Analysis of Parameter Dynamics in Image Classification Models

Simran Tiwari (st3400), Saravanan Govindarajan (sg3896)

Executive Summary

- Problem statement :
  - Understand CNNs by studying evolution of parameters and performances (loss, accuracy) of image classification models during and after training for various initialization techniques.
  - Study the feature relationships that the networks extracts for performing image classification.

- Overall in this project we try to run different experiments to answer the following questions
  - What percentage of the initialized parameters change after the training? Shallow (LeNet) vs deep networks (VGG, ResNet) - do they have different characteristics in this context? Does the weight initializer play a role in this case?
  - How distant (norm) are the trained parameters from initialized parameters? Does the learning optimizer affect this characteristic?
  - If we take a pretrained model and flip the sign of its weights, does the model again converge back to the same weights?
  - If we take a trained model and add a small distortion to the weights, does the model converge back to the same weights?
  - Are learnt features independent of each other or is there any correlation? Does choice of optimizer have any impact on it?

Problem Motivation

- Despite the astonishing success of image classification models, the mechanism by which their inner layers act has not been understood completely yet.

- Treating neural network as a successful black-box for computationally intensive problems prevents us to extend their applications to sensitive areas such as medical diagnoses and self driving cars.

Background work

- Degradation problem - Deep Residual Learning for Image Recognition
- Lazy Training - On Lazy Training in Differentiable Programming
- Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification

Technical Challenges

- Computational resources
  - We trained 126 models with millions parameters for our experiments
  - This required a lot of time to train
  - As well as a lot of computational resources
  - To help us overcome this challenge we decided to use T4 GPU

- Analysis of results
  - Training such a large number of models on multiple datasets gave us a lot of data to evaluate
  - We obtained over plots and correctly understanding and analyzing them was a challenging task

Technical Challenges

- Designing Experiments
  - Understanding the internal dynamics of CNNs is a pretty open ended problem

  - Another challenge was designing the experiments to help us solve this problem

  - We decided to solve this problem by performing 3 experiments:
    - Part 1 - Analyse weight movements on different initialization methods
    - Part 2 - Relationship between features
    - Part 3 - Effect of small weight distortions on performance

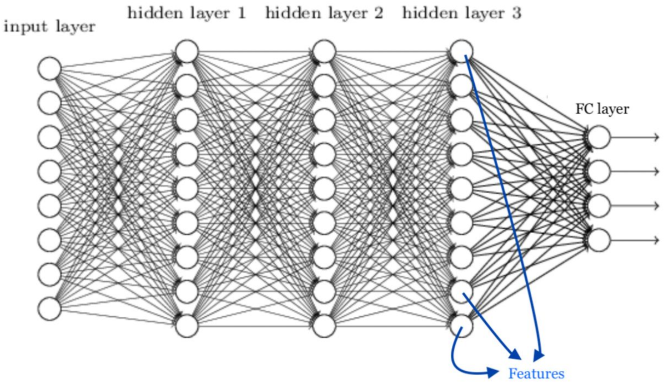
Approach

Our study contains 3 main parts:

- **Part 1: Analyse weight movements on different initialization methods**
  - Norm distances between the weights of the model between every five epochs to see the weight movements.
  - If θ_i denotes the weights at epoch 'i', and if there are 20 epochs, then we would be computing $||\theta_5 - \theta_0||$, $||\theta_{10} - \theta_5||$, $||\theta_{15} - \theta_{10}||$, $||\theta_{20} - \theta_{15}||$.

Approach

- **Part 2: Relationship between features**
  - Correlations between features (θi, θj) where $i \neq j$ in the trained feature layer - second last layer of the CNN will be computed and compared.
  - Similarly, pairwise correlations between feature layers from different models was also compared.
  - This is to evaluate if the models learn the same features irrespective of the weight initialization, learning optimizer etc.
  - Correlation can also give us an sense if the models with different initializations learn features in the same order .



Approach

- **Part 3: Effect of small weight distortions on performance**
    - We will perform weight perturbations such as sign flipping, adding small noise etc. on the learnt parameters, and retrain the network with the perturbed weights. We evaluate if the new weights converge to the original weight values.

