COMP551 Project 3

Classification of Image Data

Yubai Zhang, Xiaoman Sun, Chuqi Wang

2nd April 2022

Abstract

Multilayer Perceptrons (MLP) and Convolutional Neural network (CNN) are two most common used artificial neural network models in image classification by recent advances.In this study, MLP and CNN are implemented in Fashion-MNIST dataset to analyze the performance of these models for image classification tasks on different number of hidden layers and layer hyperparameters.We found that CNN achived better performance than MLP with 93% and 83% respectively. Generally, both algorithms perform better with data normalization. To improve the accuracy of the MLP model, we explored several approaches including changing the number of depth, using various activation functions, applying dropout regularization and data normalization. In the mean time, our research showed that tuning hyperparameters would increased converge efficiency. Additionally, we extended a CNN-based model: Residual neural networks and demonstrated a stable and strong performance from the CNN-based model on image classification tasks.

Contents

1. Introduction

In recent years, deep learning has been extensively applied to a variety of fields, such as computer vision, natural language processing, and speech recognition. Multilayer Perceptron (MLP) is the most simple deep learning algorithm and Convolutional Neural Network (CNN) is the most frequently used model for computer vision. Many computer vision tasks rely on Image classification to assign a label to an input image. Examples include: object identification, car automation, and traffic signal processing.

In this paper, we present CNNs and CNN-based residual neural networks(ResNet) for training parameters on the Fashion-Mnist dataset, a noval image dataset for benchmarking [1]. A MLP implementation was developed from scratch using the stochastic gradient descent algorithm, and CNN and ResNet from PyTorch were compared to Tensorflow Keras. In addition to identifying the best hyperparameters, we also investigated the effect of individual parameters, such as learning rate, depth, width in MLP, and kernel size, padding, strides, etc.in CNN. Results demonstrate that ResNet performs the best when it comes to classifying image data, followed by CNN and MLP.

2. Datasets

Fashion-MNIST dataset

Fashion-MNIST dataset is a collection of 28*28 grayscale Zalando's article images which consisting of a training set of 60,000 examples and a test set of 10,000 examples. There are 10 labels in total and each label corresponds to one type of cloth, and each training and testing set is able to classified into one of the 10 classes. The main task is to implement models to classify each input image data into a class. A sample of the first nine images from the training set is shown in Figure 1. The sample set were flattened and transformed into column vectors for fitting into our MLP model. This made each image a 784 dimensional vector. In terms of CNN model, we maintained their shape with 28*28. The dataset is normalized by dividing

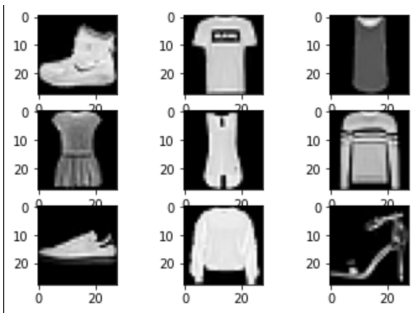

Figure 1. First nine images in training set

by 255 so that they are all of are of approximately the same scale. One hot coding is applied to labels to turn the class variable into a binary class matrix.

3. Results

3.1 Multilayer perceptron model

Impact of Number of Layers

After data processing and normalization, we trained our MLP model using no hidden layers, 1 hidden layer, and 2 hidden layers both with 128 neural to investigate how the number of layers affect the performance on the accuracy of the MLP model. By using the ReLU activation function and setting the learning rate = 0.1 and iterations = 100, the train and test accuracy for each model are shown in Table 1. As we expected, the MLP with no hidden layer has a linearity relationship between inputs and output labels which achieved poor accuracy on both test and train data. If we increase the hidden layer to 1, the train and test accuracy are about 10% better than no hidden layer MLP. However, the MLP with 2 hidden layers having the same amount of units as the 1 hidden layer model didn't have a better accuracy compared to 1 hidden layer MLP. Performance can not be improved by adding more hidden layers. The training accuracy and test accuracy of 2 hidden layers are 76.29% and 75.2% respectively. Due to the additional non-linearity involved in an MLP with two hidden layers, we can conclude that it might be too complex for our dataset. It can be found from Table 1 that of all these proposed models, the MLP with one hidden layer performed the best. Compared to the

model with 0 hidden layers, it adds more non-linearity and complexity, resulting in a better performance.

