}\sum_{i=1}^{d}\sum_{j=1}^{d}\left(\frac{1}{(m-1)}\sum_{k=1}^{m} z_i^{(k)} z_j^{(k)}\right)^2$$

$$\begin{aligned} \text{where: } \rho_{ij} &= \text{correlation between units } i \text{ and } j \text{ over the minibatch} \\ z_\bullet^{(k)} &= \text{z-scored activation of the } \bullet^{\text{th}} \text{ unit on the } k^{\text{th}} \text{ sample in the minibatch} \end{aligned}$$

The effect of this regularization can be understood by considering its gradient with respect to a single hidden unit activation $a_l^{(k)}$:

$$\frac{\partial \mathrm{P}}{\partial a_l^{(k)}} = \frac{2}{d(m-1)\sigma_l}\sum_{i=1}^{d}\rho_{il}(z_i^{(k)} - z_l^{(k)}\rho_{il})$$

$$\text{where } \sigma_l = \text{standard deviation of the } l^{\text{th}} \text{ unit over the minibatch}$$

Note that although this expression contains a summand where $i = l$, this term is always equal to zero because the correlation of the l^{th} unit with itself is always constant and equal to 1. Expanding out the above expression and interpreting each of the terms separately:

$$\frac{\partial \mathrm{P}}{\partial a_l^{(k)}} = \frac{2}{d(m-1)\sigma_l}\left(\left(\sum_{i=1}^{d} z_i^{(k)}\rho_{il}\right) - \left(z_l^{(k)}\sum_{i=1}^{d}\rho_{il}^2\right)\right) = \mathrm{A}\cdot\left(\mathrm{B}-\mathrm{C}\right)$$

where: A is small when the l^{th} unit is highly variable and therefore less informative

B is large when activations in the k^{th} sample are unusually correlated with the l^{th} unit

C is large when the l^{th} unit is strongly correlated with many other units and its z-scored activation on the k^{th} sample is large

Therefore the gradient will be large and positive when the unit activations in the k^{th} sample are strongly correlated with the l^{th} unit, where the activations of the l^{th} unit do not vary much, and do not tend to be well-correlated with the activations of other units in the layer. This makes sense, because under these conditions the l^{th} unit is a reliable indicator for correlation due to its low variance, and the layer-wide unit activations in the k^{th} sample are all pairwise correlated due to their mutual correlation with the l^{th} unit.

As with any additive regularization term, the correlation regularizer can be assigned a multiplicative weight in the cost function. For each of the five network sizes, we trained two versions of CORRNET, one with a weight of 0.1 at each layer ("CORRNET-0.1") and one with a weight of 1.0 at each layer ("CORRNET-1.0"). The layer-wise correlations of these networks at test time are shown in Figure 2, where they are compared against VANILLANET. It is clear that as more weight is placed on the correlation regularization terms, the pairwise unit activations within each layer become less correlated.