Solution Diagram/ Architecture

3 Networks

LeNet
ResNet
VGG19

X

Name	Image Size	Description
CIFAR10	3 * 32 * 32	Objects including airplane, automobile, ..., truck.
MNIST	1 * 28 * 28	Handwritten digits
Fashion MNIST	1 * 28 * 28	Clothing including T-shirt/top, Trouser, ..., Ankle boot.

X

3 Initializations (Initial weights $\boldsymbol{\theta_0}$)

kaiming-uniform
kaiming-normal
pretrained model

X

2 Optimizers

SGD
ADAM

54 Trained Models (Trained Parameters - $\boldsymbol{\theta}$)

Trained Models (54) (Trained Parameters - $\boldsymbol{\theta}$)

Weight Disturbances & Retraining (2)

Flipped Sign $\theta \mapsto -\theta$
Distortion $\theta \mapsto \theta + 10^{-6}$

After retraining, 54 x 2 = 108 models

162 Trained Models

-

Special cases

~

126 models

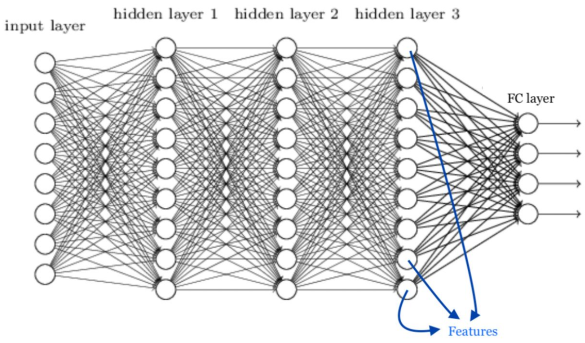
Implementation Details

- Colab Pro with GPU was used for training the models
- Pytorch framework was used to train LeNet, Resnet18, and VGG19, which have 3246, 11×10^6, 138×10^6 trainable parameters with 5, 18, and 19 layers, respectively
- Datasets used: CIFAR10, MNIST, Fashion MNIST

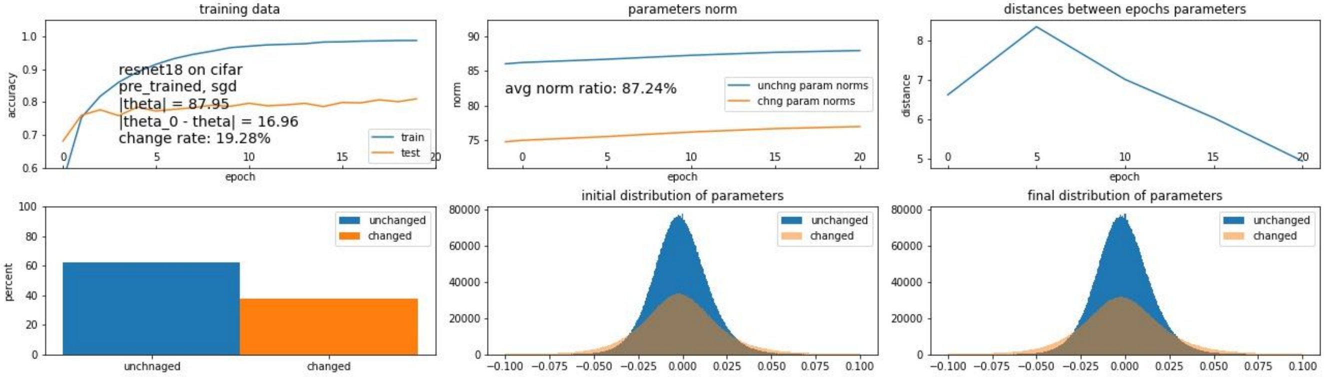
Name	Image size	Description
CIFAR10	$3 * 32 * 32$	Objects including airplane, automobile, ..., truck.
MNIST	$1 * 28 * 28$	Handwritten digits $(0 - 9)$
Fashion MNIST	$1 * 28 * 28$	Clothing including T-shirt/top, Trouser, ..., Ankle boot.

