IE 663: Advanced Topics in Deep Learning **Jan-May 2023**

End-term Project Report: Efficient Training of Low-Curvature Neural Network

Team Name: Supernova *Team Members:* Tarun Bisht

Abstract

This project studies a new approach called low-curvature neural networks (LCNNs) [10] to tackle issues such as low adversarial robustness and gradient instability in standard deep neural networks. LCNNs demonstrate lower curvature compared to standard models while maintaining similar predictive performance, leading to improved robustness and stable gradients. The authors decompose overall model curvature in terms of the constituent layer's curvatures and slopes. They also introduce two novel architectural components: the centered-softplus non-linearity and a Lipschitz-constrained batch normalization layer. We compare the performance and adversarial robustness of LCNNs in the MNIST dataset. We then propose parametric swish non-linearity. Our experiments showed that centered-softplus proposed by the author is better than our proposed parametric swish.

1 Introduction

The high degree of flexibility present in deep neural networks is critical to achieving good performance in complex tasks such as image classification, language modeling, and generative modeling of images. However, excessive flexibility is undesirable as highly nonlinear models can have a high degree of sensitivity to small changes in the input, which can lead to large changes in the output. This affects the adversarial robustness of the model. Also, highly non-linear models can also suffer from exploding and vanishing gradient problems. The paper for this project work focuses on training neural network models without excess non-linearity in their input-output map, such that predictive performance remains unaffected using the idea of curvature and introduces training of low curvature Neural Networks.

Curvature is a mathematical quantity that encodes a function's flexibility or degree of nonlinearity at a point. In deep learning, the curvature of a function at a point is often quantified as the norm of the Hessian at that point. Hessian norms are zero everywhere if and only if the function is linear, making them suitable to measure the degree of nonlinearity. However, this measure suffers from the dependence on the scaling of model gradients. To be able to study robustness independent of nonlinearity, the authors proposed normalized curvature, which normalizes the Hessian norm by its corresponding gradient norm.

There are several approaches to training models with low-curvature input-output maps. One approach is to directly penalize curvature locally at training samples, as proposed by previous studies [7, 8]. However, this method involves expensive Hessian computations and minimizes local point-wise curvature rather than everywhere. Another approach proposed by Dombrowski et al. [2] involves using architectures with small global curvature but does not explicitly penalize this curvature during training. The author's proposed approach involves efficient mechanisms for globally penalizing normalized curvature. Furthermore, while previous methods [7, 3] penalize the Frobenius norm of the Hessian, authors penalize its spectral norm, which provides tighter and more interpretable bounds on robustness.

We provide a survey of existing literature in Section 2. Our proposal for the project is described in Section 3. We give details on experiments in Section 5. A description of future work is given in Section 7. We conclude with a short summary and pointers to forthcoming work in Section 8.

2 Literature Survey

2.1 Curvature of a Model

In mathematics, the curvature is the amount by which a curve deviates from being a straight line. Intuitively, it is a measure of how much a curve or surface is bending at a particular point. Hessian norm at that point provides a measure of curvature at that point. Hessian norms are zero everywhere if and only if the function is linear. Therefore, curvature $C_f(x) = 0 \ \ \forall x \in \mathbb{R}^d \iff f$ is linear.

In the machine learning (ML) context, curvature measures the degree of nonlinearity of a model. A common measure of the curvature of ML models can be described via Hessian norms

$$||\nabla_x^2 f(x)||_2$$

Since the Hessian norm measure suffers from a dependence on the scaling of model gradients, authors have proposed normalized curvature, which normalizes the Hessian norm by its corresponding gradient norm.

$$\frac{||\nabla_x^2 f(x)||_2}{||\nabla_x f(x)||_2}$$

2.2 Adversarial Robustness of Neural Networks

Adversarial vulnerability of neural networks is a phenomenon that shows that adding small amounts of imperceptible noise can cause deep neural networks to misclassify points with high confidence. One evident method to defend against this vulnerability is adversarial training [6], which trains models to accurately classify adversarial examples generated via an attack such as projected gradient descent (PGD). However, this approach is computationally expensive and provides no formal guarantees of robustness. Previous studies, such as randomized smoothing [1], local Lipschitz constant identification [5], penalization of the Frobenius norm of the Hessian [7], and local linearity regularization [8], have all proposed methods to induce robustness by enforcing low curvature. In this paper, the authors focus specifically on the out-of-the-box robustness of LCNNs.

