Exploring Convexity in Neural Networks

Gurkirat Singh
Center for Visual Information Technology, IIIT, Hyderabad
gurkirat.singh@students.iiit.ac.in

ABSTRACT

Neural Networks have started to over fit everything. Many tricks are used to prevent then from overfitting but none of which works for everything, a lot of parameter tuning is involved. Because of which any NN (specially Deep NNs) lack on generalization. We aim to tackle it by making tweaking the basics of a Neural Network. As shown in [3], convexity helps the neural networks to generalize in a lot of architectures and we achieve better results than non-convex counterpart. Here we try the same concept on SOTA models (MlpMixer [5] and CycleMLP [1]) and see how convexity behaves for these models on ImageNet 2012 dataset [2]

KEYWORDS

neural networks, generalizations, convex functions, overfitting

1 INTRODUCTION

Neural Networks have become a go-to for any task. They perform exceptionally well as compared to any classical ML models. They are able to learn complex non-linear functions and hence perform good if given enough data and time to train. Deep Neural Networks as being able to learn complex boundaries, easily overfit the training set. This creates a lot of issues causing Neural Network to not generalize and hence performing very badly on unseen data.

As shown in [3], this is a very big problem. If we randomly shuffle the labels in the training dataset and let the model train for some time, we achieve close to 100% accuracy on training dataset, indicating that the Neural Network has "memorized" the dataset instead understanding the underlying pattern and classifying on the basis of that.

To tackle this [3] proposed to convexify the function that Neural Network creates. This helps in giving smoother boundaries and hence better generalization.

2 LITERATURE REVIEW

I follow Input-Output Convex net proposed in [3] as it gives the best of the results. They constraint the neural network to learn only convex functions of the output. This generalization is achieved by using convex activation functions and only using only positive weights.

3 INPUT OUTPUT CONVEX NEURAL NETWORKS

Here a brief explaination of IOC-NN has been given. Please refer [3] for more in-depth understanding.

Supervised by Sarath S Sivaprasad and Vineet Gandhi.

Given a simple MLP with k layers, the output of i^{th} neuron in l^{th} layer is given as:

$$h_i^{(l)} = \phi(\sum_j w_{ij} h_j^{(l-1)} + b_i^{(l)})$$

where,
for first layer $h_j^{(0)} = x_j (j = 0..d)$, the input layer
for last layer $h_j^{(k+1)} = y_j$, the output layer

to ensure y_js to be a convex function of x_js, only 2 conditions need to be satisfied

- ϕ should be a convex function and non-decreasing functions
- $w^{(2:k+1)} >= 0$

The proof follows for properties of the operator as

- $f(g(x)$ is convex if g is convex and f is a convex and non-decreasing function
- Non-negative sum of convex functions is also a convex function.

Also, as the first layer is just a affine transformation of the inputs hence, $w^(1)$ can be negative without comprimising on convex constraint

4 MODELS

4.1 Classic Models

I first recreate the results achieved in [3]. I re-implemented the 3 layer MLP network and trained it on CIFAR10 to compare the results

4.1.1 MLP Network. I created MLP network with 3 hidden layers with 800 nodes in each hidden layer. Batch Normalisation is used after each layer and ReLU is used as an activation layer. I also created the IOC counterpart, say IOC-MLP. Everything else remains the same only the activation layer was swapped with ELU, a convex activation layer. After each train step I added an extra step to ensure $w >= 0$ for all layers after the first hidden layer

4.2 MLP Mixer

MLP Mixer [5] is SOTA model develop by Google Research which is a pure MLP model for image classification. It is one of the first works which show that MLP networks can also be used in image classification tasks as well. MLP Mixer breaks the image into tokens and projected into a hidden dimension C. And uses an mlp across tokens for inter-token information flow and an mlp across C channels in each token for inter-channel information flow. More details about the model architecture can be found in [5].

To create the convex counter-part (IOC-MLPMixer), I changed the GeLU activation layer to convex ELU convex layer and also added a extra step after each train step to ensure $w >= 0$ for all layers after first token mixing layer in the first mlp mixer block.

