TRAINABILITY AND GENERALIZATION OF SMALL-SCALE NEURAL NETWORKS

By

MYUNG HWAN SONG

THESIS
Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Industrial Engineering
with a concentration in Advanced Analytics
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urabana, Illinois

Advisor:
Assistant Professor Ruoyu Sun

Abstract

As deep learning has become solution for various machine learning, artificial intelligence applications, their architectures have been developed accordingly. Modern deep learning applications often use overparameterized setting, which is opposite to what conventional learning theory suggests. While deep neural networks are considered to be less vulnerable to overfitting even with their overparameterized architecture, this project observed that properly trained small-scale networks indeed outperform its larger counterparts. The generalization ability of small-scale networks has been overlooked in many researches and practice, due to their extremely slow convergence speed. This project observed that imbalanced layer-wise gradient norm can hider overall convergence speed of neural networks, and narrow networks are vulnerable to this. This projects investigates possible reasons of convergence failure of small-scale neural networks, and suggests a strategy to alleviate the problem.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

Deep neural networks have undergone vast advances in recent years. Numerous machine learning, artificial intelligence tasks such as image classification, object detection, and natural language processing are now using deep neural networks as their main model. Architectures of deep neural networks have been developed accordingly with their application. Modern deep learning architectures often have hundreds of layers and millions of parameters. In case of image classification, ResNet[1], and DenseNet[2] models are much deeper than early VGG[3] or AlexNet[4]. In practice, the number of parameters of these architectures often go far beyond the number of training samples. The capacity that these overparameterized models can represent easily exceeds underlying complexity that input data can have.

Contrary to the practice, according to statistical learning theory, the gap between expected and empirical risk can grow as the model capacity grows. From this point of view, it is natural for those complex neural networks to become highly overfitted and have poor accuracy. However, despite of their overparameterized architecture, modern deep neural networks have shown nice generalization ability than their smaller counterparts. This is essentially due to various implicit/explicit regularization techniques. While explicit regularization techniques such as dropout[5], weight decay or data augmentation can improve the quality of solution, various researches also found implicit regularization effect of model architecture and learning algorithm such as batch normalization[6], skip connection[7], and SGD[8][9].

Still, there are many examples that reveal the limitation of generalization ability of complex neural networks. It is well known that various adversarial attacks can fool a deep learning model, even when one cannot directly approach to the model architecture and parameters [10][11]. Moreover, test score of a model can significantly decrease even when test data from the same distribution with training samples was used[12]. These issues can be thought as a result of overfitting of neural networks. From a theoretical perspective, one possible solution is decreasing model capacity with smaller models.

Nevertheless, the reason why overparameterized models are used is not just because of their seemingly nice generalization ability. One major advantage of deeper and wider models is that they can learn faster than small networks[13][14], with appropriate acceleration techniques and architecture. We observed that when neural networks are narrow and underparameterized, they often fail to learn in practical amount of time, even when they have enough capacity and were not converged to any stationary point (Figure 1.1). Small-scale networks are overlooked in many practices because they seems to generalize poorly due to their extremely slow convergence speed and considered to have insufficient capacity. To properly measure the generalization ability of small-scale networks, they should be trained until they reach to their full capacity.

This project investigates trainability and generalization ability of small-scale, underparameterized neural networks. While this is the concept that can be applied to any architectures and tasks, this work only considers models for image classification task, due to their success in deep learning and ease of implementation and analysis. Through carefully designed simulation, this project found the possible cause of convergence failure of training small-scale networks, and observed that balancing layer-wise gradient norm at initialization can improve convergence speed of small-scale networks. This project observed that overparameterized networks achieve high test accuracy partially due to their faster convergence speed, and if small-scale networks are trained properly, they can achieve even higher accuracy than overparameterized models.

In chapter 2, background of this project will be presented. Statistical learning theory, and various regularization techniques which improved quality of deep neural networks will be introduced. The trainability and possible reasons of convergence failure of small-scale

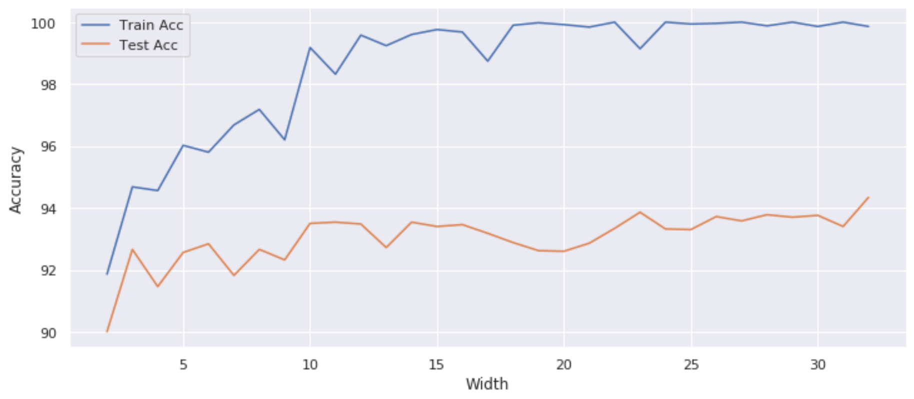

Figure 1.1: Even though models with any width in this figure have enough capacity to achieve 100% training accuracy, large-scale models are more easily optimized and therefore achieve higher test accuracy.

networks will be discussed in chapter 3. Chapter 4 will introduce initialization strategy to alleviate the problem. Finally, conclusion of this project will be made in chapter 5.

CHAPTER 2
BACKGROUND

2.1 Generalization Bound in Learning Theory

In learning theory, one conventional way of defining capacity of a machine learning model is VC dimension. VC dimension measures the capacity of a model by measuring the maximum number of n vectors which can be separated in all 2^n ways[15]. If VC dimension of a model is high, this implies that the model can classify large number of vectors in any possible ways.

One important result from learning theory about VC dimension is that the gap between empirical and expected risk can grow as the VC dimension of a model grows. With probability at least $1 - \eta$, gap between expected risk R and empirical risk R_n learned with n samples will be bounded by

$$\sup_{h \in H} |R(h) - R_n(h)| \leq O\left(\sqrt{\frac{1}{2n}\log\left(\frac{2}{\eta}\right) + \frac{d_H}{n}\log\left(\frac{n}{d_H}\right)}\right), \tag{2.1}$$

where H indicates a class of prediction functions, d_H indicates VC dimension of H.