2.3 Gradient Instability

Previous studies, including Ghorbani et al. [4] and Zhang et al. [11], have demonstrated that gradient explanations in neural networks can be unreliable because, for any input, it is possible to find adversarial inputs with highly dissimilar gradient explanations. Ros and Doshi-Velez [9] empirically showed that gradient regularization could improve robustness. Dombrowski et al.[2] shows that gradient instability occurs due to large hessian norms. Therefore, gradient regularization can improve model robustness. Dombrowski et al.[3] proposed to train low curvature models via softplus activations and weight decay. Dombrowski et al. [3] focused on the Frobenius norm of the Hessian.

$$||A|| = \sqrt{AA^H}$$

while authors penalize the normalized curvature, which is a scaled version of the Hessian spectral norm.

2.4 Lipschitz Layers in Neural Networks

A function $f : R^M \to R^N$ is Lipschitz continuous if there is a constant L such that

$$\|f(x) - f(y)\| \leq L\|x - y\|$$

$$\frac{\|f(x) - f(y)\|}{\|x - y\|} \leq L$$

The smallest value of L is the Lipschitz constant of f and is denoted $Lip(f)$.

Ideally, if inputs are similar for a neural network (NN), outputs should also be similar, implying we want the Liptschitz constant L of the neural network to be small as possible. As the neural network is the composition of functions, we can use the property of the Liptschitz constant defined in theorem 0.1

Theorem 0.1 *Let $f = g \circ h$. If g and h are Lipschitz continuous, then f is also Lipschitz continuous with $Lip(f) \leq Lip(g)Lip(h)$.*

Therefore, if we make each component of a neural network such that it is Lipschitz continuous with small Lipschitz constants, the whole neural network will also be Lipschitz continuous with small Lipschitz constants. Activation functions and Pooling layers generally have Lipschitz constant $L = 1$ ex. ReLU activation. Using Spectral Normalization, where linear layers are re-parameterized by dividing by their spectral norm, ensures that the overall spectral norm of the parameterized layer is 1, which bounds Lipschitz constant L to be 1.

3 Methods and Approaches

According to the paper, we need the following components to train low-curvature neural networks(LCNNs).

3.1 Measuring model's Nonlinearity

Since we know higher the curvature C_f of the model it is far away from being linear. $\max_{x \in R^d} C_f(x)$ can be seen as a measure of a model's non-linearity. One of the common measures of curvature is the Hessian norm which depends on the scaling of gradients to deal with this issue authors proposed normalized curvature.

$$C_f(x) = \frac{\|\nabla^2 f(x)\|_2}{\|\nabla f(x)\|_2 + \epsilon}$$

Here, $\|\nabla^2 f(x)\|_2$ and $\|\nabla f(x)\|_2$ are spectral norm of Hessian and l_2 norm of gradient of function $f(x)$ respectively and $\epsilon > 0$. In order to have a network of low curvature, we can penalize the normalized curvature. Although, directly penalizing the curvature is computationally expensive as it requires two backpropagation steps in the network. On backpropagation estimating the Hessian norm requires backpropagating gradient-vector products. We require an efficient method penalization procedure that takes a single backpropagation step. If we can somehow control the curvature and Lipschitz constant of each layer of a neural network it will enable us to control the overall curvature of the model. In the next section, we will look at some methods to control the curvature of nonlinear activations and the Lipschitz constant of linear layers.

3.2 Centered Softplus Activation

As the curvature of a neural network depends on the curvature of its constituent activation functions authors propose to use activation functions with minimal curvature. Here author proposes to use Softplus activation which is a smoother version of ReLU. Softplus function is defined as

$$s(x; \beta) = \frac{log(1 + exp(\beta x))}{\beta}$$

The curvature of softplus activation depends on β. Using small values of β ensures low curvature. However, for small values of β softplus is diverging $s(x;\beta \longrightarrow 0) = \infty$. To deal with this issue, the authors proposed centered softplus activation, which adds normalization terms to softplus.

$$s_0(x;\beta) = s(x;\beta) - \frac{\log 2}{\beta} = \frac{1}{\beta}\log\left(\frac{1+exp(\beta x}{2}\right)$$

The authors further propose to cast β as a learnable parameter and penalize its value which directly penalizes that layer's curvature.

3.3 Spectrally Normalized Convolutions and Fully Connected Layers

To penalize the Lipschitz constant of linear layer and convolutions layers spectral normalization technique is used. For fully connected layers weights are normalized using simple spectral normalization, it is applied as

$$\frac{W}{\|W\|_2}$$

For convolution power iteration method that work directly in linear mapping implicit to convolution. Using normalization we ensure spectral norm of these layers is 1.

