MATH 6380O Final Project: Reproducible Study of Training and Generalization Performance

Ng Yui Hong[1] yhngap@connect.ust.hk
[1]: Department of Mathematics, HKUST

1. Introduction

This poster aims at reproducing serval interesting and significant paper on the topic of generalization, over-parametrization, overfitting, randomization test, etc... We will try to explain the reason and intuition behind the results too. We hope we cam get some insight on various unanswered questions: Why do the deep learning models, despite being highly over-parameterized, still predict well?

1.1 Dataset

The Cifar10 dataset consists of 60,000 color images of size 32x32x3 in 10 classes.
We center-cropped our image into 28*28 dimensions. We standardize our image by removing the mean and then dividing the standard deviation for each image independently. For random pixel, shuffle pixel, random pixel, random label, we created a dataset object to implement all these operations.

2.1 Capacity of neural network (experiment setup)

We run experiments with the following modifications of the labels and input images.
1. True label 2. Random labels 3. Shuffled Pixels(same/random permutation)
5. Gaussian (Gaussian distribution is used to generate random pixels for images)
6. Mosaic (Replace training image by mosaic image generated by Gaussian distribution)

2.2 Capacity of neural network (result)

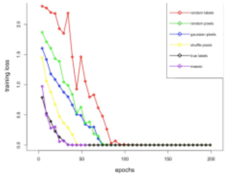

Figure 1: Training loss of various settings

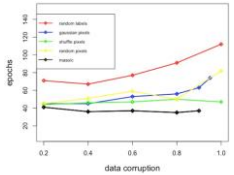

Figure 2: Relative convergence epoch of different corruption ratio

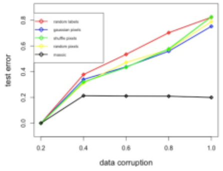

Figure 3: Test error under different data corruption ratio

Training loss: It is surprising that data with random labels will converge to zero training accuracy. As there is no relationship between the instances and the class labels, learning is impossible. Intuition suggests that this impossibility should manifest itself clearly during training, e.g. by training not converging or slowing down substantially

⇒ **Effective capacity of neural networks is sufficient for memorize the entire dataset.**

Random label has 'Fast' learning rate: large predictions errors are back-propagated to make large gradients for parameter updates as labels and samples are uncorrelated. since the random labels are fixed and consistent across epochs, the network starts fitting after going through the training set multiple times.

Random pixels and Gaussian converging faster: Random pixels and Gaussian converging faster than "random labels". This might be because with random pixels, the inputs are more separated from each other than natural images that originally belong to the same category, therefore, it is easier to build a network for arbitrary label assignments.

Partially corrupted labels: The networks fit the corrupted training set perfectly for all the cases. Since the training errors are always zero, the test errors are the same as generalization errors. As the noise level approaches 1, the generalization errors converge close to the performance of random guessing on CIFAR10.

Mosaic have great performance: As shown in Figure 2, 3, mosaic dataset behaves better than other noisy dataset. For example, increase in the corruption ratio does not lead to great change in convergence time and test error. It seems that deep neural network can automatically 'learn' from those correct images and ignore those mosaic images. On the other hand, deep learning is not that robust to massive noisy labeled dataset.

⇒ **Neural network can learn effectively even if there are 'moderate' noises in dataset**

3.1 Non-overfitting of test error and overfitting of test loss (model)

We follow [1] and adopt an all-convolution architecture. Specifically, we first put together five convolutional layers, then connect the last one to a fully connected layer which has ten outputs and take them as the model output. For the input channel of the first layer and the output channel of the last one, we fix them as three. The kernel is applied with stride 1 and without any padding.

To vary the number of parameters, we change the number of channels, n, and the size, s, of kernels used in convolution. To simplify the setting, we make all the internal channels among those convolutional layers have the same number. As each channel of each layer has its own kernel, the number of kernel parameters is $(3n^2 + 6n)s$^2. The fully connected layer has input of size $[32 - 5(s - 1)]$^2 because the image size is 32×32 and each convolutional layer will reduce the size by $s - 1$. $10 \cdot (37 - 5s)^2$. Henceforth, we control the number of parameters through n and s

Owing to overly long training time, we only use 1/10 of CIFAR-10 dataset, i.e., 5000 images, as training samples in our experiment. The learning rate is set to 0.01 and no exponential decay or other acceleration technique is used. For each model, we train it with 500 epochs and record the loss value and error rate. The error rate is obtained with tests on 1000 images, 1/10 of the original CIFAR-10 test set.

