Experimental Analysis of Overparameterized Neural Networks

EECS E6699: Mathematics of Deep Learning

Brendan Kondracki (bk2793)
Shambhavi Roy (sr3767)
Deepak Dwarakanath (dd2676)

Overview

- In recent years, a great deal of research has been done analyzing the behavior of overparameterized neural networks.
    - Du 2019 showed that overparameterized networks can achieve linear convergence to a global minimum (while trained in the lazy regime).
    - Chizat 2020 showed that neural networks can be linearized through certain scalings/weight initializations.
- In this paper, we aim to analyze the following:
    - The behavior of increasing width, shallow networks when trained with more advanced optimization techniques
    - The convergence and generalization of shallow networks when trained in various lazy/active regimes.

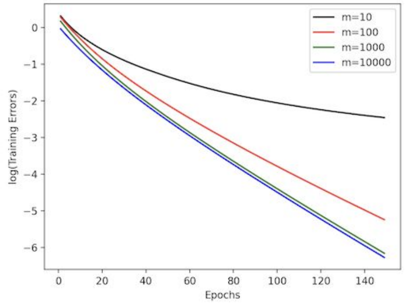
Adagrad Optimization

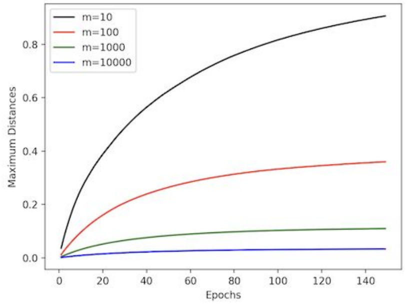
- Similar to the experiments conducted in Du 2019, we analyzed the behavior of an increasing width, shallow ReLU network using standard gradient descent. In addition, we also analyzed the behavior of the network using Adagrad optimization.
- We found that in both cases, as the network width increased, both the training errors over time and the change in model parameters, decreased. However, in the case of Adagrad optimization, we saw faster convergence for large width networks, and a slower convergence for small width networks, compared to that of SGD.

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \cdot \frac{\partial L(W(t), a)}{\partial w(t)}$$

SGD/Adagrad Comparison

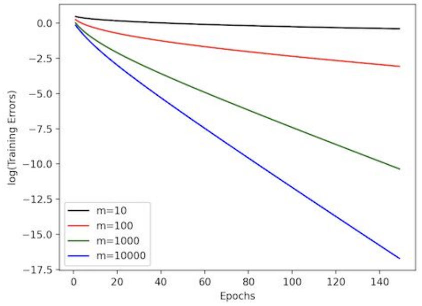
SGD:

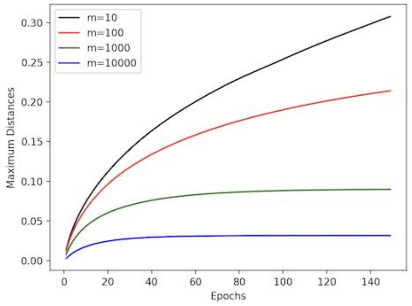


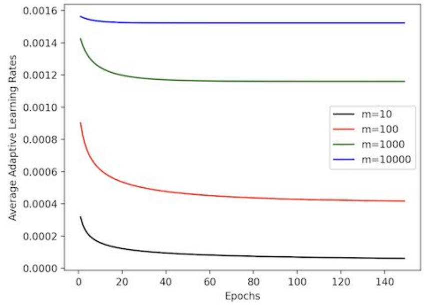


$max_{r \in m} ||w_r(t) - w_r(0)||_2$

Adagrad:

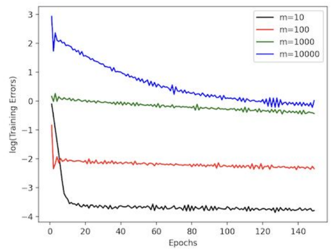


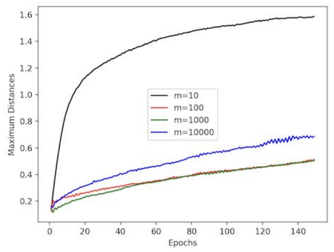


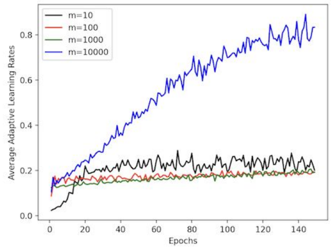


RMSProp Optimization

- Despite the flexibility gained by Adagrad optimization through the dynamic tuning of our model's learning rates, the accumulation of squared gradients can lead to a suboptimal stopping point in our network's training.
- RMSProp tries to address this by using an exponential moving average of squared gradients.
- By training our model through this optimization scheme, we found undesirable outcomes in the way the learning rate is tuned over time, and the overall convergence of our model, when compared to both SGD and Adagrad optimization.





