The energy landscape of a simple neural network

Anthony Gamst acgamst@ccrwest.org
Center for Communications Research, La Jolla, CA 92121
Alden Walker akwalke@ccrwest.org
Center for Communications Research, La Jolla, CA 92121

Abstract

We explore the energy landscape of a simple neural network and demonstrate that the space of trained networks is strictly smaller than the space of all networks with the same architecture. In particular, it is possible to design a simple neural network to produce a function with both low and high-frequency oscillations, while networks with an identical architecture, trained on the output of that function, can only learn the low-frequency features. This is a kind of implicit regularization. Depending on perspective, this can be beneficial.

1 Introduction

Successfully trained neural networks often have many more parameters than seems appropriate for the size of the training data. Given the potential complexity of these networks, why do they not immediately overfit?

We gave partial answers to this question in [Chen et al.(2016a)] and [Chen et al.(2016b)]. Namely, randomly initialized networks are much less complex than one might expect from a naive parameter count, and trained networks retain this simplicity. In this paper, we expand upon the second point. Specifically, we will describe an example function and neural network architecture such that the network is capable of perfectly fitting the function but never does under training. We then study the energy landscape of this network: we find that global minima (perfect fits) are hidden in narrow valleys, while local minima, which are common, occur in broad, flat plains, and correspond to a version of the function which has had the high-frequency components removed. That is, the energy landscape has a regularizing effect.

The paper [Ballard et al.(2017)], discovered after the initial draft of this paper was written, also explores the energy landscape of neural networks. We make one clarifying remark: in the conclusion of [Ballard et al.(2017)], they mention that it is easy to find a local minimum competitive with the global minimum *when fitting a network to an already trained network of the same architecture*. Our previous work shows that the previously trained network is likely to output a simple, low complexity function, so training a new network to match its output should be easy. Our observation in this paper is that it is easy to construct (not train) a network whose output cannot be matched by training a network with the same architecture.

2 Construction of a perfect fit

For simplicity, all our networks will have one input, one output, and K hidden densely connected layers of N nodes each. We call this a $K \times N$ network. The hidden layers all have ReLU activation, and the output layer has a linear activation. We now hand-construct a particular 5×5 network W_f whose output we call the function f. Using the left 5×2 nodes, we construct a high-frequency sawtooth, and using the right 5×3 nodes, we construct a low-frequency spline. The top node computes the sum, as shown in Figure 1.

Note, though, that when we train a 5×5 model to the output of W_f, we do not enforce the two sided structure — the network is fully connected on each layer. The $2 + 3$ format is just used in the construction.

3 Experimental results

3.1 Method

For the remainder of the paper, we describe various features of the energy landscape of the weight space of 5×5 networks using mean squared error loss against the function f output by W_f. Our training data

OPTML 2017: 10th NIPS Workshop on *Optimization for Machine Learning* (NIPS 2017).

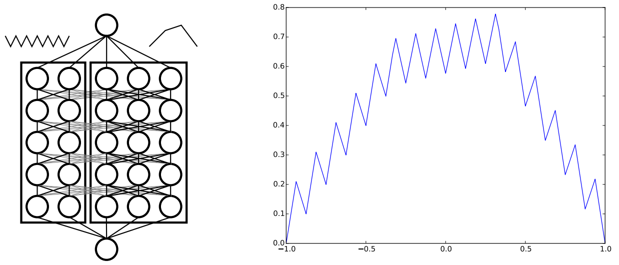


Figure 1: Our main example model W_f, constructed as a sum of a high-frequency sawtooth wave and a low frequency spline (left) and the graph of the output function f of this network (right). Our hand-constructed model has nonzero weights only on the bold lines, but the models trained on its output have no constraints on the weights.

consisted of sampling the function f, a linear spline, at all of its knots and at 9 points between each pair of knots. We find that providing more data does not affect our results. Every training step provided the entire dataset to the optimizer and used either gradient descent with momtentum or Adadelta. We trained all models for 50K steps; this is far more than necessary for the training to stabilize, but we want to be certain that our trained models are close to true local minima. For our main experiment, we trained 320K models, half with Adadelta and half with gradient descent.

3.2 Average fitted models

Figure 2 (red) shows the average output of the trained networks and a standard deviation tube for both optimization methods. Note that neither method can find the high frequency component.

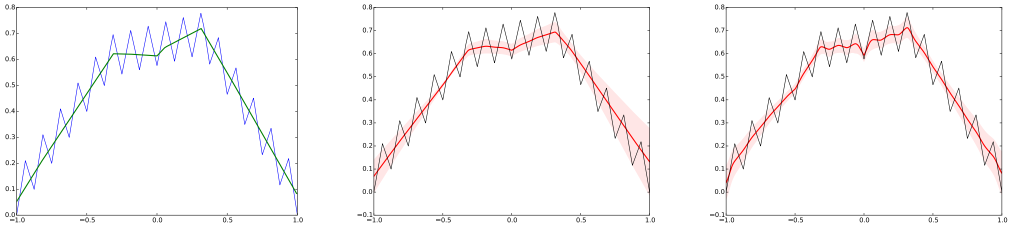


Figure 2: Fitting a 5×5 network to f (blue) always produces a network with output similar to the green plot, left. The other two plots show in red the average fit and standard deviation bands for 160K trials of gradient descent (middle) and Adadelta (right).