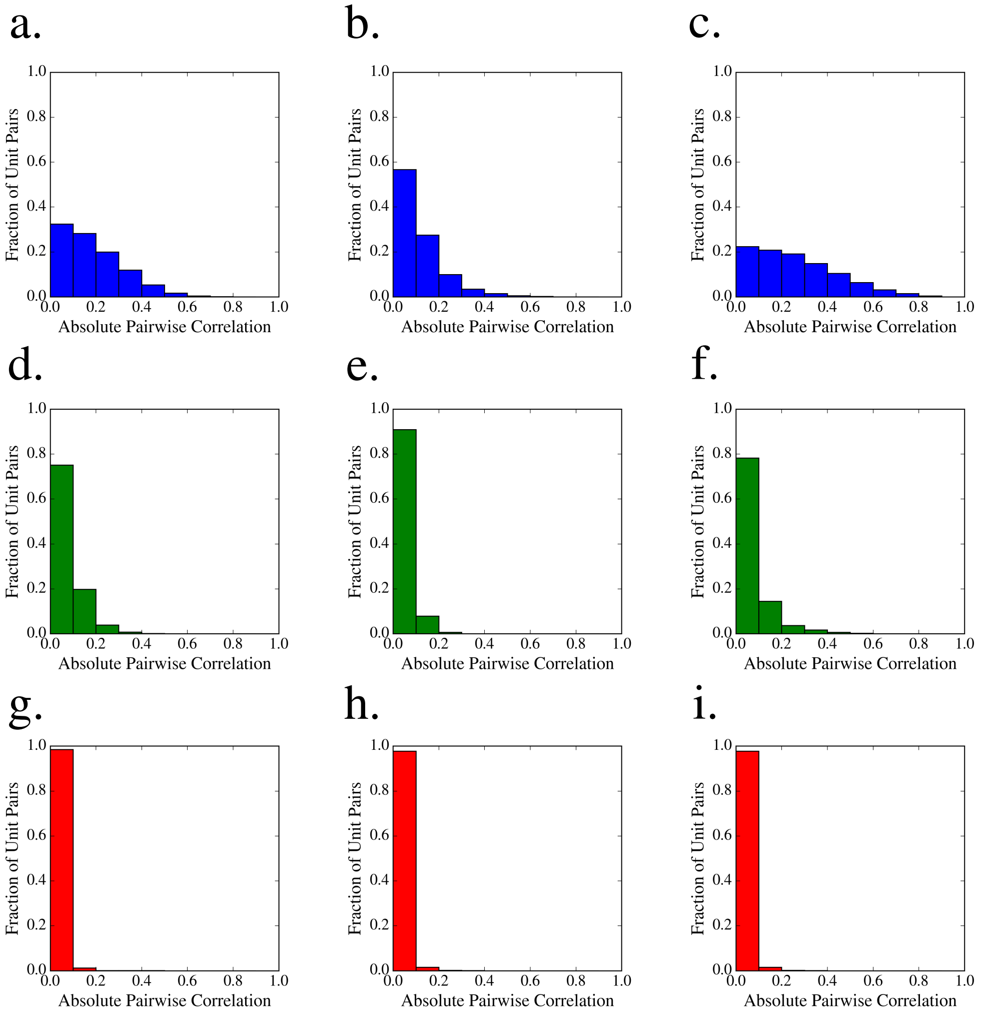

Figure 2: Comparison of layer-wise activation correlations between Huge-VanillaNet, Huge-CorrNet-0.1, and Huge-CorrNet-1.0. (a-c) Huge-VanillaNet correlations for hidden layers 1-3, (d-f) Huge-CorrNet-0.1 correlations for hidden layers 1-3, (g-i) Huge-CorrNet-1.0 correlations for hidden layers 1-3

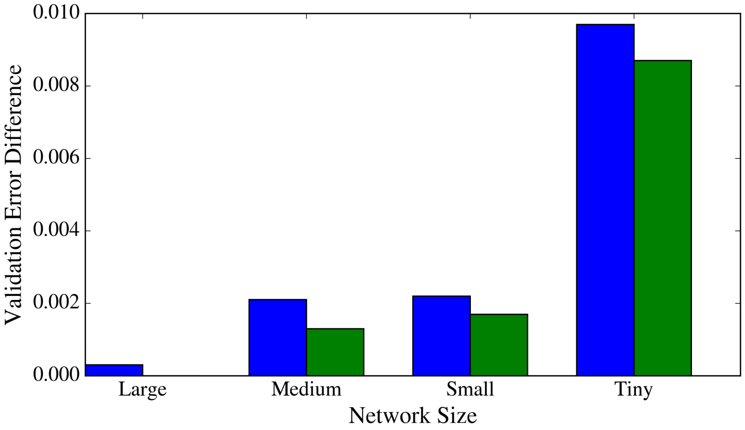

Figure 3: Difference in validation error compared to HUGE network of the same type. VANILLANET in blue, CORRNET-0.1 in green.

3.2 Validation Error

One of the hypothesized benefits of reducing correlations between hidden layer units was that it would reduce the amount of redundancy in the features encoded by different hidden units, and therefore allow for smaller networks to encode more task-relevant information. To test this, we measure how the validation error grows as the network size decreases. This is shown in Figure 3. As expected, the validation error grows more slowly for CORRNET-0.1 than it does for VANILLANET. However, this trend reverses itself when substantially larger weights are placed on the regularization terms. This is likely because the larger networks have enough units to learn the classification task while also satisfying these strict regularization constraints, whereas smaller networks are only able to meet these constraints by sacrificing classification accuracy.

3.3 Training Speed

We observe a consistent phenomenon where networks using correlation regularization experience a more rapid drop in validation error over the first few training epochs when compared with networks not using the regularization. Furthermore, the amount of weight placed on the regularizer is found to be directly correlated with the rate at which this drop occurs, as shown in Figure 4. This is somewhat surprising, as strongly-weighted regularizers typically slow down the rate at which learning occurs, since their purpose is to ensure that the network does not overfit. We believe that this observed change in learning rate is due to the fact that in the early stages of the training process the gradient estimates are extremely noisy, which makes it difficult for the network to update its parameters in way which reliably decreases validation error. Therefore, at this stage having the additional constraint that the network's units be mostly uncorrelated helps to guide the network parameters into a regime which is more conducive to learning. However as discussed in Section 3.2, placing too high a weight on the correlation regularizer adversely effects a network's ability to learn in the long-term.