While this result gives intuition about the relation between model capacity and its generalization bound, VC dimension is rarely used as a proper capacity measure for deep learning. This is because overparameterized networks, which can perfectly fit to training data with 100% accuracy, are often used in practice, and VC dimension can not explain the generalization ability of deep network well. Various researches observed that large-scale networks generalize well nevertheless of their high capacity[16][17], and Figure 1.1 also observed this.

2.2 Regularization in Deep Learning

In deep learning, overfitting due to overparameterization seems to be largely solved. In case of image classification, ISLVRC already achieved accuracy higher than human performance, stating that deep neural networks can generalize to unseen images better than human. On CIFAR-10 dataset, modern models achieve accuracy higher than 95%. These successful results are essentially due to various explicit and implicit regularization effects.

Explicit regularization techniques such as data augmentation, weight decay and dropout [5] are commonly used when training deep neural networks. Data augmentation increases number of training samples by generating new input data by slightly perturbing existing data while maintaining its meaning. Training with large number of data can obviously help closing the gap between training and test error. Weight decaying is also widely used regularization techniques not just for deep learning but for other various machine learning optimization. By suppressing trivial parameters to zero, it helps reducing the complexity and variance of a model. Dropout[5] was introduced for training deep learning to make better generalization performance. Eliminating some portion of activations while training prevents model to be overfitted with training samples.

More importantly, implicit regularization effect of training algorithm and model architecture helps generalization of deep learning. In high dimensional nonconvex space, escaping from sharp minimum turned out to be important for generalization[8]. Sharp minimum has smaller basin volume, whose function increases rapidly in a small neighborhood of the solution. Thus, sharp minimum is more sensitive to new unseen data, while flat minima has more potential to generalize better to new data. From this perspective, [8] observed that training deep neural networks with large batch SGD can damage the generalization ability of the model. They argued that inaccurate gradient direction caused by small mini-batch helps the model to escape from sharp minimum. Also, [6] stated that the average gradient noise of SGD is upperbounded as follows.

$$\mathbb{E}[||\alpha\nabla l(x) - \alpha\nabla_{SGD}l(x)||^2] \leq \frac{\alpha^2}{|B|}C, \tag{2.2}$$

where α is learning rate and B is size of mini-batch. Based on sharp/flat minima concept, increasing gradient noise can help escaping from sharp minimum. Therefore, using small batch size or increasing learning rate can help generalization. Various training algorithms were able to get better generalization result using this concept. [18] and [19] found out that cyclically resetting learning rate to its original value after decaying helps model to get better solution. One explanation of this phenomenon is that resetting the learning rate to large value helps the model to escaping from sharp minimum, while large leaning rate also makes convergence much faster[20].

Batch normalization[21] not only makes training faster, but also has implicit regularization effect. [6] found that batch normalization makes the gradient distribution small and balanced. Making gradient small enables model to use bigger learning rate, helps escaping from sharp minimum and converging fast. Also, [7] observed that skip connection makes the overall loss landscape more convex-like and flat, which makes overall loss landscape favorable. [22] found that loss landscape of deep network is basically dominated by flat minimum, therefore random initialization will almost surely leads to solution with good generalization.

2.3 Overfitting in deep learning

However, even with all of above regularization effect, many researches pointed out the limitation of generalization ability of deep neural networks. Adversarial examples created with fast gradient sign method (FGSM) can fool a deep network to misclassify even though the recreated images looks identical to human eye[10]. [11] observed that deep neural networks cannot generalize to the images created by spatial transformation, and [12] found that CIFAR-10 classifiers does not generalize very well even for images from same distribution.

Geometrically speaking, training a machine learning model for a classification task is a process of defining separating surfaces which can divide given input data according to their labels. A common way to define a separating surface is to maximize margin between labels, which can be seen as maximizing thickness of the separating surface. Intuitively, separating surface with maximum margin can generalize well to unseen data, because new data are likely to be nearly distributed with training samples, if those data are from the same distribution.

Training a deep neural network is basically same process. With appropriate loss function, the training process tries to generate a separating surfaces with large margin. However, since overparameterized networks have extremely high capacity, they have enough possibility to be overfitted with training samples, and fail to generalize. Adversarial examples are intentionally generated input data, which lies very near to the existing training sample, but on different side of separating surface. From this point of view, one possible solution is to make large margin and simpler separating surface with small-scale models.

2.4 Related Works

Number of researches observed that deep and overparameterized networks generalize better than smaller networks[16][17][23]. Nevertheless, this projects covers generalization ability of small-scale networks due to following reasons.

1. To compare the generalization ability of multiple models, they should be trained until they reach comparable training accuracy.

2. Small-scale networks are often difficult to be trained, therefore, previous researches may not have used properly trained small-scale networks for their comparison.

Actually, there have been number of trials to make neural networks small. Knowledge transferring or knowledge distillation, tries to transfer the behavior of a large-scale network into smaller one[24][25]. [26] tried to compress complex model by pruning neurons without affecting model performance. However, these approaches are different from this work, in the sense that they tried to train small networks in order to make them easy to be exploited with small hardware resource. This project, on the other hand, focus more on the trainability and generalization ability of small-scale networks.

CHAPTER 3
TRAINABILITY OF SMALL-SCALE NEURAL NETWORKS

Before considering about small-scale networks, concept of *small* should be clarified. There are many ways to control the capacity of neural networks, and simply measuring the capacity by counting its number of parameters can obfuscate the relationship between model capacity and generalization ability[17]. Even when multiple models have same number of parameters, depth of neural networks is considered closely related to generalization ability. In case of image classification, it is well known that former layers learn simple features of images such as vertical/horizontal lines, or curves. Latter layers learn more complex features, such as letters, or shape of face. Finally at the last layer, the network decides what is the most probable class of the given input[27]. The basic features learned in former layers help the model to generalize well on unseen data, because those simple features are likely to be also exist in new images. Transfer learning[28] and parameter sharing[29] essentially use this property of deep architecture. Therefore, not to damage the generalization ability while reducing model capacity, this project only considered underparameterization through narrowing down the number of neurons in each layer. In case of convolutional layers, this is equivalent to reducing the number of output channels.