Figure 3 shows the smallest MSE fits over 160,000 trials each. Clearly, Adadelta does better, but it can't quite find an optimum.

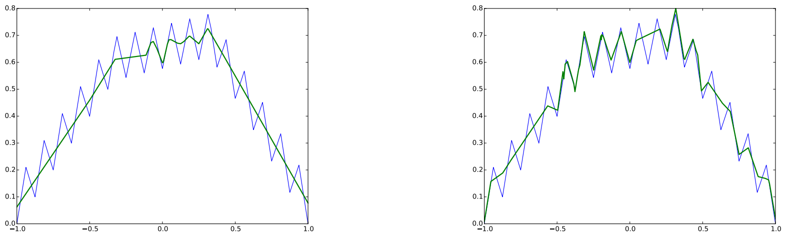


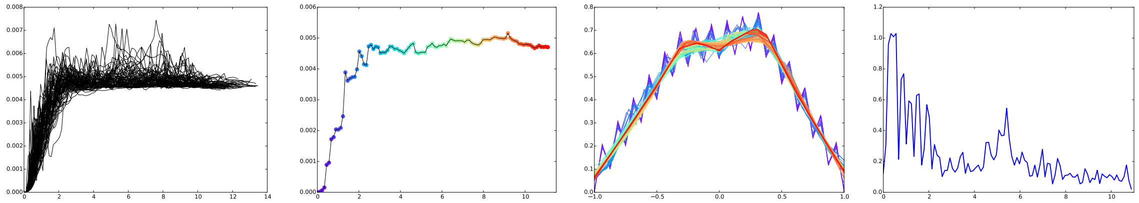
Figure 3: The best fits for gradient descent and Adadelta over 160,000 trials.

3.3 NEB paths

It is difficult to gain a sense of the topography of the energy landscape in the 136 dimensional weight space. We found it useful to use the nudged elastic band (NEB) method (see [Jónsson et al.(1998)]), which produces locally minimal paths between points in weight space; plotting the energy along these paths shows that the energy landscape is flat and slightly bumpy until a dramatic dropoff near the global minimum.

There is a technicality, however: dense neural networks have a large number of symmetries; for example, we can permute the indices of the weights or change the scale of the weights in one layer and reverse it in the next. This implies that the symmetry group contains a copy of $\mathbb{R}_+^{KN} \rtimes S_N^K$. The result is that there are large submanifolds within weight space which have identical function output, so the global minimum is not unique. To address this issue, we performed the following de-symmetrization procedure on W_f and all trained networks: first, we found the invariant weight scaling minimizing the L_2 norm of the entire weight vector. Then we permuted the nodes in each layer so the columns of the weight matrices were in increasing L_2 norm order. We found this smoothed the NEB paths somewhat but does not affect any conclusions we draw; we mention it here for completeness. There is a question of whether removing symmetries from the weight space will make training easier. See [Badrinarayanan et al.(2015)], for example, for more background. We emphasize, however, that our symmetrization does *not* affect the training of the networks here. Symmetrization is performed only after training to select a single representative from the set of all weights which produce the same output as the fitted network.

Figures 4a and 4b show NEB paths in weight space between the globally optimal W_f and some locally minimal trained networks.



(a) (b) (c) (d)

Figure 4: One hundred NEB paths (4a) from the global minimum to local minima, and a single NEB path (4b) color coded to correspond to the path in function space shown in (4c). Figure 4d shows the angle (radians) between successive difference vectors along the NEB path. Plots a, b, and d have the same horizontal axis: distance along NEB paths.

In Figures 4a and 4b, it is apparent that the basin of attraction of the global minimum has radius approximately 2. In the one-dimensional NEB plot, it appears as though a random walk from a local minimum might stumble upon this basin of attraction. However, in 136 dimensional weight space, the ball of radius 2 around the global minimum is exponentially miniscule compared to the shell of radius 9.5 to 10.5 which contains most of the local minima, so the chance of randomly finding the valley around the global minimum is essentially 0.

Figures 4b and 4c show that across the large, flat region of the path, the output changes very little as the function rearranges itself while preparing to create many spikes, which it does at the very end as it plunges towards the global minimum. Figure 4d shows how the NEB path requires dramatic twisting as it approaches the global minimum.