3.2 Non-overfitting of test error and overfitting of test loss (result)

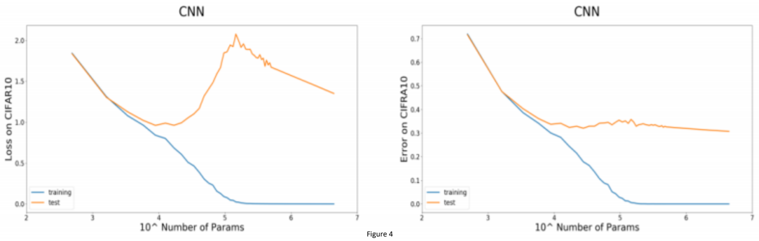

Figure 4

Phenomenon: As CNN complexity increases, test loss exhibits overfitting, while test accuracy remains steady. Possible explanation: It was expected that loss would exhibit overfitting, the question is why accuracy does not. Figure 5 shows the proportion of examples which are consistently classified correctly or incorrectly. From the graphs, one can see that around 60% of the test examples are consistently classified correctly, whereas 20% are consistently classified wrongly. A possible reason is that 60% of the test examples might be similar to the training ones, thus are easier for the model to give the right classifications.

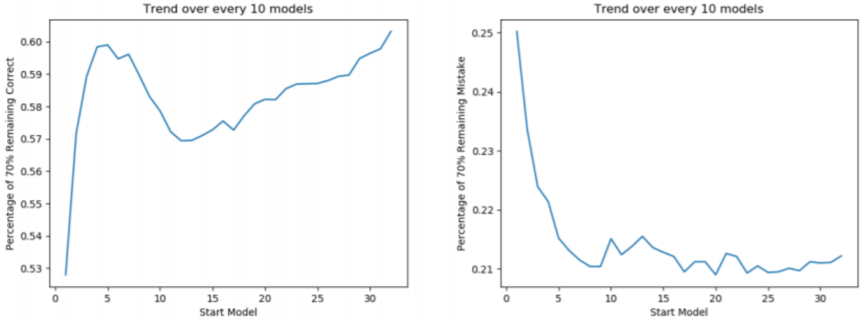

Figure 5: The proportion of examples that are classified correctly (left) or wrongly (right) 7 times out of 10 consecutive models of increasing complexity.

4. Mystery of good generalization performance of overparameterized models

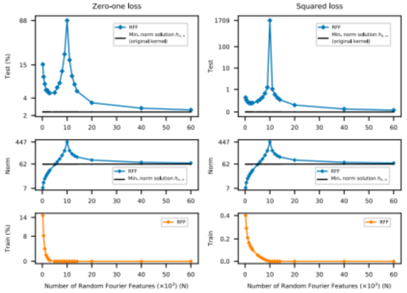

Figure 6: Figure copied from Mikhail Belkin's paper

The mystery can be explained by the left graph copied from Mikhail Belkin's paper and Rademacher complexity.
The network is trained on subset of MNIST, a dataset with around 10000 data. That's why on the top 'test' graphs, test% reach the peak at 10000 where $parameters = no. of\ data\ pts.$ Just to add, the U shape on the left of peak are the shift from underfit to overfit.
Key Fact: norm of solution decrease after the peak, when the network is overparameterized.

Modern architecture tends to find a low norm solution. When the network is overparameterized, what might do with the correct inductive bias is to find a low norm solution. Low norm solution have a lower complexity. With reference to Rademacher complexity, when overparameterized, training error becomes zero, complexity is low, as a result, the test error is small.

Reference:

1. Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. Nov 10, 2016.
2. Tomaso Poggio, K. Kawaguchi, Q. Liao, B. Miranda, L. Rosasco, X. Biox, J. Hidary, and H. Mhaskar. Theory of Deep Learning III: the non-overfitting puzzle. Jan 30, 2018.
3. Mikhail Belkin, Daniel Hsu, Siyuan Ma, Soumik Mandal. Reconciling modern machine learning practice and the bias-variance trade-off. PNAS , 2019, 116 (32).