In order to compare the pure generalization ability of multiple neural networks, it is necessary to train them until they get comparable training accuracy. When two models achieve 0 training loss, or at least near 0 loss, the test score of the two models can be compared. However, we observed that small-scale networks are difficult to be trained up to 0 training loss. They usually take much more time to achieve same degree of training accuracy than large networks, and even fail to converge. There can be a few possible reasons. First, models should have enough representation power to fully represent the underlying complexity of input data. If the model has too small capacity to achieve low empirical risk, then it obviously cannot achieve good test accuracy either.

Also, since deep neural networks lie on high dimensional nonconvex space, its loss landscape can have many obstacles that interrupt models from converging to good solution. For instance, gradient based first-order optimization algorithms cannot distinguish local minimum with global minimum. Theoretically, if a network has converged to bad local minimum with high loss value, it would have poor training/test accuracy, even when the model has enough representation power. Moreover, even though saddle points are not attractors of first-order gradient methods, it can seriously slow down the convergence speed, which makes unable to train the model in practical amount of time [30]. In addition to these, other undefinable and unfavorable landscape can always slow down convergence.

Finally, there can exist optimization issues, which can be characterized by ill-conditioned hessian matrix and inappropriate learning rate. Large and unstable Lipshitz constant L can hinder stable convergence of optimization algorithms, makes training extremely slow. Inappropriate learning rate also can slow down convergence speed or even make the model diverge. From a theoretical perspective, optimal constant learning rate would be $1/L$. If learning rate is too far away from $1/L$, the model will diverge or will converge but in extremely slow speed. However, Lipshitz constant is difficult to be computed in deep learning practice due to its computational complexity. While local Lipshitz constant can be computed by $L = \lambda_{max}(\nabla^2(F(\theta)))$, (where $F = loss(f(\theta;x),y)$, f is the prediction function), computing maximum eigenvalue of hessian matrix is difficult due to limited hardware resource and computation time.

To diagnose the possible reasons of convergence failure of small-scale networks, an experiment with artificial data was designed. Teacher and student networks were used to remove unnecessary factors and make the problem clear. For input data, re-scaled CIFAR-10 data was used, but the label of each image was reassigned with the teacher network. The teacher network is a small-scale network and randomly initialized with unit normal distribution $N(0,1^2)$. Binary label was assigned according to the output of the teacher network.

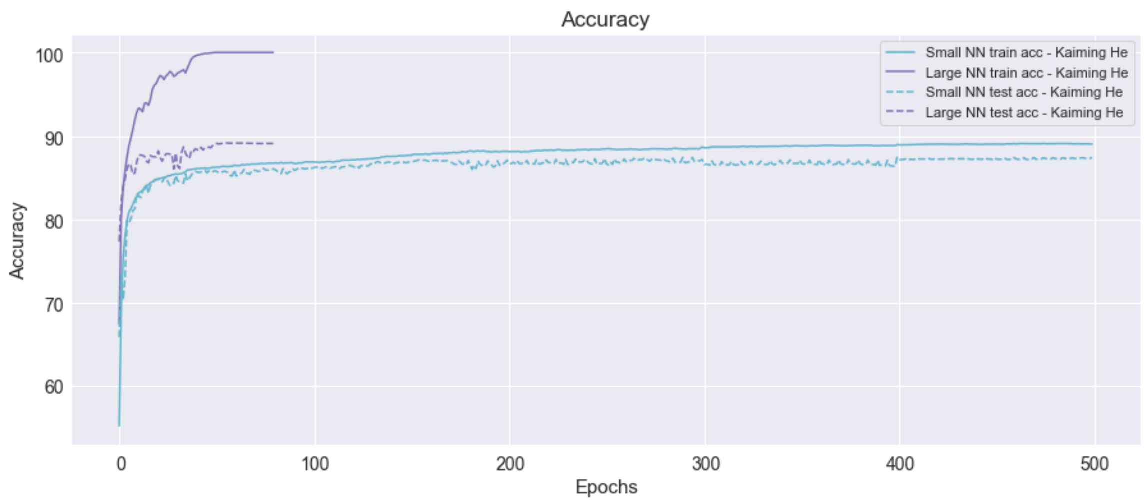

Figure 3.1: Learning curve of two models with different width. Although small-scale network has enough representation power to achieve 100% training and test accuracy, it failed to converge to 100% training accuracy. Large-scale network, on the other hand, achieved 100% training accuracy in short time, also achieved better test accuracy.

Smooth hinge loss was used for this task to make exact 0 loss for perfect classification. Thus, the weight of the teacher network can be seen as one of perfect solutions of the problem, which can achieve 0 loss and 100% accuracy for every training and test sample. Then student networks were generated and trained with created artificial data. Since all the label was reassigned, what student networks try to learn is irrelevant to the content of the images. Detailed simulation setting and architecture are introduced in the appendix A.

Multiple student networks with different width were generated and trained. The training was stopped when the training accuracy achieved 100% or reached long plateau. The brief result of this training is described in Figure 1.1. The first student network is a small-scale network with the exactly same architecture with the teacher network. Therefore, this network is guaranteed to have solution with 0 loss and 100% training/test accuracy, by achieving the same weight with the teacher. The rest of the student networks have wider width than the teacher. These networks are also guaranteed to have perfect solution, which can be easily achieved by setting 0 to all additional parameters.

Figure 3.1 shows the learning curve of two student networks. The small-scale student network failed to achieve 100% training accuracy while the large-scale network with wide width easily achieved. The test accuracy of the small-scale network is lower than that of the large network either. However, to compare the generalization ability of the two network, the small-scale network should be trained until it has comparable training accuracy with the large-scale network. Since the small student network has enough capacity, lack of representation power is not a reason for the convergence failure. Also, it has not converged to any stationary point. Figure 3.2 shows the gradient norm changes throughout the training process. If the small-scale network has converged to any local minimum, its gradient norm should be near 0 as the graph of large-scale network shows. This indicates that small-scale network is still converging toward some stationary point. It is theoretically possible that the small-scale network will eventually reach to 100% training accuracy with longer training, but it is not at all practical because it will take enormous amount of time.

