Model Design for Deep Learning in Practice

Beibei Feng (bf2465)
Department of Statistics
Columbia University
`bf2465@columbia.edu`

Dongrui Han (dh2993)
Department of Statistics
Columbia University
`dh2993@columbia.edu`

Xinwen Miao (xm2242)
Department of Statistics
Columbia University
`xm2242@columbia.edu`

Zixuan Chu (zc2480)
Department of Statistics
Columbia University
`zc2480@columbia.edu`

Abstract

In this report, neural network models with different design choices related to network architecture, optimization, initialization and regularization were tested on MNIST dataset and fashion-MNIST dataset separately. Comparisons between the performance of each design choice on these two datasets were conducted in order to generate a realization with respected to the effect of different hyperparameter design choices on machine learning models. Based on the outcome from these two specific datasets, same design choices of hyperparameters from these four segments all resulted in different performance ranking, except for the initialization hyperparameters. With this being said, a further investigation on more datasets should be conducted in relation to deciding the potential best design choices.

1 Introduction

1.1 Problem Description

For this project, the main goal is to understand the influences resulting from different design choices for deep learning models. Then, further reaching some implications in terms of generalization and robustness (or lack thereof) of these models relative to hyperparameter choices. In details, different hyperparameter choices related to network architecture, optimization, initialization and regularization are trained on a neural network model. The best performed model is chosen based on the criteria, out-of-sample accuracy. Afterwards, the same procedure is conducted on a new dataset in order to reach a more general implications in the light of model generalization and robustness involved in hyperparameter options.

1.2 Literature Review

To tackle this problem, our team start by reviewing papers relavent to this topic, namely classification on MNIST dataset using neural network. In the paper *Assessing Four Neural Networks on Handwritten Digit Recognition Dataset (MNIST)* [1], models such as CNN, ResNet, DenseNet and CNN with CapsNet were trained on the MNIST dataset. With these models, the authors achieved a 99.75% accuracy rate. Additionally, in the journal *Comparisons of Deep Learning Algorithms for MNIST in Real-Time Environment*[2], the authors used four models on the basis of unlike

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

algorithms which are capsule network, deep residual learning model, convolutional neural network and multinomial logistic regression. They also give comparisons of train and evaluation time, memory usage and other essential indexes of all four models.

Based on the understanding on above papers, we proposed a rough conjecture that how the hyperparameters controlling network architecture, optimization, initialization and regularization are set inside the neural network model might play an important role in obtaining a higher out-of-sample accuracy.

1.3 Data Description

For this project, the MNIST and fashion-MNIST datasets are chosen to serve for the objective. MNIST dataset is a dataset of 60,000 small square 28×28 pixel grayscale images of handwritten single digits between 0 and 9. It contains 60,000 training images and 10,000 testing images. Fashion-MNIST is a dataset of Zalando's article images also consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes, namely clothes.

To train and test the proposed models, the MNIST dataset are split into training set and testing set. For each proposed model, it is trained on the training set. Then, the out-of-sample accuracy is set to be the criteria to determine the performance of the trained model. Finally, the best performance model choosing from the MNIST dataset will be tested on the new dataset, fashion-MNIST dataset. Also, the data are scaled by dividing 255.0 to make sure that all X values are between 0 and 1.

2 Method

2.1 Baseline Model

To start, the baseline model is trained as an unregularized MLP with 2 hidden layers of 16 hidden units in each layer, and ReLU activations on the MNIST hand-written digits dataset. Specifically, this baseline model is trained for 100 epochs using the Adam optimizer with a mini-batch size of 128. As a base model, optimization and regularization hyperparameters are set to be the default values from keras. Specifically, the learning_rate is set to be 0.001, beta1 is 0.9, beta2 is 0.999 and epsilon is 1e-08. The default hyperparameter value for initialization is glorot uniform for kernel and zeros for bias. With this baseline model, the out-of-sample accuracy on the MNIST test data is reported to be 0.9521, and the out-of-sample accuracy on the fashion-MNIST dataset is 0.8605

2.2 Hyperparameter Design

Four different hyperparameters related to neural network model are tested separately in order to grasp an understanding on how the design choices affect the performance of a neural network model. Specifically, they are **network architecture**, **optimization**, **initialization** and **regularization**. For each of the hyperparameter, three different design choices are tested on the MNIST dataset first, thus obtain an understanding on the performance of these different design choices. Upon this MNIST dataset, the same design choices are used again to train in the fashion-MNIST dataset in order to see whether it follows the same performance ranking as the previous dataset, namely MNIST dataset.