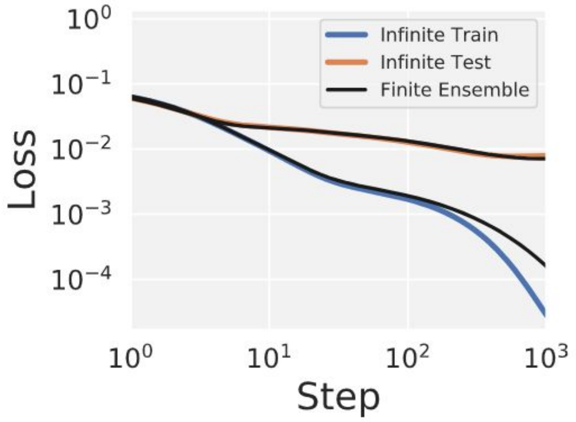


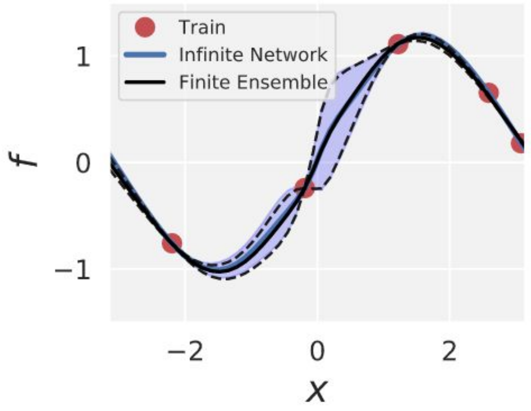
Analyzing Overparameterized Neural Networks and their performance

- We have demonstrated experimentally that increasing network width with SGD and Adagrad (not RMSProp) decreases the training error over time.
- In Du 2019, they have shown that this phenomenon appears almost akin to convex optimization, where you can reach a global minimum, assuming your learning rates are also optimized (i.e. don’t overshoot, a problem in traditional SGD, but improved with Adagrad).
- This is a curious phenomenon, in that traditionally neural networks are not functionally speaking convex. Practically speaking, that means that you can get stuck in local minima and never converge to the true minimum, which is corroborated by the results by Du and Arora.
- Nevertheless, this phenomenon needs to be studied more, and additional studies and methods have been performed and developed to understand overparameterized optimization better.
- One method, discussed in the next slide, is the Neural Tangent Kernel (NTK), which allows the study of neural networks using mathematical functions rather than parameters.

Neural Tangent Kernels (NTK)

- Infinite-width neural networks on training with gradient descent optimization can be computed using the NTK function.
- The NTK can be studied in function space, rather than solely in parameter space, and can be used to approximate an infinite-width neural network (helpful to understand why increasing layer width improves training performance).
- The NTK describes how the gradient descent evolves over time.
- Experiments help compare prediction of NTK theory with the result of training an ensemble of networks and observed agreement in the plot.