Table 1: MLP Mixer on CIFAR 10

Model	Train Accuracy	Test Accuracy
MLP Mixer	85.72	71.1
IOC-MLP Mixer	78.57	67.97

4.3 CycleMLP

In cycle MLP we use MLP which has a larger receptive field than Channel FC / Token FC as in MLP Mixer and has a single Cycle FC layer which expands over both channels and tokens in the image. It also only has linear complexity and is independent of image size therefore can be used for segmentation tasks as well. More details about the model architecture can be found in [1].

5 EXPERIMENTS AND RESULTS

I used various techniques to train each model, the process and results are given below for each model

5.1 Classical Model: MLP

This model was for a POC. I used Adam optmizer with initial learning rate of 0.00001. Results are as expected and generalization is seen (as shown in results). I did not run this for many epochs as this was only to test the implementation of the code.

I also tested for other architectures in [3] and the datasets, the results of which can be visualized here. As I did not run for many epochs, the results though being on the line did not get to expected numbers

5.2 MLP Mixer

I used only 4 mixer layers (smaller than Mixer-S) with hidden size of 128, patch size of 9x9, resized the image to 72x72 image, Token mixing mlp with 64 neurons and channel mixing mlp with 128 neurons. Trained for I trained both convex and ioc-counterpart MLP Mixer with Adam optimizer and with learning rate of 0.001 on CIFAR10 dataset for first testing if generalization holds in MLP Mixer model, before testing on a large scale dataset (such as ImageNet). I used only 4 mixer layers (smaller than Mixer-S) with hidden size of 128, patch size of 9x9, resized the image to 72x72 image, Token mixing mlp with 64 neurons and channel mixing mlp with 128 neurons. Trained for 100 epochs Results are given in table 1. For loss / accuracy graphs please click here As we can see the generalization holds for CIFAR 10 and we get decent results to SOTA method.

For ImageNet, I tried training MLP Mixer, but the due to **huge** size of the dataset and the model (even the smaller one with only 4 layers) was taking approximately 4 hours for one epoch! I put it for 3 days but the results were not comming out promising due to less number of layers in MLP Mixer. I tried a lot of ways and parameter, reducing batch size, reducing the layers. But none of them gave a conclusive result Therefore after talking to Sarath I dropped this model.

5.3 CycleMLP

I followed a similar pattern with Cycle MLP to first test on a smaller dataset like CIFAR10. I tested CycleMLP with both CIFAR10 and CIFAR100. I used out of the box Cycle-B1 model for this experiment.

Table 2: CycleMLP on CIFAR 10

Model	Train Accuracy	Train Loss	Test Accuracy	Test Loss
CycleMLP	99.71	0.51	82.87	0.69
IOC-CycleMLP	81.42	0.92	72.26	0.84

Table 3: CycleMLP on CIFAR 100

Model	Train Accuracy	Train Loss	Test Accuracy	Test Loss
CycleMLP	99.69	0.806	51.98	2.362
IOC-CycleMLP	79.98	1.574	39.56	2.516

I also used augmentation and hyper parameters suggested by the authors. I trained for 300 epochs.

The results can be found in table 2 and 3

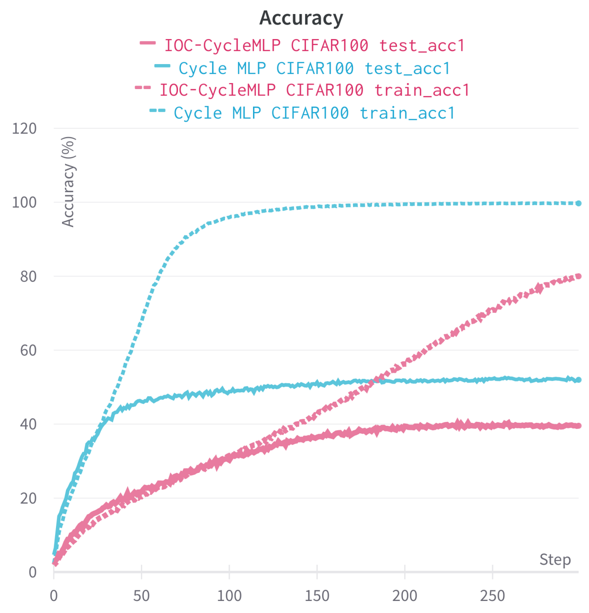

Figure 1: CycleMLP and IOC-Cycle MLP accuracy on CIFAR 100

As we can see from the results, the generalization is not holding in CycleMLP. There is a 10% drop from non-IOC counter part in test results. Even generalization breaks in CIFAR100 as we see a gap of 40% between train and test results. hence we drop this model and don't test on ImageNet

5.4 Densenet

Convexity worked well as verified in [3] in densenets, therefore I tried to train it on Imagenet directly. training on image net took a very long time. There results on normal Densenet were as expected close to 75%. But the IOC-Densenet was way below the expected result. I was able to achieve 10% on test set which is way below the non-Convex part.