For **network architecture**, 1 hidden layer and 26 nodes, 2 hidden layers and 26 nodes, as well as 3 hidden layers and 26 nodes are selected to be the three deign choices. In the following content, models with these three design choices are named as model 1, model 2 and model 3. For **Optimization**,Adam with learning rate 0.0001, batch size 64 and 60 epochs, Adam with learning rate 0.0001, batch size 32 and 100 epochs, Adam with learning rate 0.0001, batch size 10 and 100 epochs, and SGD with learning rate 0.01, momentum=0.8, decay=0.0001, batch size 32 and 100 epochs are selected to be the four deign choices. Model 4 to model 7 are named for these specific hyperparameter settings consecutively. For **Initialization**, there are two choice including kernel and bias. There are 4 models

Table 1: out-of-sample accuracy

Model	MNIST dataset	fashion-MNIST dataset
baseline model	0.9521	0.8605
model 1	0.9623	0.8631
model 2	0.9647	0.8581
model 3	0.9661	0.8737

in this section. The first model used random normal with mean 0 and standard deviation 0.05 for kernel and zeros for bias. The second model used random uniform with lower limit -0.05 and upper limit 0.05 for kernel and zeros for bias. The third model used truncated normal with mean 0 and standard deviation 0.05 for kernel and zeros for bias. The forth model used random normal with mean 0 and standard deviation 0.05 for kernel and ones for bias. Similarly, model 8 to model 11 are assigned to these four models. For **Regularization**, this part include dropout, and batch normalization method. All models are created by adding one or two parameters on baseline model. There are 6 models in this section. 4 model use different dropout rate(0.01, 0.03, 0.05, 0.1), 1 model use batch normalization method, and 1 model use both dropout = 0.03 and batch normalization method. Since the early stopping is aiming to avoid the situation where the training accuracy does not improve over too many epochs, and does not effect the model accuracy, it is used on all models with different hyperparameter design choices throughout this project.

3 Result

3.1 Network Architecture

Upon implementing the methods stated in the previous section, table 1 indicats the performance results obtained from the MNIST dataset and fashion-MNIST dataset. Moreover, out-of-sample accuracies are plotted as a function of training epoch for both datasets, which are shown on figure 1 and figure 2.

For this network architecture segment, the effect of it on model performance is tested by attempting different hidden layers and nodes. As it indicated on table 1, these three models with new architecture network designs all outperform the baseline model in both datasets. Additionally, from these results, the performance ranking from model 1 to model 3 does not quite share the same pattern on MNIST dataset and fashion-MNIST dataset. While model 1 generates the lowest accuracy on MNIST dataset, model 2 has the lowest accuracy on fashion-MNIST dataset. However, model 3 with 3 hidden layers and 26 nodes generates the highest out-of-sample accuracy on both MNIST dataset and fashion-MNIST dataset.