Demo/Experiment design flow

- As mentioned in the previous slides, we will be evaluating the following to gather some insights:
  - Percentage of the initialized parameters that change after training
  - Comparison of Norm of the parameters at every 5 epochs
  - Correlation study on the feature vectors
    - Pairwise correlation between the features of the trained model

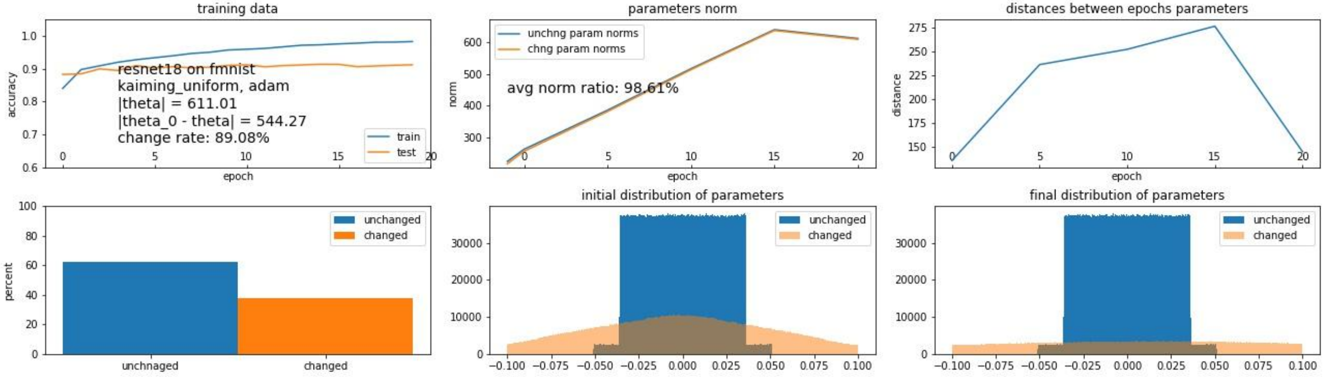


Experimental Evaluation - Dynamics of Parameters

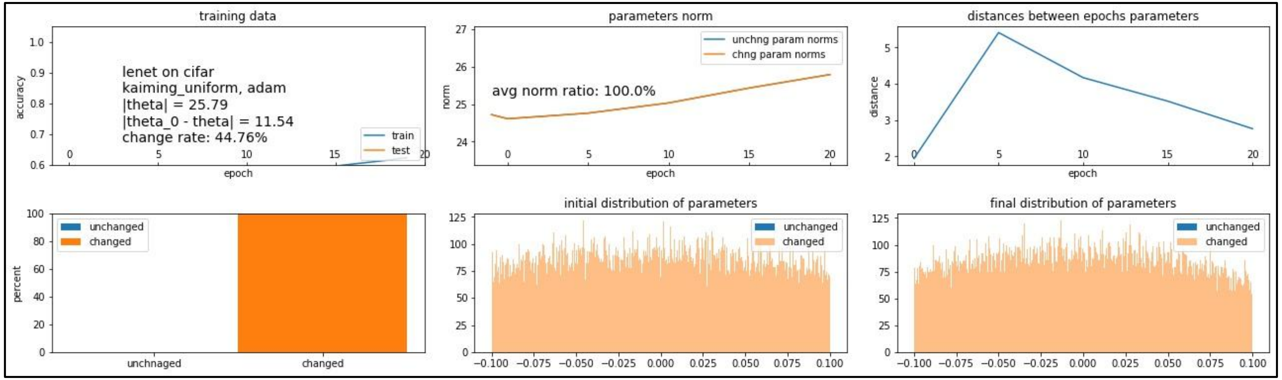


Resnet tends to keep over 60% parameters unchanged regardless of

- dataset
- optimizer
- initial point

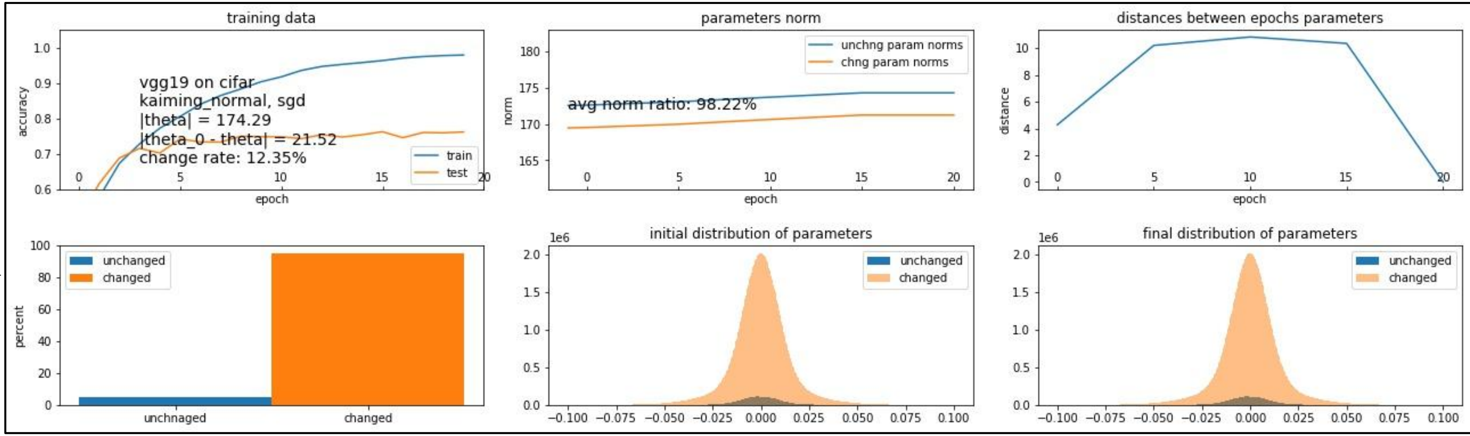


Experimental Evaluation - Dynamics of Parameters



Lenet on Cifar:

100% Parameters changed

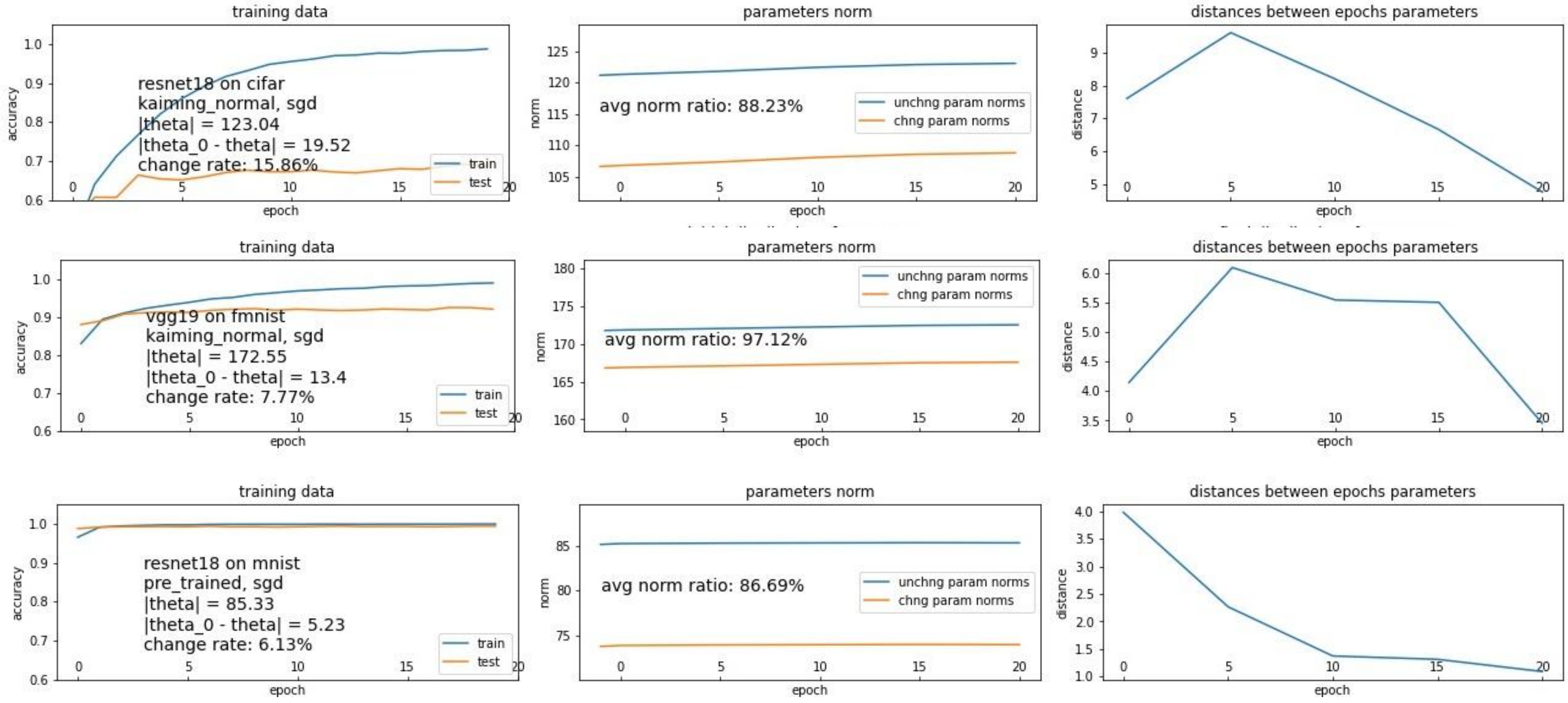


VGG on Cifar:

>90% Parameters changed

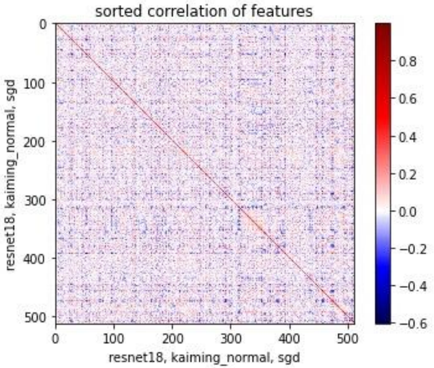
SGD finds a close local minima for huge networks

Experimental Evaluation - Dynamics of Parameters

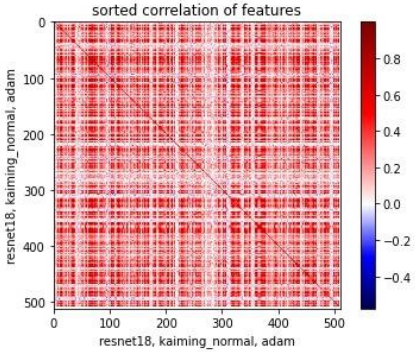


Experimental Evaluation - Correlation Analysis

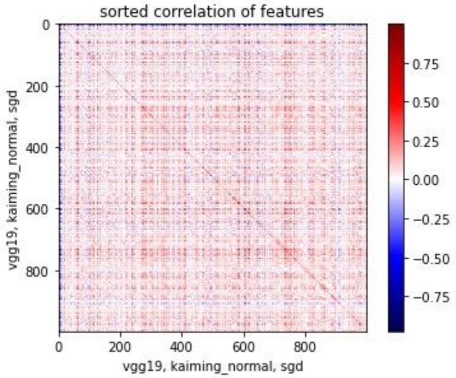
Stronger correlation once trained with ADAM