Since the small-scale network only has small number of parameters, complete hessian matrix throughout the training process could be computed. Figure 3.3 shows the maximum eigenvalue change of the hessian matrix throughout the training. Maximum eigenvalue indicates local Lipshitz constant, which is closely related to the convergence speed of gradient descent method. The local Lipshitz constant is gradually increasing in manageable range, which corresponds with common strategy used in practice (gradually decrease learning rate). Even when theoretically derived learning rate $1/L$ was used for the small-scale network, the convergence speed was not noticeably improved. This indicates the possible reason for convergence failure of small-scale network is not because of unstable Lipshitz constant, nor improper learning rate.

Taken together, the small-scale networks fail to converge because of complex, unfavorable loss landscape, which can occur in high dimension space. More specifically, this project observed that when layer-wise gradient norm is unbalanced, convergence speed can become much slower (Figure 4.5). Geometrically, this can occur when a current solution is lying in the space where landscape is abnormally steep in certain directions. We found that small-scale, narrow networks are especially vulnerable to this problem.

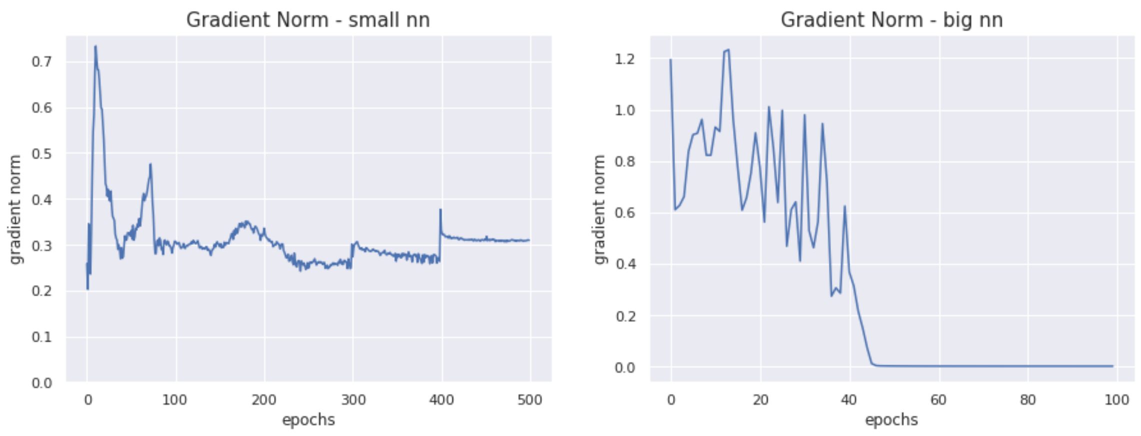

Figure 3.2: Gradient norm changes of two student networks. Small-scale network has not converged to any stationary point, while large-scale network has converged to solution with 100% training accuracy.

There can be two possible ways to cope with unfavorable loss landscape: by changing the landscape itself, and carefully choose initial point so that the model can easily approach to potential solution. Loss landscape is decided by model architecture and input data. Considering that input data is normalized in practice, we can assume that input data always follows normal distribution $N(0, 1)$ in general scenario. Thus, one can improve landscape by changing model architecture. For instance, [7] observed that skip connection makes overall landscape convex-like and flat. [6] found that using batch normalization makes Lipshitz constant small, which implies smoother loss landscape. However, even with above two techniques, small-scale networks were difficult to be trained in this simulation. This project focus more on the latter solution, careful initialization, with balanced layer-wise gradient norm.

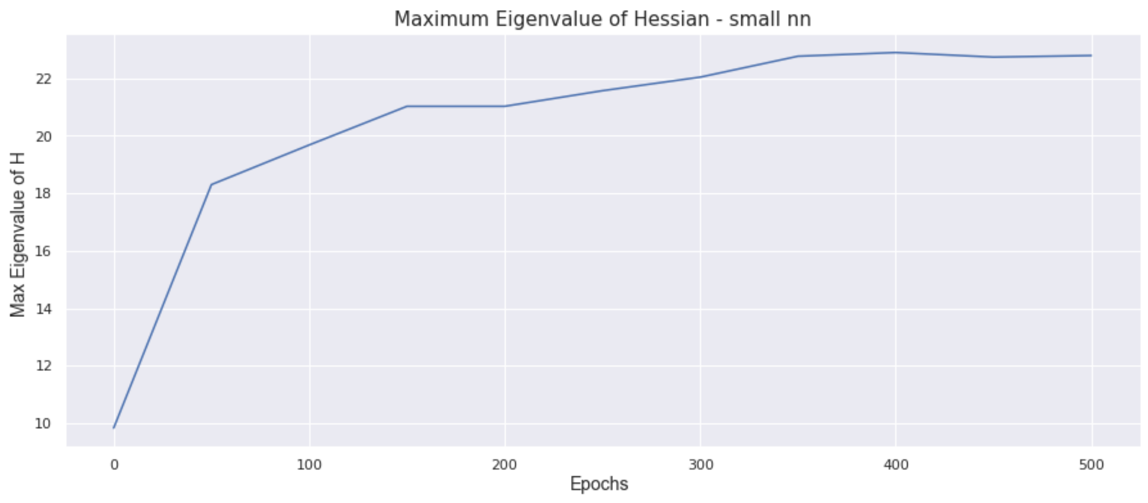

Figure 3.3: Maximum eigenvalue change of the small-scale student network. Lipshitz constant gradually increases as the training proceeds in manageable range.

CHAPTER 4

GENERALIZATION OF SMALL-SCALE NETWORKS

This project observed that balancing layer-wise gradient norm is important for training neural networks. Intuitively, extremely unbalanced layer-wise gradient norm indicates unstable gradient propagation, and makes huge difference in the convergence speed of different parameters. Batch normalization and skip connection helps training in this perspective, by making layer-wise gradient small and balanced at initialization (Figure 4.4).