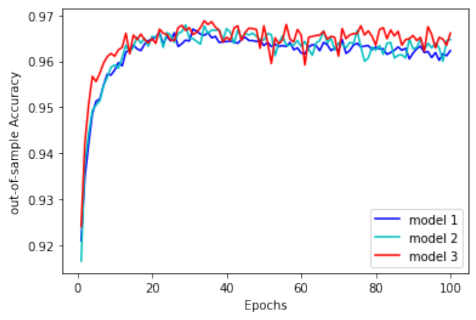

Figure 1: MNIST data

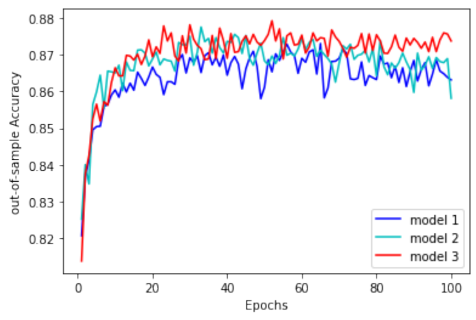

Figure 2: Fashion-MNIST data

Table 2: out-of-sample accuracy for different optimizations

Model	MNIST dataset	fashion-MNIST dataset
model 4	0.9529	0.8595
model 5	0.9552	0.8641
model 6	0.9528	0.8635
model 7	0.9594	0.8659

3.2 Optimization

Similar to the previous section, table 4 indicated the out of sample accuracy. Figure 3 and figure 4 show the plot of the relation between Epochs and accuracy.

When tuning the optimizer hyperparameters, it suggests that when learning rate is high, the accuracy converges in a quite high speed, specifically within 10 epochs. Thus, reducing epochs and learning rate is attempted to fix the overfitting issue. Moreover, reducing the batch size is likely to increase the accuracy. As for the performance results, model 7 with SGD generated the best out-of-sample accuracy on both MNIST dataset and fashion-MNIST dataset. However, the rank of the performances of model 4,5,6,7 is not consistent on these two different data sets. A possible conclusion is that the model selected based on the digit data set does not necessarily perform as good as it does on the fashion data set.

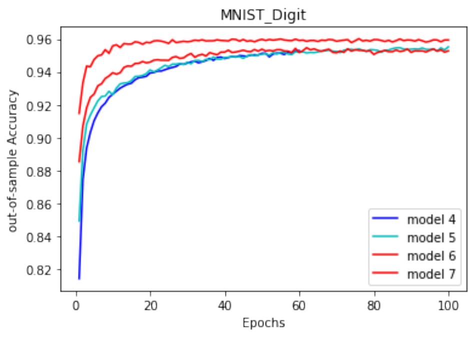

Figure 3: MNIST data

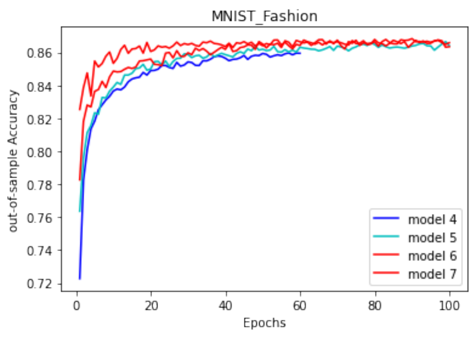

Figure 4: Fashion-MNIST data

3.3 Initialization

Table 3 shows the out-of-sample accuracy under different initialization hyperparameter settings. Also, the plots of out-of-sample accuracy versus training epochs are presented in figure 5 and figure 6.

For initialization, four models are created by using different initialization for kernel and bias in fully-connected layers. In the baseline model, it uses the default initialization: glorot uniform for kernel and zeros for bias. In order to try different things here, the bias is set to be using. The result for the first three models fitted on MNIST shows that when using zeros for bias, the random normal with 0 mean and 0.05 sd gives the best result. After trying to keep the kernel initialized by random normal and change zeros to ones for bias, it gives the best result, which is the yellow line on the top of the others in Figure 5. After applying all four models to fashion-MNIST dataset, it gives the same result. In details, random normal outperforms random uniform and truncated normal, and ones bias performs better than zeros bias when keeping random normal for kernel.