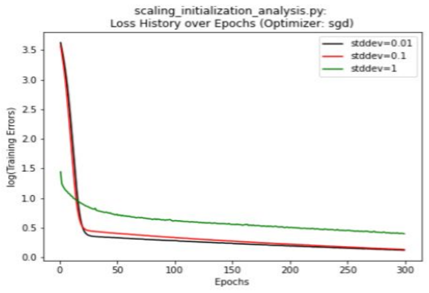


Overview of Lazy Training

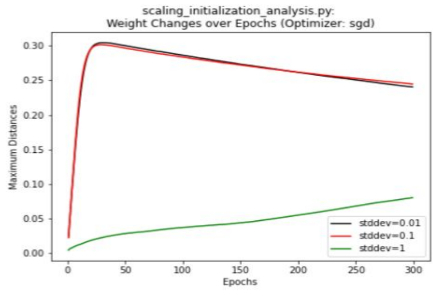
- So far we have seen that wide networks, by using gradient descent, can minimize training error.
- The motivation for studying lazy training was because, as discussed in Chizat 2019, that neural networks with weights initialized to 0 mean, $O(1/n_l)$ variance (previous layer width) converge quickly as layer width increases.
- This phenomenon has been interpreted as "Lazy Training", i.e. network weights don't move much after initialization, and the model behaves almost like a linearization around its initialization, making it equivalent to learning with positive-definite kernels (i.e. with strict convexity, so optimizing to the global minimum should be quick with the right learning rate and SGD function).
- However, this interpretation is incorrect, as explicitly scaling the movement of the weights causes degradation of performance in overparameterized 2-layer neural networks, as well as convolutional neural networks (Chizat 2019).
- We experimented weight initializations and lazy training (scaling output activation directly and loss inversely with $\alpha \geq 1$), and corroborate the observation that lazy training degrades training performance.

Scaled Model Experimentation

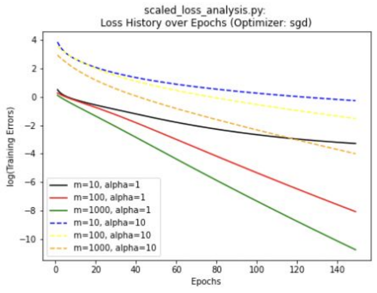
- Parameterized model can be trained in lazy regime if its weights are close to 0.
- Explored the use of varying node weight initialization as a precursor to scaling factor.
- Observed that keeping low variance (and mean = 0) for weights improves errors over epochs, though with large initial change in weights.
- Increased variance worsens generalization by increasing overfitting with increasing variance.
- Large scaling parameter→ smaller change in weights and not good convergence (lazy training)
- Deeper ReLU networks show smaller parameter changes and perform better.



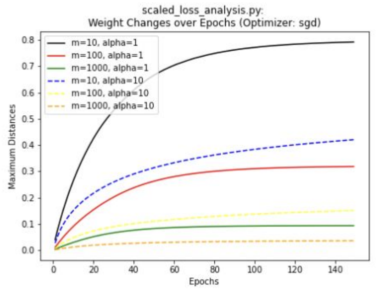
(a) Log Loss over Epochs



(b) Weight Change over Epochs



(a) Log Loss over Epochs



(b) Weight Change over Epochs

Generalization in Wide 2-Layer Neural Networks

- We have seen that hidden layer width in 2-layer neural networks is associated with better training error.
- However, do these models generalize well? Traditional machine learning theory implies that when training error bounds to the minimum, test error bounds in the opposite direction over time, hence the problem of overfitting (Shai, UMLT book).
- As a result, complexity of the learning model comes with a cost, i.e. generalization ability.
- In the case of wide, but shallow neural networks, some interesting results have been found, in particular that even while increasing complexity (layer width), the test error on validation test sets does not really degrade much even when increasing layer width, i.e. complexity (Neyshabur, 2017).

Generalization and Overparameterized Neural Networks Cont.

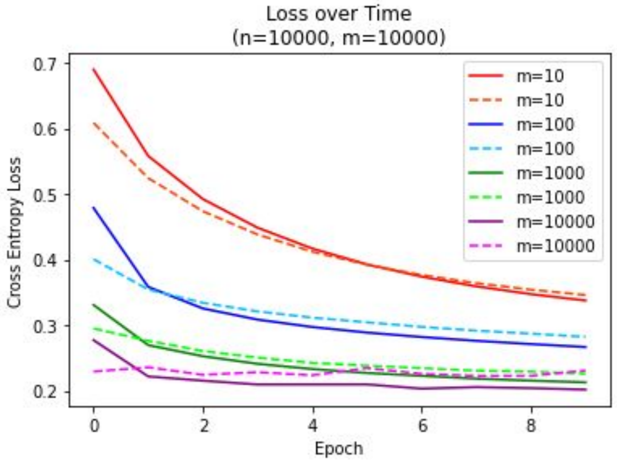
- PAC-learnability and VC theory can be used to explain how neural networks can be seen as a set of binary classifiers that can separate a dataset, and can set bounds on generalization error.
- One estimate of VC dimension of a neural network for binary classification is O(W*L*log(W)) and Ω(W*L*log(W/L) (Bartlett 2017). Practically speaking, training a model requires the sample size N to be much greater than the VC dimension to avoid overfitting.
- Based on the studies in Neyshabur 2017, however, the generalization error is well below what VC dimensions would predict.
- However, subsequent results have found that there is a limit in terms of how well a model can generalize as the hidden layer width increases (Weinan 2019).
- So, as with lazy training, it seems like there is a limit to how well a network will optimize, albeit in the scope of generalization error in this case rather than training error.