4 Conclusion

The analysis of dropout in the simplified setting of z-scored activations proved to be informative insofar as the assumptions underpinning the analysis were met. In the future this analysis could be performed in the more general setting where the activations are not assumed to be z-scored. The correlation regularizer performed as expected, reducing the relative validation error in smaller networks. It also provided the unexpected benefit of decreasing validation error at the start of training. Some preliminary experiments have indicated that it may be possible to use a dynamic learning rate which shrinks over the course of training in order to take advantage of this initial drop in error

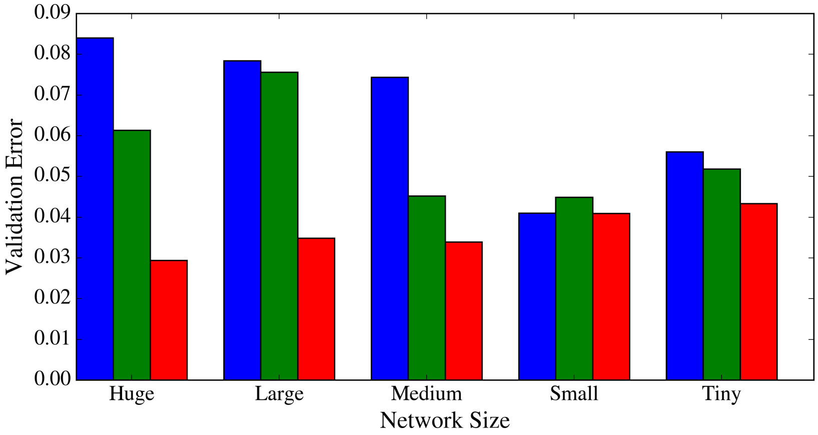

Figure 4: Validation error after 2 epochs of training. VanillaNet in blue, CorrNet-0.1 in green, CorrNet-1.0 in red.

without impeding the long-term training process. Another useful extension would be to extend the correlation regularizer so that it can be applied to convolutional layers in addition to fully-connected layers, as this would greatly expand the range of possible applications.

Acknowledgments

I would like to thank my MEng advisor Dr. Kilian Weinberger for providing help and guidance, and also thank Daniel Sedra for helping to writing the Torch code on which the neural networks used in this paper were based.

References

[1] A. Lusci, G. Pollastri, and P. Baldi. Deep Architectures and Deep Learning in Chemoinformatics: The Prediction of Aqueous Solubility for Drug-Like Molecules. *Journal of Chemical Information and Modeling*, 53(7): 1563-1575, 2013.

[2] M. Spencer, J. Eickholt, and J. Cheng. A Deep Learning Network Approach to ab Initio Protein Secondary Structure Prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics/IEEE, ACM 12.1*, 103-112, 2015.

[3] G. Hinton, L. Deng, D. Yu, et al. Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, 29(6): 82-97, 2012.

[4] R. Socher, B. Huval, B. Bhat, C.D. Manning, A.Y. Ng. Convolutional-Recursive Deep Learning for 3D Object Classification. *Advances in Neural Information Processing Systems*, 656-664, 2012.

[5] J. Yosinski. J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding Neural Networks Through Deep Visualization. *Deep Learning Workshop, International Conference on Machine Learning*, 2015.

[6] M. Cogswell, F. Ahmed, R. Girshick, L. Zitnick, and D. Batra. Reducing Overfitting in Deep Networks by Decorrelating Representations. *International Conference on Learning Representations*, 2016.

[7] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Proceedings of The 32nd International Conference on Machine Learning*, 448-456, 2015.

[8] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15: 1929-1958, 2014.

[9] Y. LeCun, L. Bottou, G. Orr, and K. Muller. Efficient backprop. *Neural Networks: Tricks of the trade*, Springer, 1998.

[10] S. Wiesler and H. Ney. A convergence analysis of log-linear training. *Advances in Neural Information Processing Systems 24*, 657-665, 2011.

[11] G. Desjardins, K. Simonyan, R. Pascanu, and K. Kavukcuoglu. Natural Neural Networks. *Advances in Neural Information Processing Systems 28*, 2071-2079, 2015.

[12] T. Raiko, H. Valpola, and Y. LeCun. Deep Learning Made Easier by Linear Transformations in Perceptrons. *International Conference on Artificial Intelligence and Statistics*, 924-932, 2012.

[13] D. Povey, X. Zhang, and S. Khudanpur. Parallel Training of Deep Neural Networks with ?Natural Gradient and Parameter Averaging. *CoRR*, abs/1410.7455, 2014.

[14] B. Rosen. Ensemble Learning Using Decorrelated Neural Networks. *Connection Science*, 8(3): 373-384, 1996.

[15] B. Cheung, J.A. Livezey, A.K. Bansal, and B.A. Olshausen. Discovering hidden factors of variation in deep networks. *Proceedings of the International Conference on Learning Representations*, abs/1412.6583, 2014.

[16] J. Bergstra and Y. Bengio. Slow, Decorrelated Features for Pretraining Complex Cell-like Networks. *Advances in Neural Information Processing Systems 22*, 99-107, 2009.

[17] K. P. Kording, C. Kayser, W. Einhauser, and P. Konig. How Are Complex Cell Properties Adapted to the Statistics of Natural Stimuli? *Journal of Neurophysiology*, 91: 206-212, 2004.

[18] B.A. Olshausen and D.J. Field. Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1? *Vision Research*, 37(23): 3311-3325, 1997.

[19] G.E. Hinton. Distributed Representations. Technical Report, 1984.