3.4 Regularization

Similarly, table 4 demonstrates the out-of-sample accuracy under different regularization hyperparameter settings. Plots of out-of-sample accuracy versus training epochs are presented from figure

Table 3: out-of-sample accuracy for different initialization

Model	MNIST dataset	fashion-MNIST dataset
model 8	0.9530	0.9116
model 9	0.9517	0.9071
model 10	0.9513	0.9039
model 11	0.9550	0.9182

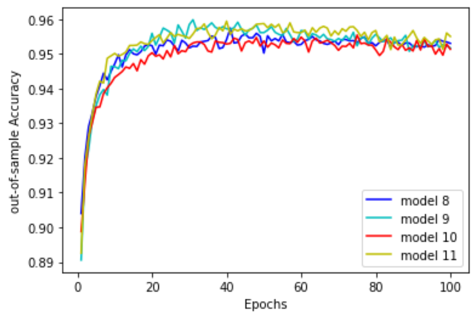

Figure 5: MNIST data

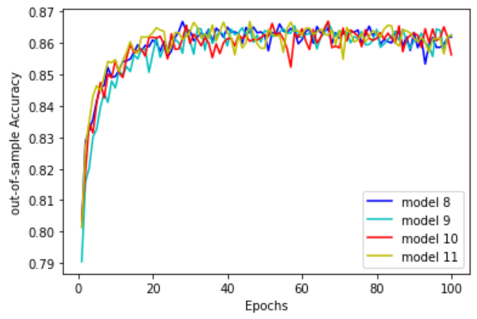

Figure 6: Fashion-MNIST data

7 and figure 10. Since the baseline model is trained again in this section, the training and testing accuracy is a little different from previous part.

This section uses dropout, and batch normalization method. All models are built by adding one or more hyperparameters to the baseline model. There are 6 models in this section. In details, 4 models use different dropout rate, 1 model use batch normalization method, and 1 model use both dropout and batch normalization method.

Dropout rate represents the percentage of nodes will be ignored for each hidden layer. This method can avoid overfitting. By comparing the regularization result, the training accuracy decreases with the increasing of the dropout rate. When dropout rate < 0.03, the test accuracy will increase as the dropout rate increases, which indicates that more nodes could be ignored. When dropout rate $>$ 0.05, the test accuracy will decrease when increasing the dropout rate, which indicates that too many nodes are ignored. Thus the best choose of dropout rate should between 0.03 and 0.05. Since the training accuracy and testing accuracy is closest for dropout rate = 0.03, this rate makes the model performance most stable in both train and test data set. In others words, it is less possible that the model overfitting or underfitting if dropout rate = 0.03.

Batch Normalization normalizes the inputs to layers within the network, thus it is a conservative method to avoid weights explosion. The result shows that batch normalization increase the testing accuracy and decrease the training accuracy. In other words, this method makes the model more generally, and make the same role as dropout.

Table 4: Regularization Result

Model	M Train Acc	f-M Train Acc	M test Acc	f-M test Acc
Baseline Model	0.9842	0.9126	0.9476	0.8666
Dropout=0.03	0.9566	0.8826	0.9606	0.8655
Dropout= 0.01	0.9709	0.8984	0.9559	0.8685
Dropout=0.05	0.9478	0.8711	0.9585	0.8717
Dropout=0.1	0.9277	0.8422	0.9547	0.8612
Batch Normalization	0.9782	0.9141	0.9558	0.8607
BN&Dropout=0.03	0.9609	0.8901	0.9582	0.8688

The last model combined batch normalization and dropout rate = 0.03. The result shows that this model performs better than the others, since its training accuracy and testing accuracy is closest, and both of them are relatively high among the result.

For fashion-MNIST dataset, the model with dropout rate = 0.05 performs more general than others