	Train accuracy	Test accuracy
MLP with no hidden layers	67.60%	66.45%
MLP with 1 hidden layers	77.98%	76.86%
MLP with 2 hidden layers	76.29%	75.2%

Table 1. MLP with different number of hidden layers that performance on train and test accuracy

Impact of Activation Functions

To choose the best activation function for our MLP model, we compared three different activation functions that are ReLU, tanh and Leaky-ReLU. We chose MLP with 2 hidden layers having 128 units and set the learning rate to be 0.1 and iterations to be 100 times again. The result is shown in Table 2. As we can see, the MLP using tanh activation has the best accuracy performance on both the train and test datasets. The hyperbolic tangent activation is often preferable for optimization since as close to zero it is similar to a linear function rather than an affine function when using logistics. Based on the improvement in deep network training, as well as the fix to the zero-gradient issue, the Leaky ReLU activation was expected to perform better. Thus the result was not as we expected.

	Train accuracy	Test accuracy
MLP using ReLU activation	76.29%	75.2%
MLP using tanh activation	78.89%	77.7%
MLP using Leaky-ReLU	77.41%	76.73%

Table 2. MLP with 2 hidden layers having 128 units with different activation function that performance on train and test accuracy

Impact of Dropout Regularization

By using the same iterations and learning rate as before, we trained 2 hidden layers models having 128 units but added dropout regularization this time. The result is shown in Table 3, from the table we can conclude that adding dropout regularization decreased test accuracy instead of improving the accuracy of our dataset. The more neural we drop, the less accuracy we would get. This is because the train and test accuracy of 2 hidden layers MLP without dropout are very close which indicates that the model is a little underfitting instead of overfitting, therefore, adding dropout regularization would make the model simpler, having a worse train and test accuracy on the dataset.

	Train accuracy	Test accuracy
MLP without dropout	76.29%	75.2%
MLP with 10% dropout percentage	73.90%	73.03%
MLP with 20% dropout percentage	72.68%	71.67%
MLP with 30% dropout percentage	73.02%	72.10%
MLP wiht 40% dropout percentage	72.41%	72.16%
MLP wiht 50% dropout percentage	71.06%	70.73%

Table 3. MLP with 2 hidden layers having 128 units with different dropout rate that performance on train and test accuracy

Impact of Normalization data

To investigate the impact of normalization, we still picked up 2 hidden layers MLP with ReLU activations. By setting the learning rate to be 0.1 and iterations to be 100 times, we found out that the accuracy of the model on normalized data is about 34% higher than that of model on unnormalized data from Table 4. As we can see data normalization significantly improve the accuracy of the MLP model on the Fashion-MNIST dataset.

	Train accuracy	Test accuracy
MLP using normalized iamges	76.29%	75.2%
MLP using unnormalized images	52.13%	51.41%

Table 4. MLP with 2 hidden layers having 128 units using normalized and unnormalized data that performance on train and test accuracy

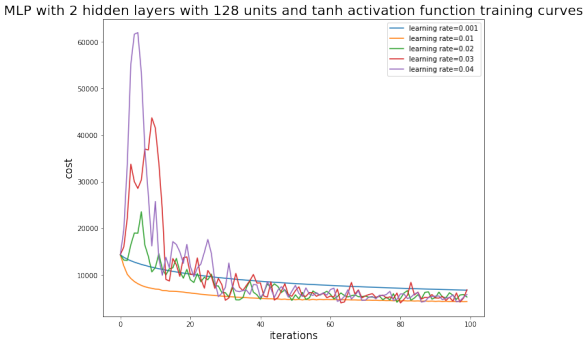

Figure 2. 2 hidden layers MLP training curves

Best MLP Model

To choose the best MLP model, since MLP with 1 hidden layer has similar accuracy performance to 2 hidden layer MLP. We set both models with tanh activation and 100 iterations and then change the learning rate from 0.001 to 0.16 to find the best learning rate for both MLPs and compare with the accuracy. From Figures 1 and 2 we could conclude that increasing the large learning rate would cause a significant change in cost and oscillation as iterations get larger and choose the best learning rate for 2 hidden layers MLP is about 0.09. Similarly, the training curves and impact of learning rate