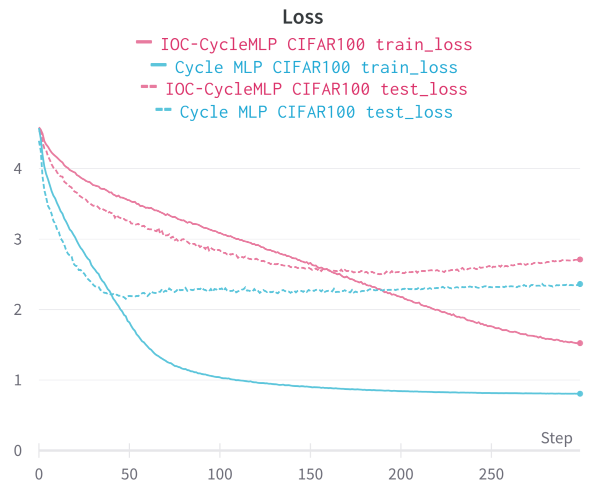

Figure 2: CycleMLP and IOC-Cycle MLP loss on CIFAR 100

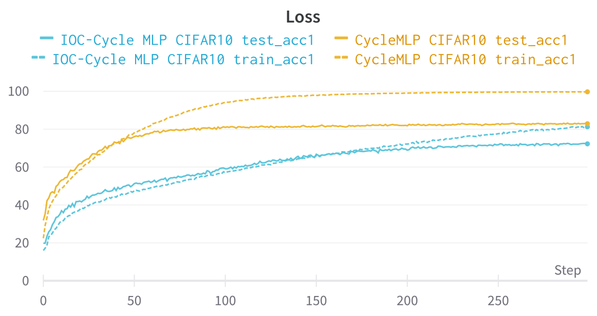

Figure 3: CycleMLP and IOC-Cycle MLP accuracy on CIFAR 10

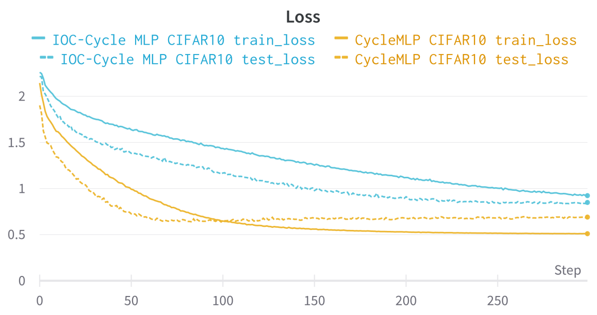

Figure 4: CycleMLP and IOC-Cycle MLP loss on CIFAR 10

6 FURTHER WORK AND CONCLUSION

As we can see, Convex Neural Networks doesn't seem to perform well on large scale datasets. However my next aim is to try convex counter part of efficient net [4] which has a lower number of parameters with competitive results with other state-of-the-art models. Also if high compute is available, I would like to try MLP Mixer on image net which I think should work as it performs good on CIFAR10.

I would also like to try a counter part of neural networks where we enforce convexity only on the FC part of CNNs as the convolutional part is just trying to model the input into a latent feature space, so if the output is a convex function of this feature space, and hence a generalization should follow.

REFERENCES

[1] Shoufa Chen, Enze Xie, Chongjian Ge, Runjian Chen, Ding Liang, and Ping Luo. 2021. CycleMLP: A MLP-like Architecture for Dense Prediction. https://doi.org/10.48550/ARXIV.2107.10224
[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848
[3] Vineet Gandhi Sarath Sivaprasad, Naresh Manwani. 2021. The Curious Case of Convex Networks. In *ECML*.
[4] Mingxing Tan and Quoc V. Le. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. (2019). https://doi.org/10.48550/ARXIV.1905.11946
[5] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. 2021. MLP-Mixer: An all-MLP Architecture for Vision. https://doi.org/10.48550/ARXIV.2105.01601