Balancing layer-wise gradient norm can improve convergence speed of small-scale networks. In the teacher-student simulation, the small student network was able to reach comparable training accuracy with the large student network, when its layer-wise gradient norm was balanced at initialization (Figure 4.1). The gradient norm of each layer was simply manipulated by changing variance of each layer. Figure 4.2 and Table 4.1 shows the result of the trained small-scale student network. The small-scale student network now achieved near 100% training accuracy, and also achieved higher test accuracy than the large-scale student network. This simulation shows that properly trained small-scale

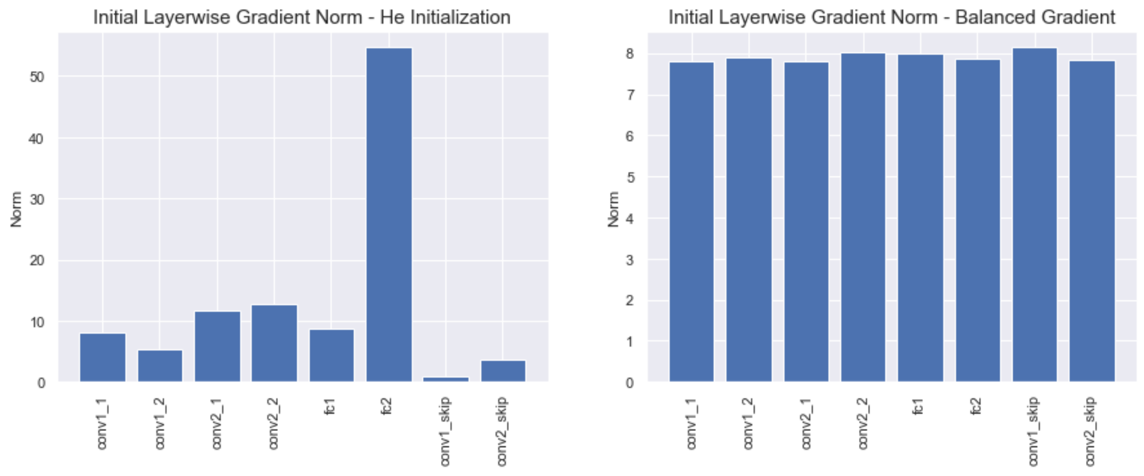

Figure 4.1: Initial layer-wise gradient norm distribution. Make gradient norm balanced and small is important for training small-scale networks.

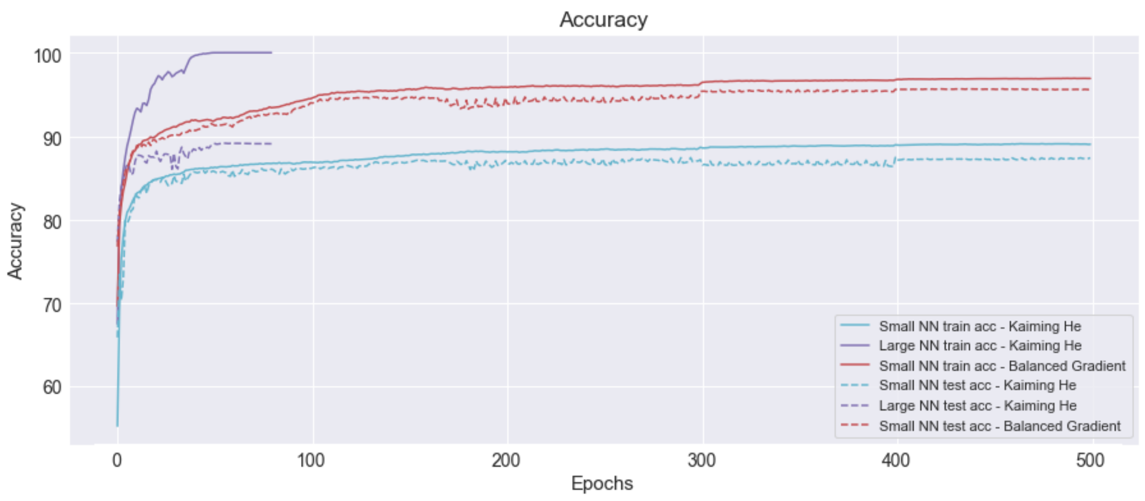

Figure 4.2: Learning curve of small and large-scale neural networks. Small-scale network with proper initialization can converge faster. Also, it can achieve better test accuracy than overparameterized setting.

network can indeed achieve better generalization ability than large and overparameterized networks, which corresponds with the intuition from learning theory.

The reason why balancing layer-wise gradient norm helps training is because it allows all parameters to learn evenly. Figure 4.5 shows the layer-wise gradient norm changes for the first five epochs of training. When standard initialization was used to small-scale networks, some layers learn much slower than the others, which hinders overall training speed. This can be partially solved by balancing layer-wise gradient norm at initialization as showed in the second graph of the Figure 4.5. However, this trick did not noticeably boosted the training speed of large-scale networks. This is because that narrow networks are more vulnerable to unbalanced gradient, and wide networks are usually have more balanced layer-wise gradient norm. This can be also simply observed in a simulation with

	Train		Test	
	Accuracy	Loss	Accuracy	Loss
Small NN - He init	88.60%	0.3710	87.08%	0.4236
Small NN - Balanced Grad	97.39%	0.0882	96.40%	0.1370
Large NN - He init	100.00%	0.0000	88.99%	1.8944

Table 4.1: Final accuracy and loss of trained student networks.

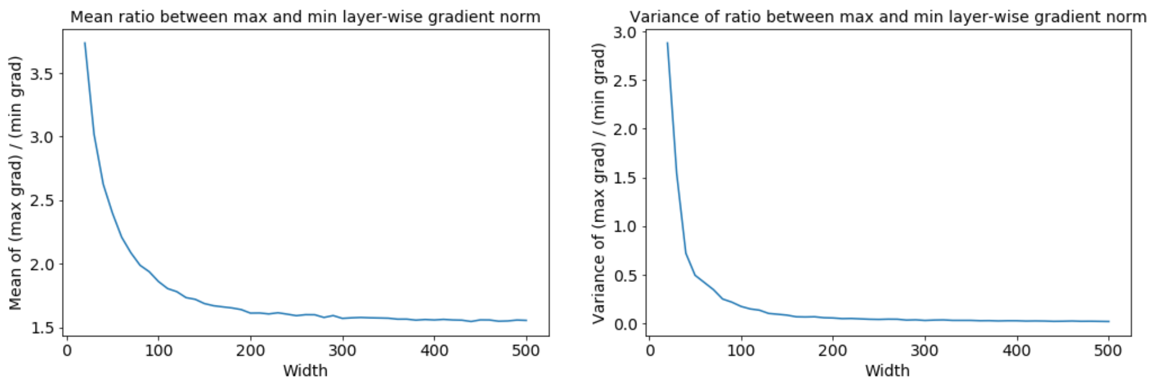

Figure 4.3: Mean and variance of ratio between maximum and minimum layer-wise gradient norm in a 10-layer fully connected linear network. As the width of the network increases, the layer-wise gradient norm becomes more stable and balanced.

a 10-layer fully connected linear network, with Xavier[31] initialization. Figure 4.3 shows the mean and variance of ratio between maximum and minimum gradient norm among all layers. As width of the network increases, the ratio becomes small and stable, implies that wider networks have more balanced layer-wise gradient norm. On the other hand, small-scale networks tend to have imbalanced layer-wise gradient norm, which can lead to slow convergence.