3.4 Parameterized Swish Activation

The reason behind choosing softplus activation in paper is due to the fact that it is continious and it resembles with ReLU activation. So different activation function similar to ReLU other than softplus can be taken. We choose Swish activation because of its continuity and the fact that with parameterized swish activation we can control its curvature.

$$s(x) = \frac{x}{1+e^{-x}}$$

$$s(x;\beta) = \frac{x}{1+e^{-\beta x}}$$

As we decrease value of β the curvature of swish activation decreases. For $\beta = 0$ it becomes a straight line. Also, it do not diverges like softplus for lower values of beta.

3.5 Work done before mid-term project review

We conducted a set of tasks, including reading and understanding relevant materials and references, as well as the code provided by the author, testing it for errors, and fixing some of those errors. Similar to the authors, we ran initial experiments using four ResNet18 models (two with LCNN and two without LCNN) but only for 50 epochs with curvature and gradient norm regularizers on the CIFAR 10 dataset. The performance of LCNN models was then compared with non-LCNN models.

3.6 Work done after mid-term project review

We analyzed the adversarial robustness of LCNN and non-LCNN networks. However, the models used in the author's code were Resnet 18, 34, and VGG, which took a lot of time to converge. Therefore, in the midterm review, only 50 epochs were shown, and attacking those networks was not helpful as the accuracy was low in LCNN than non-LCNN variants. Additionally, training for more epochs was not possible due to the resource constraint. To overcome this, I used a small CNN model with 2 convolutions, batch normalization, and 2 dense layers with dropout along with the MNIST dataset instead of CIFAR10. We combined parts of the author's code with new training and validation methods and attack methods in a Python notebook to achieve this. We also created a parameterized Swish activation function and made changes to the author's code to make it work. Finally, we analyzed the curvature and adversarial robustness of the proposed author's activation with the parameterized Swish.

4 Data set Details

The MNIST (Modified National Institute of Standards and Technology) is a large database of handwritten digits $\{0, 1, ...9\}$. This dataset contains $60,000$ training samples and $10,000$ test samples. This dataset is a subset of the larger set made available by NIST. Each image in the dataset has been size-normalized and centered in a fixed-size image. Each image is a grayscale image of size 28×28. This dataset is widely used for benchmarking in the field of machine learning.

5 Experiments

In this section we perform experiments to (1) evaluate performance of LCNNs (2) compare the adversarial robustness of LCNNs (3) evaluate the effectiveness of our proposed parameterized swish activation and its comparison with centered softplus. All experiments are done on machine with NVIDIA Geforce 940MX with 2GB VRAM and 12 GB RAM. For longer run Kaggle environment was used that provide NVIDIA Tesla P100 with 16gb VRAM and 13 GB of RAM. We choose simple classifier with 2 convolutional layer + maxpooling and 2 linear layers with dropout was used. All the above models are trained using Adam optimizer with initial learning rate of 10^{-4}

6 Results

Table 1 and Table 2 shows experiment to comapare adversarial robustness of LCNNs with non-LCNNs. From these tables we can see that LCNN are robust compared to non-LCNNs. Table 3 and Table 4 compare proposed performance of Swish activation with Softplus for high and low values of β. Non-LCNNs models although perform better with swish while LCNNs models perform better with softplus activation. Table 5 and Table 6 shows curvature of with proposed swish activation and softplus activation. We can see taking swish activation also decreases curvature of model but parameterized softplus outperforms swish by significant margin.

7 Future Work

LCNNs converge very slowly compared to non-LCNN variants for larger datasets a possible direction can to come up with a training procedure to speed up the process this can include the proposal of a new optimizer or training method.

Model	**Acc (%)**	$\|\epsilon\| = 0.05$	$\|\epsilon\| = 0.1$	$\|\epsilon\| = 0.15$	$\|\epsilon\| = 0.2$
Net	98.289	88.09	62.82	62.82	62.82
LCNN	97.5699	90.10	71.22	71.22	71.22
Net + GradReg	98.4899	89.18	64.18	64.18	64.18
LCNN + GradReg	97.3999	89.50	68.11	68.11	68.11

Table 1: PGD attack accuracy of models with different regularization techniques

Model	**Acc (%)**	$\|\epsilon\| = 0.05$	$\|\epsilon\| = 0.1$	$\|\epsilon\| = 0.15$	$\|\epsilon\| = 0.2$
Net	98.289	92.24	79.33	58.09	32.63
LCNN	97.5699	91.72	79.44	59.46	38.49
Net + GradReg	98.4899	92.82	78.69	55.61	31.82
LCNN + GradReg	97.3999	91.18	77.07	57.63	37.22

Table 2: FGSM attack accuracy of models with different regularization techniques

Model	**Val Acc**
Net	98.5699
LCNN	97.0799
Net + GradReg	98.3299
LCNN + GradReg	97.4399

(a) Validation Accuracy with Swish activation

Model	**Val Acc**
Net	98.289
LCNN	97.5699
Net + GradReg	98.489
LCNN + GradReg	97.3999