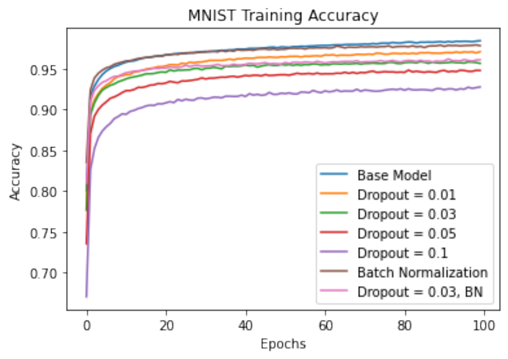

Figure 7: MNIST Train accuracy

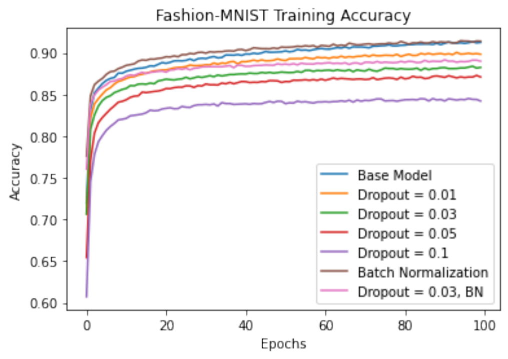

Figure 8: fashion MNIST Train Accuracy

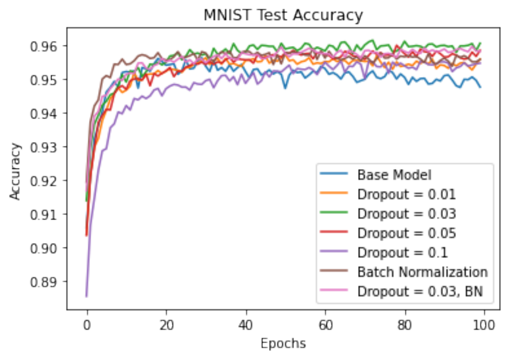

Figure 9: MNIST Test accuracy

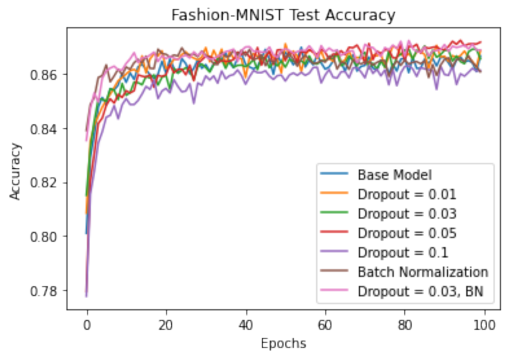

Figure 10: fashion MNIST Test accuracy

4 Discussion

In conclusion to the results section, with all hyperparameter equalling to the baseline model, the model with 3 hidden layers and 26 nodes generates the best out-of-sample accuracy on MNIST dataset among all attempted models. However, as for the fashion-MNIST dataset, the model using random normal with mean 0 and standard deviation 0.05 for kernel and ones for bias gives the highest out-of-sample accuracy.

After a further analysis on the results, it suggests that the ranking of performance from different hyperparameters design choices varies on these two datasets, MNIST and fashion-MNIST. However, under the hyperparameters related to initialization, the performance of different settings shares alike ranking on MNIST dataset and fashion-MNIST dataset. Additionally, the change of network architecture hyperparameter settings improves the out-of-sample accuracy most on the MNIST dataset, while the change of initialization enhances the out-of-sample accuracy most on the fashion-MNIST dataset.

With these being said, current results obtained from attempting the chosen hyperparameter values on MNIST and fashion-MNIST dataset are not quite sufficient to generate implications with respect to the effect of hyperparameter choices in a wider scope. The best performed model with certain hyperparameter choices on certain dataset might not be able to generate the equally highest accuracy on a different dataset. For the purpose of obtaining more insights on this topic, the possible methods would be to test these hyperparameter out on more datasets with more design choices. Through these, one might be able to perceive some pattern or trend of the effect on model performance under certain hyperparameter.

Broader Impact

As the fact that the MNIST dataset and fashion-MNIST dataset do not quite capture the complexity and diversity of real life data, it might be insufficient to completely apply the results from this specific paper to a broader use. When building neural network models, testing out potential reasonable hyperparameter design choices according to corresponding dataset might be the best approach to reach the foremost performance.

References

[1] Feiyang Chen, Nan Chen adn Hanyang Mao, and Hanlin Hu. Assessing Four Neural Networks on HandwrittenDigit Recognition Dataset (MNIST). *CHUANGXINBAN JOURNAL OF COMPUTING*, 2018.

[2] Akmaljon Palvanov and Young Im Cho. Comparisons of Deep Learning Algorithms for MNIST in Real-Time Environment. *LIJFIS*, 2018.