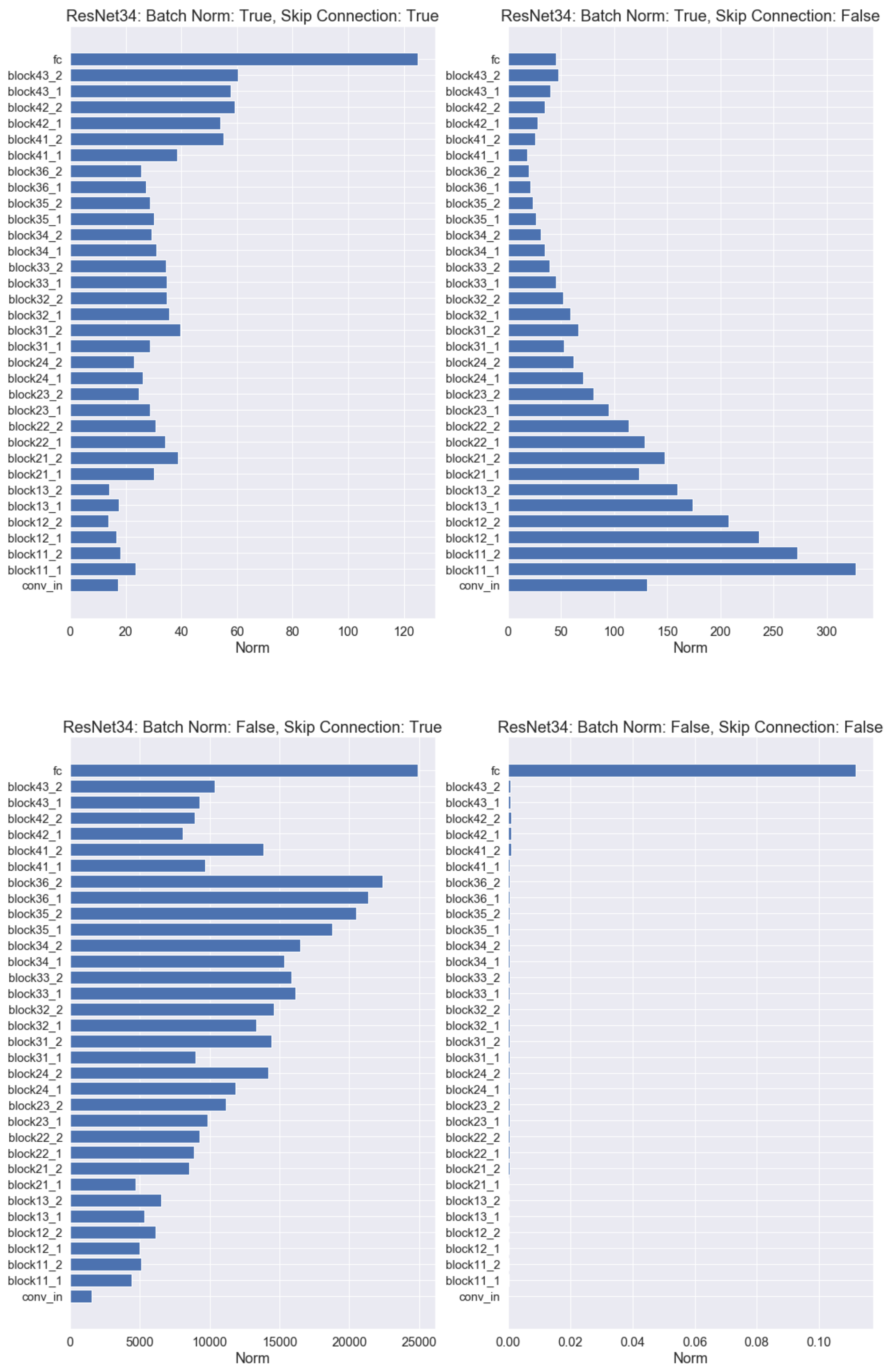

Figure 4.4: Layer-wise gradient norm of ResNet34 at initial point. Batch normalization and skip connection helps gradient norm to be small and balanced.