(b) Validation Accuracy with Softplus activation

Table 3: Comparing Validation accuracy Swish and Softplus for higher values of β

Model	**Val Acc**
Net	98.5699
LCNN	97.0799
Net + GradReg	98.20999
LCNN + GradReg	97.4399

(a) Validation Accuracy with Swish activation

Model	**Val Acc**
Net	98.1899
LCNN	97.5699
Net + GradReg	98.5599
LCNN + GradReg	97.3999

(b) Validation Accuracy with Softplus activation

Table 4: Comparing Validation accuracy with Swish and Softplus for lower values of β

Model	Val Acc
Net	21.71792
LCNN	3.29209
Net + GradReg	22.37583
LCNN + GradReg	3.33040

(a) Curvature with Swish activation

Model	Val Acc
Net	15.4774
LCNN	2.6635
Net + GradReg	15.2459
LCNN + GradReg	2.57437

(b) Curvature with Softplus activation

Table 5: Comparing Curvature with Swish and Softplus for higher values of β

Model	Val Acc
Net	7.9653
LCNN	3.29209
Net + GradReg	7.77148
LCNN + GradReg	3.33040

(a) Curvature with Swish activation

Model	Val Acc
Net	7.48276
LCNN	2.6635
Net + GradReg	7.4035
LCNN + GradReg	2.57437

(b) Curvature with Softplus activation

Table 6: Comparing Curvature with Swish and Softplus for lower values of β

Model	Acc (%)	$\|\epsilon\| = 0.05$	$\|\epsilon\| = 0.1$	$\|\epsilon\| = 0.15$	$\|\epsilon\| = 0.2$
LCNN (SP)	97.56999	90.10	71.22	71.22	71.22
LCNN (SW)	97.07999	86.58	60.92	60.92	60.92
LCNN + GradReg (SP)	97.3999	89.50	68.11	68.11	68.11
LCNN + GradReg (SW)	97.43999	88.10	64.03	64.03	64.03

Table 7: PGD attack accuracy of LCNNs with parameterized Swish and Softplus activation

Model	Acc (%)	$\|\epsilon\| = 0.05$	$\|\epsilon\| = 0.1$	$\|\epsilon\| = 0.15$	$\|\epsilon\| = 0.2$
LCNN (SP)	97.5699	91.72	79.44	59.46	38.49
LCNN (SW)	97.07999	89.21	71.37	47.80	22.96
LCNN + GradReg (SP)	97.3999	91.18	77.07	57.63	37.22
LCNN + GradReg (SW)	97.43999	90.70	75.29	54.02	34.20

Table 8: FGSM attack accuracy of LCNNs with parameterized Swish and Softplus activation

8 Conclusion

LCNNs with softplus activation functions demonstrate lower curvature and improved robustness against adversarial attacks compared to non-LCNNs. Furthermore, it was observed that the LCNNs with softplus outperformed the proposed method with the Swish activation function, particularly in terms of adversarial robustness. While the results of the curvature comparison between the proposed activation function and the softplus activation function were similar, there was a significant difference in terms of their ability to withstand adversarial attacks. These findings suggest that the use of low-curvature activation functions, using softplus activation function, may be an effective approach to improve the robustness of neural networks against adversarial attacks.

References

[1] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1310–1320. PMLR, 09–15 Jun 2019.

[2] Ann-Kathrin Dombrowski, Maximilian Alber, Christopher J. Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. Explanations can be manipulated and geometry is to blame, 2019.

[3] Ann-Kathrin Dombrowski, Christopher J. Anders, Klaus-Robert Müller, and Pan Kessel. Towards robust explanations for deep neural networks. *Pattern Recognition*, 121:108194, 2022.

[4] Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3681–3688, Jul. 2019.

[5] Matthias Hein and Maksym Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[6] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[7] S. Moosavi-Dezfooli, A. Fawzi, J. Uesato, and P. Frossard. Robustness via curvature regularization, and vice versa. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9070–9078, Los Alamitos, CA, USA, jun 2019. IEEE Computer Society.

[8] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Krishnamurthy Dvijotham, Alhussein Fawzi, Soham De, Robert Stanforth, and Pushmeet Kohli. Adversarial robustness through local linearization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[9] Andrew Slavin Ros and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018.

[10] Suraj Srinivas, Kyle Matoba, Himabindu Lakkaraju, and François Fleuret. Efficient training of low-curvature neural networks. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[11] Xinyang Zhang, Ningfei Wang, Hua Shen, Shouling Ji, Xiapu Luo, and Ting Wang. Interpretable deep learning under fire. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1659–1676. USENIX Association, August 2020.