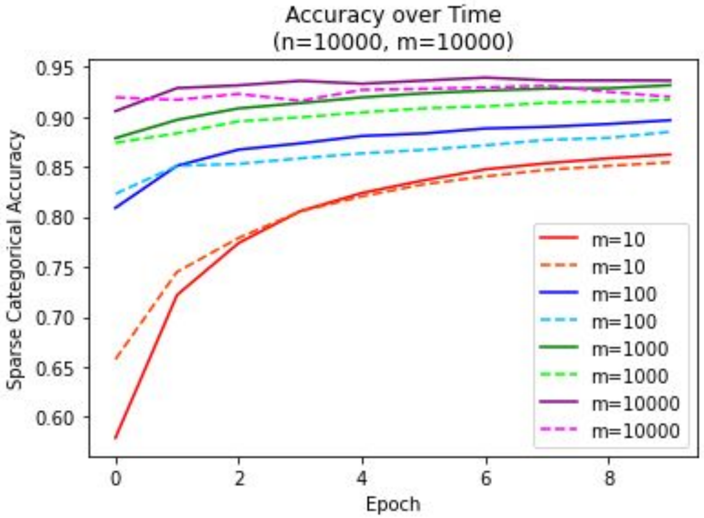
Experimenting with Generalization and Network Width

- In order to study the issues raised in the literature on this subject, we ran some experiments to check the generalizability of the networks.
- As before, we used 2-layer networks of varying hidden layer widths, and trained the models, but this time, we used validation sets to examine the potential training/test divergence, i.e. overfitting.
- Our results were mixed, particularly as epochs increased.

Results of Experiments

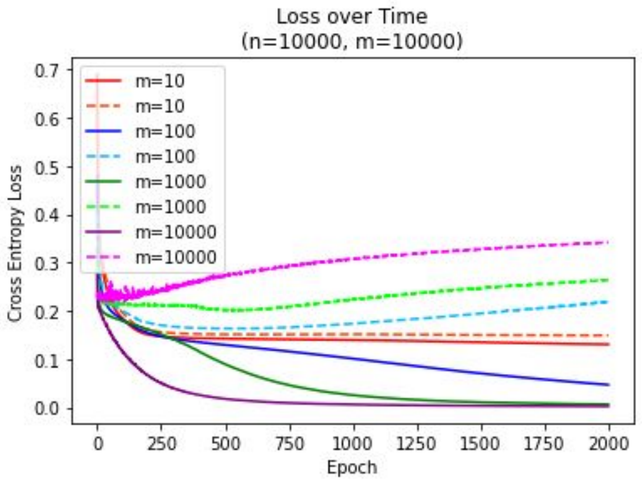
- For varying widths, we see that increasing width improves both training and test error, but only up to a point, as shown in the following plots (solid lines are training data, dashed lines are validation data). Data was randomly generated binary classification from SciKitLearn, of size 10000, and dimension 20.
-

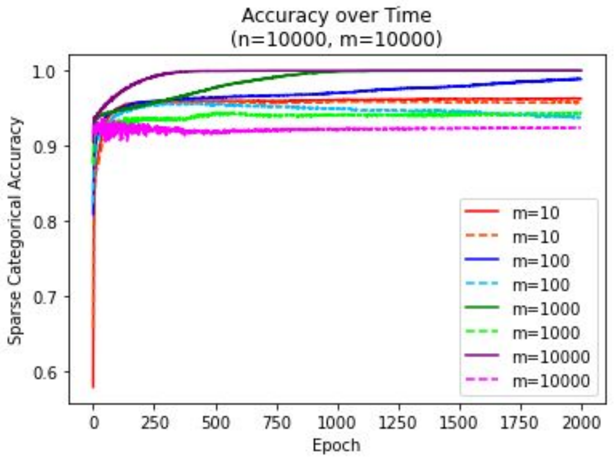




Results of Experiments (Cont.)