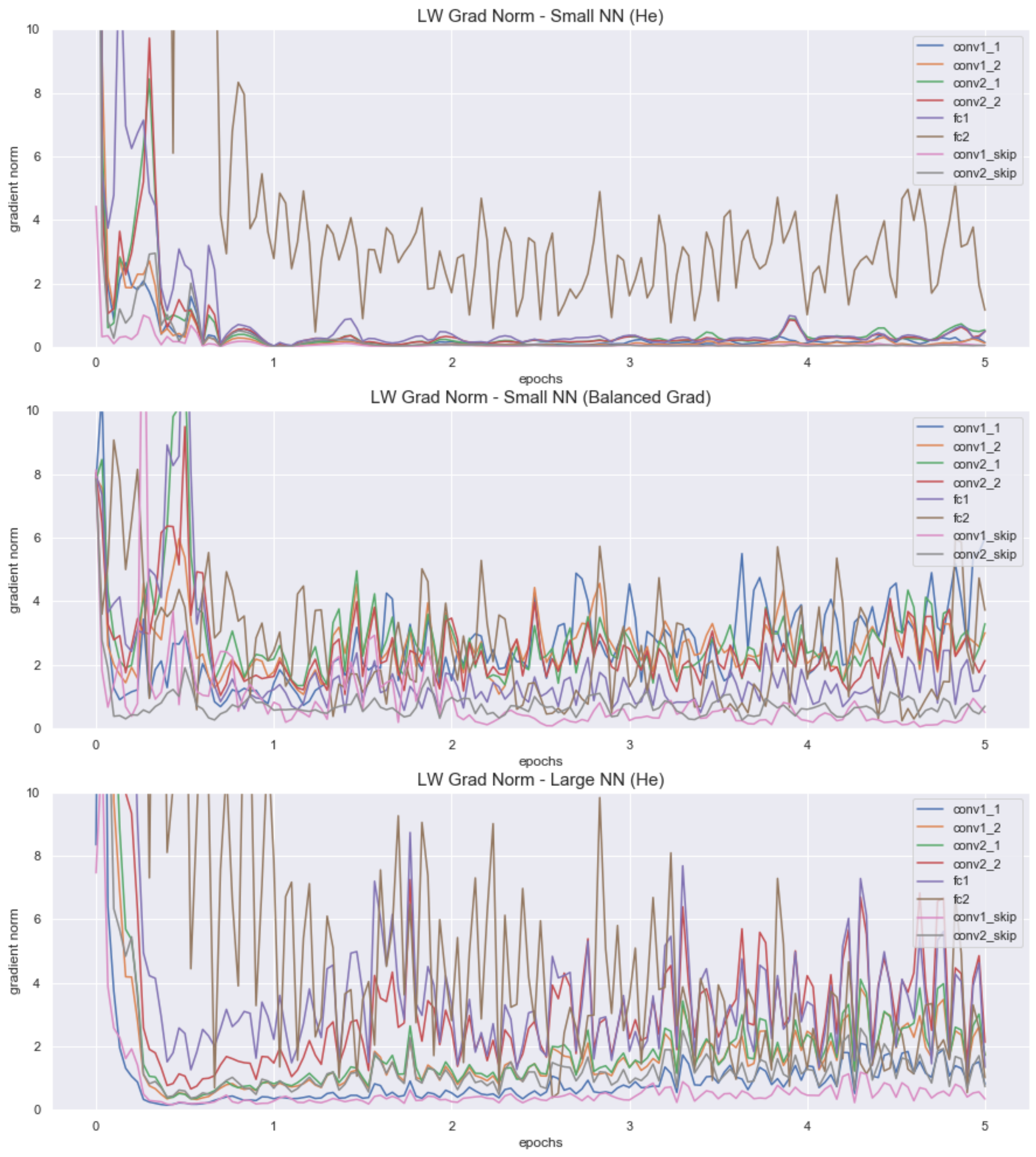

Figure 4.5: Layer-wise gradient norm changes of the first 5 epochs of training. Imbalanced layer-wise gradient norm of small-scale networks makes its convergence slow.

CHAPTER 5
CONCLUSION

This project investigates trainability and generalization ability of small-scale neural networks. While intuition from statistical learning theory suggests to use small-scale network to avoid overfitting, deep neural networks were considered exception to this. However, this project observed that small-scale networks are difficult to be trained, therefore, comparison of generalization ability implemented in many researches so far may have used not fully trained small-scale networks. Small-scale networks can fail to learn, even with the following situations.

- The network has not converged to any stationary point.
- The gradient norm has not exploded or vanished.
- The network has enough capacity to represent whole training samples.
- The Lipshitz constant is stable and small.
- Learning rate approximately match with $1/L$.

It is often considered in practice that small-scale networks are converged to local minimum or have insufficient capacity when they fail to converge, but our simulation suggests that extremely slow convergence speed of small networks also can make them look that way. This project observed that one possible reason of the slow convergence is imbalanced layer-wise gradient norm, which turned out to be especially significant issue in narrow networks. When layer-wise gradient norm was balanced at initialization, we observed that small-scale networks converge faster than standard initialization, and also achieve higher accuracy than their wider counterparts. This indicates that small-scale networks are not generalize poorly as widely known.

This project cast a doubt on a current belief about generalization of deep neural networks. While this concept can be applied to any task and architecture, this project only considered image classification task. More comprehensive research with various architectures and tasks will be left to future work. For instance, deep neural networks are often less powerful in tasks with structured data set than decision tree based models because of overfitting. Stable initialization scheme which makes balanced layer-wise gradient norm for small networks should be studied as well.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[2] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[3] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *ArXiv preprint arXiv:1409.1556*, 2014.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[5] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[6] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger, “Understanding batch normalization,” in *Advances in Neural Information Processing Systems*, 2018, pp. 7705–7716.

[7] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, “Visualizing the loss landscape of neural nets,” in *Advances in Neural Information Processing Systems*, 2018, pp. 6391–6401.

[8] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, “On large-batch training for deep learning: Generalization gap and sharp minima,” *ArXiv preprint arXiv:1609.04836*, 2016.

[9] R. Kleinberg, Y. Li, and Y. Yuan, “An alternative view: When does sgd escape local minima?” *ArXiv preprint arXiv:1802.06175*, 2018.

[10] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” *ArXiv preprint arXiv:1412.6572*, 2014.

[11] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry, “A rotation and a translation suffice: Fooling cnns with simple transformations,” *ArXiv preprint arXiv:1712.02779*, 2017.

[12] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, “Do cifar-10 classifiers generalize to cifar-10?” *ArXiv preprint arXiv:1806.00451*, 2018.

[13] S. Arora, N. Cohen, and E. Hazan, “On the optimization of deep networks: Implicit acceleration by overparameterization,” *ArXiv preprint arXiv:1802.06509*, 2018.

[14] B. Hanin and D. Rolnick, “How to start training: The effect of initialization and architecture,” in *Advances in Neural Information Processing Systems*, 2018, pp. 571–581.

[15] V. N. Vapnik, “An overview of statistical learning theory,” *IEEE transactions on neural networks*, vol. 10, no. 5, pp. 988–999, 1999.

[16] B. Neyshabur, R. Tomioka, and N. Srebro, “In search of the real inductive bias: On the role of implicit regularization in deep learning,” *ArXiv preprint arXiv:1412.6614*, 2014.

[17] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro, “Exploring generalization in deep learning,” in *Advances in Neural Information Processing Systems*, 2017, pp. 5947–5956.

[18] I. Loshchilov and F. Hutter, “Sgdr: Stochastic gradient descent with warm restarts,” *ArXiv preprint arXiv:1608.03983*, 2016.

[19] L. N. Smith, “Cyclical learning rates for training neural networks,” in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2017, pp. 464–472.