3.4 Basin of the global minimum

The obvious initial rise of the NEB paths gives us an indication of the size of the basin of attraction of the global minimum, but we can measure it another way: if we jitter the weights of the global minimum by a small amount and then retrain, we see how much noise the model can take before it cannot find the global minimum again. Figure 5 shows the results of doing this experiment 4096 times with gradient descent and Adadelta.

For each experiment, we added a small amount of Gaussian noise (with L_2 norm chosen approximately uniformly in $[0, 2]$) and trained for 20K batches, just as with our main experiment above. We then compute the loss of the final model and plot it against the amount of noise added. Note that the dynamic learning rate in Adadelta appears to not allow the step size to vanish, so it hovers around but does not stop precisely on a local minimum.

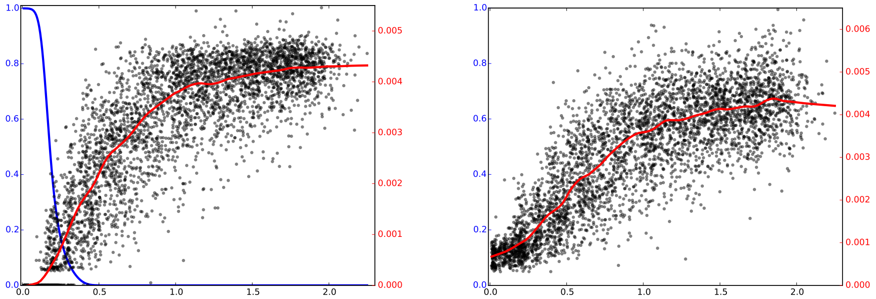


Figure 5: For both gradient descent and Adadelta, a scatter plot of the final loss plotted as a function of the L_2 norm of the added noise (initial distance), and a lowess approximation (red). Also plotted is a logistic fit to the probability (blue) of returning to a global minimum as a function of the added noise amount. Note the two different vertical scales, and note that the probability curve for Adadelta is not visible because it is identically zero.

3.5 Other experiments

We performed other experiments exploring the output of gradient descent and Adadelta, the organization of the local minima in weight space, and the eigenvalues of the energy Hessian at all minima in weight space. The interested reader should consult our full paper [Gamst and Walker(2017)].

We make one final remark: we are not claiming that it is difficult for *any* network to fit the given function f. Indeed, fitting a 10×10 network to f with Adadelta quickly finds a fit very close to the global minimum. This is because f is a far less complex function from the perspective of a 10×10 network. The point is that the output of a moderately complex 5×5 network is difficult to replicate by training another 5×5 network. In fact, our chosen network W_f is actually much simpler than it could be. It is easy to construct 5×5 networks with far higher frequency components.

4 Conclusion

We have shown that the weight space of neural networks can be a hilly place, and there can be global minima which are essentially unreachable using typical training techniques. The wide, flat local minima make optimization very difficult. In one sense, this is just the observation that nonconvex optimization is hard. But this difficulty grants neural networks an interesting opportunity: because complex, high-frequency functions are difficult to fit, there is an implicit noise-dampening regularization built into the network, and training time and architecture become parameters which control how complex the output of a network can be. So, in regression examples at least, networks of moderate size will smooth rather than reproduce the training data.

In a perfect world, we might hope for a global-optimum oracle. In this case, we could obtain the same sort of regularization by limiting the effective size of our models. This is the standard approach in non-parametric regression, for example. Neural networks, on the other hand, with real-world optimization techniques appear to have the rather fortuitous property of being somewhat self-regularizing. How much we can rely on this fact remains a model selection problem.

References

[Badrinarayanan et al.(2015)] V. Badrinarayanan, B. Mishra, and R. Cipolla. Symmetry-invariant optimization in deep networks. arxiv:1511.01754, 2015.

[Ballard et al.(2017)] A. J. Ballard, R. Das, S. Martiniani, D. Mehta, L. Sagun, J. D. Stevenson, and D. J. Wales. Perspective: energy landscapes for machine learning. arxiv:1703.07915, 2017.

[Chen et al.(2016a)] K. K. Chen, A. Gamst, and A. Walker. The empirical size of trained neural networks. arxiv:1611.09444, 2016a.

[Chen et al.(2016b)] K. K. Chen, A. Gamst, and A. Walker. Knots in random neural networks. In *NIPS Workshop on Bayesian Deep Learning*, 2016b.

[Gamst and Walker(2017)] A. Gamst and A. Walker. The energy landscape of a simple neural network. arxiv:1706.07101, 2017.

[Jónsson et al.(1998)] H. Jónsson, G. Mills, and K. W. Jacobsen. Nudged elastic band method for finding minimum energy paths of transitions. In B. J. Berne, G. Ciccoti, and D. F. Coker, editors, *Classical and Quantum Dynamics in Condensed Phase Simulations*, chapter 16. World Scientific, 1998.