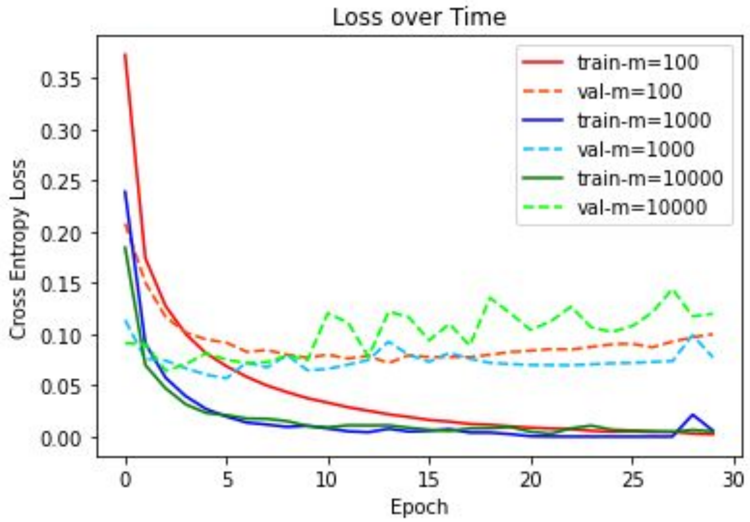
- The previous plot shows that early on, the wider networks are better for both training and test error, and that loss and accuracy is decent. This is in line with earlier ideas about how overparameterized networks converge quickly. Also, the size of the data is much smaller than what VC theory would tell us is optimal.
- However, it can be seen that there is a point, around epoch 5, that the widest (m=10000) network starts to degrade with respect to loss relative to the next widest network (m=1000).
- If we go even further in the epochs, the degradation is even worse, but seems to converge over time rather than expanding exponentially as seen in common overfitting models.

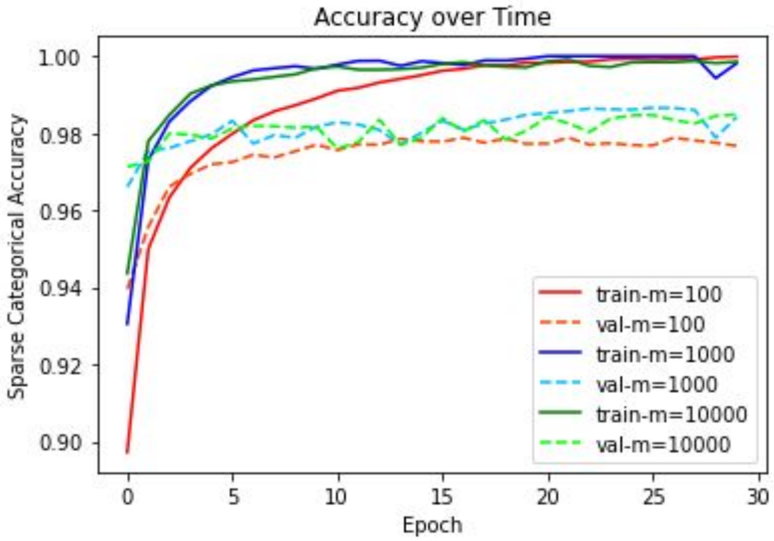




Additional Results

- Varying widths with the popular MNIST dataset (dim = 28x28=784, 60000 training examples, 10000 test examples).
- Since this is not binary, it could be described using Rademacher complexity. Nevertheless, generalization error is less than expected given the number of network parameters and the data size.





Further Discussion

- On one hand, overparameterized networks clearly overperform relative to traditional machine learning theory.
- On the other hand, we have seen that there is a limit in terms of how well you can generalize.
- As you keep training over time (epochs), training error is shown to converge to 1, while test/validation error decreases, but generalization error increases relatively slowly as seen in our graphs.
- As a result, we still need to include some forms of regularization, implicit or explicit. It has been shown in the papers by Du and Arora that the network width behaves like an implicit regularizer, possibly because input data is distributed along more neurons so that outliers may have less of an effect.
- Still, we may have to rely on common regularization techniques, such as L1, L2 regularization, and as seen from our graphs, early stopping. Weinan 2019 uses a modified form of L1 regularization, multiplying the L1 parameter by ln(data dimension)/data size.
- Overall, the results we have seen and experimented with show that neural networks don't behave as poorly as you would expect, but as results with lazy training and our generalization results show, there are limitations.