[20] L. N. Smith and N. Topin, “Super-convergence: Very fast training of residual networks using large learning rates,” 2018.

[21] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *ArXiv preprint arXiv:1502.03167*, 2015.

[22] L. Wu, Z. Zhu, *et al.*, “Towards understanding generalization of deep learning: Perspective of loss landscapes,” *ArXiv preprint arXiv:1706.10239*, 2017.

[23] R. Novak, Y. Bahri, D. A. Abolafia, J. Pennington, and J. Sohl-Dickstein, “Sensitivity and generalization in neural networks: An empirical study,” *ArXiv preprint arXiv:1802.08760*, 2018.

[24] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *ArXiv preprint arXiv:1503.02531*, 2015.

[25] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, “Fitnets: Hints for thin deep nets,” *ArXiv preprint arXiv:1412.6550*, 2014.

[26] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding,” *ArXiv preprint arXiv:1510.00149*, 2015.

[27] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson, “Understanding neural networks through deep visualization,” *ArXiv preprint arXiv:1506.06579*, 2015.

[28] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1717–1724.

[29] H. Pham, M. Y. Guan, B. Zoph, Q. V. Le, and J. Dean, “Efficient neural architecture search via parameter sharing,” *ArXiv preprint arXiv:1802.03268*, 2018.

[30] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, “Identifying and attacking the saddle point problem in high-dimensional non-convex optimization,” in *Advances in neural information processing systems*, 2014, pp. 2933–2941.

[31] X. Glorot and Y. Bengio, “Understanding the difficulty of training deep feedforward neural networks,” in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

APPENDIX A
EXPERIMENTAL DETAIL

Here I introduce detailed setting for the teacher-student experiment. The architecture of the model was composed with two basic ResNet block, followed by fully connected layer, and output layer. The teacher network has width of 4, and has 721 parameters in total. The smallest student network used in Figure 3.1 has the exact same architecture with the teacher, and the large-scale student network has width of 64, has 134,401 parameters in total.

The teacher network was initialized with standard normal distribution. Input data was feed to the teacher network and artificial label was assigned according to the output. For the input data, re-scaled CIFAR-10 images were used. The images are re-scaled to 8×8 to make models to have small number of parameters compared to the number of samples, and to make computation faster. Polynomial smooth hinge loss, $l(z) = (1+z)^p$, was used in this simulation. This function returns exact 0 to the data that are farther than 1(margin) from the separating surface, therefore, artificial data with exact 0 loss can be created. Those data samples that give positive loss value were just dropped. The number of generated training samples was 30,000, and test samples was 10,000.

In the training process, piece-wise linear learning rate with momentum was used and the learning rate was divided by 10 after the training accuracy reached plateau. The initial learning rate was set to 0.1, and the momentum was set to 0.9.

APPENDIX B
TRAINING SMALL-SCALE NETWORKS ON GENERAL DATA

It was difficult to directly apply the result from the simulation to general image data. The main reasons is because it is hard to know the underlying complexity of data set. One cannot easily know whether the convergence failure of small-scale networks is due to lack of model capacity or due to other reasons. However, for a simple task, this project was able to observe a similar pattern using CIFAR-10 data set. Figure B.1 and Table B.1 shows the result of training on CIFAR-10 binary classification task. In order to completely control the number of training samples, no data augmentation was used. As the acceleration effect gained from this initialization is limited and naively changing variance of each layer damaged forward signal propagation intended in the original initialization scheme, small-scale networks could not catch up overwhelmingly fast convergence speed of large-scale networks for more complicated tasks. However, this example still shows that properly trained small-scale networks can result to better generalization performance than its large-scale counterparts.

For this experiment, model with two basic ResNet blocks followed by output layer was used. The small-scale networks here has width of 3, has 589 parameters in total. The large-scale network has width of 48, has 69,889 parameters in total. The input images are from original CIFAR-10, but only two classes of them were used. Therefore, the number of training samples is 10,000 and test samples is 2,000.

	Train		Test	
	Accuracy	Loss	Accuracy	Loss
Small NN - He init	92.63%	0.2507	89.15%	0.3836
Small NN - Balanced Grad	95.67%	0.1436	93.15%	0.3230
Large NN - He init	100.00%	0.0030	91.8%	0.3572

Table B.1: Final accuracy and loss on CIFAR-10 binary classification.

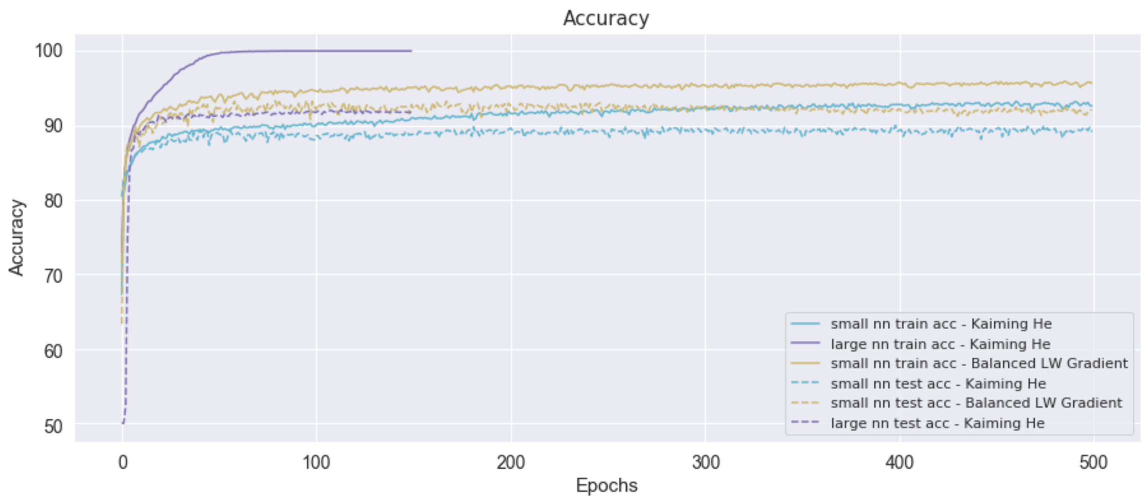

Figure B.1: Learning curve of small and large-scale networks. When layer-wise gradient norm is balanced, small-scale networks can learn faster.