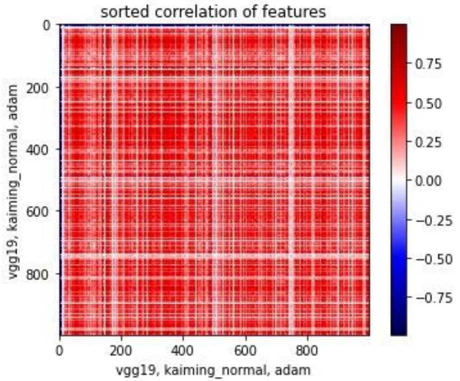
data: mnist
|theta_1| = 120.32
|theta_2| = 120.32
|theta_1 - thetha_2| = 0.0



data: mnist
|theta_1| = 209.18
|theta_2| = 209.18
|theta_1 - thetha_2| = 0.0

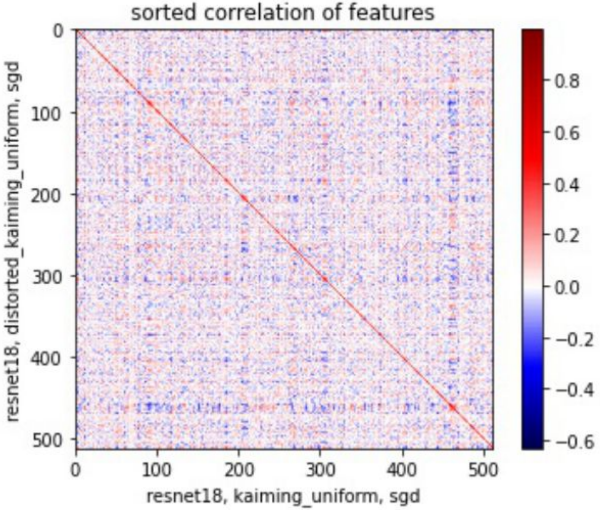


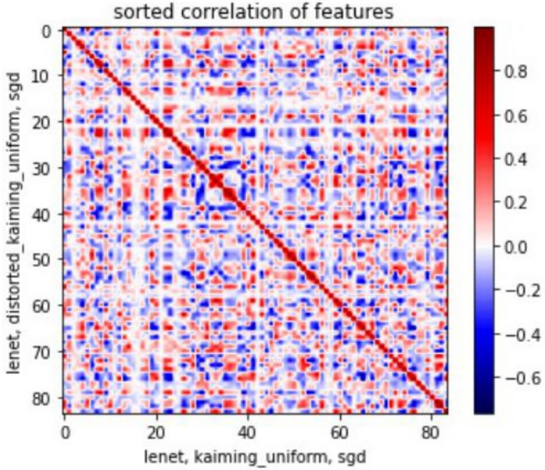
data: cifar
|theta_1| = 174.29
|theta_2| = 174.29
|theta_1 - thetha_2| = 0.0

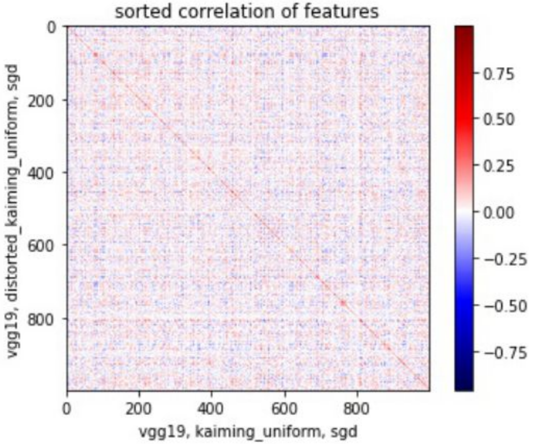


data: cifar
|theta_1| = 175.84
|theta_2| = 175.84
|theta_1 - thetha_2| = 0.0

Experimental Evaluation - Distortion Effect





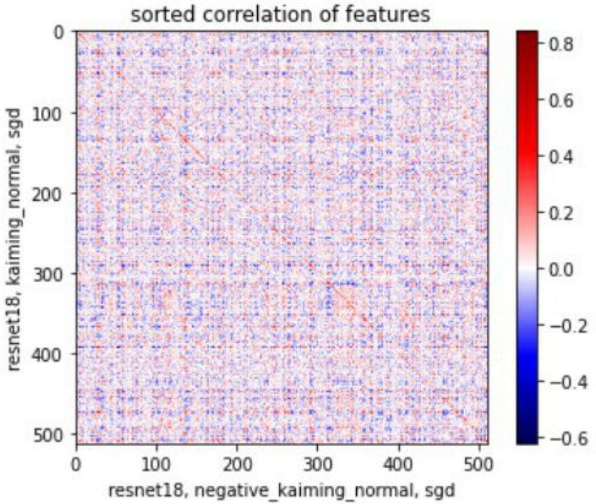


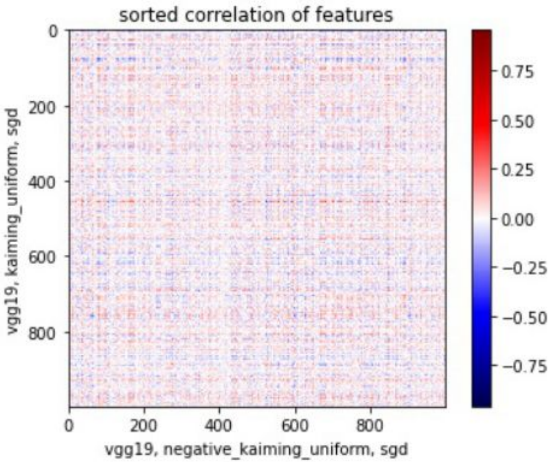
data: mnist
|theta_1| = 120.3
|theta_2| = 120.29
|theta_1 - thetha_2| = 0.27

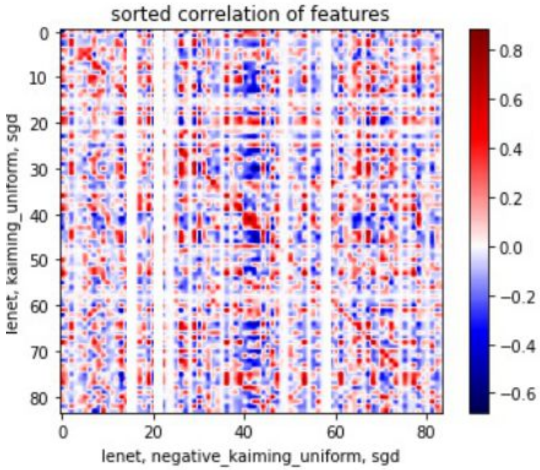
data: mnist
|theta_1| = 23.43
|theta_2| = 23.02
|theta_1 - thetha_2| = 1.3

data: mnist
|theta_1| = 171.79
|theta_2| = 171.78
|theta_1 - thetha_2| = 1.38

Experimental Evaluation - Flipping Effect







data: mnist
|theta_1| = 120.32
|theta_2| = 120.44
|theta_1 - thetha_2| = 240.67

data: mnist
|theta_1| = 171.78
|theta_2| = 180.41
|theta_1 - thetha_2| = 347.66

data: fmnist
|theta_1| = 24.42
|theta_2| = 26.06
|theta_1 - thetha_2| = 49.6

Conclusion

- Huge architectures trained by ADAM optimizer, the extracted features have stronger positive correlations among each other as compared to trained model with SGD. This didn't happen for small network Lenet.
- 60% of the parameters in ResNet tends to remain unchanged as a result of lazy training
- Small distortion doesn't usually change the model and features too much. They more or less come back to the same initial model (distance of vector of parameters and very high correction of features)
- Flipping the sign of a trained parameter and retraining it forced the network to find a local minima far away from a trained one.

References

- Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In International Conference on Machine Learning, pages 531–540. PMLR, 2018.
- Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. arXiv preprint arXiv:1812.07956, 2018.
- Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In International Conference on Machine Learning, pages 1019–1028. PMLR, 2017.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE international conference on computer vision, pages 1026–1034, 2015.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.
- Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. arXiv preprint arXiv:1609.04836, 2016.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25:1097–1105, 2012.
- Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In Proceedings of the IEEE, pages 2278–2324, 1998.
- Michael A Nielsen. Neural networks and deep learning, volume 25. Determination press San Francisco, CA, 2015.
- Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE international conference on computer vision, pages 618–626, 2017.