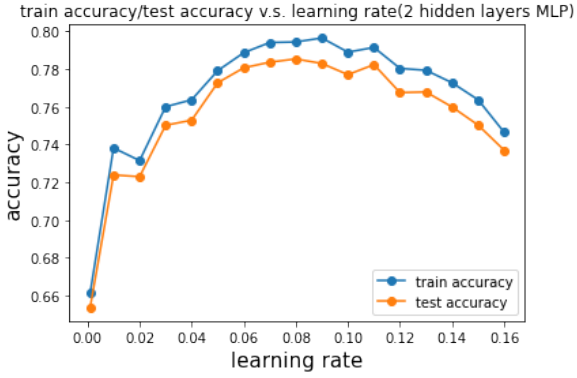

Figure 3. 2 hidden layers MLP(accuracy v.s. learning rate)

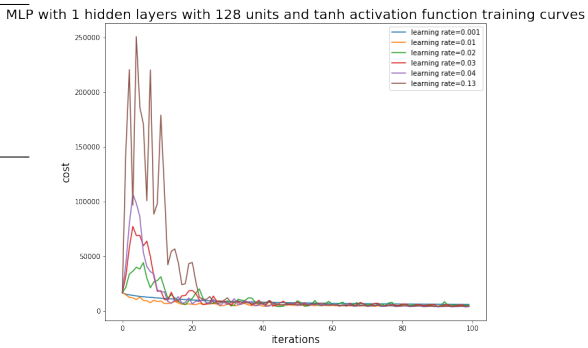

Figure 4. 1 hidden layers MLP training curves

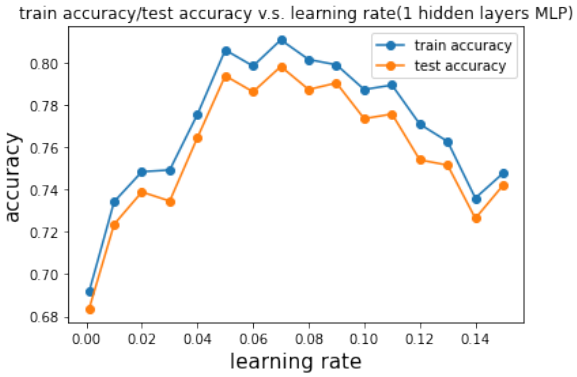

Figure 5. 1 hidden layers MLP(accuracy v.s. learning rate)

on accuracy for 1 hidden layer MLP are shown in Figures 4 and 5. The best learning rate for 1 hidden layer MLP is about 0.08. Comparing both 1 hidden layer MLP and 2 hidden layerss MLP, we found out the best test accuracy of 2 hidden layer MLP on Fashion-MNIST dataset is 78.59% when learning rate = 0.09, while the best test accuracy for the other MLP is 79.83% which is slightly higher than the test accuracy 2 hidden layers MLP. Therefore, we decided to choose 1 hidden layer MLP with a learning rate = 0.08, increased iterations and trained it with ReLU and Leaky-ReLU to get the best model.

	Train accuracy	Test accuracy
MLP with ReLU activation	84.42%	82.63%
MLP with tanh activations	84.46%	82.46%
MLP with Leaky-ReLU activations	84.50%	82.61%

Table 5. MLP with 1 hidden layers with 128 units using different activations

Table 5 shows the training accuracy and test accuracy on different activation functions for 1 hidden layer MLP, by increasing iterations to 1000 and choosing the best performance learning rate to be 0.08, we finally improved the test accuracy to about 82%. We could also conclude that ReLU and Leaky-ReLU activation functions have a better performance on test accuracy than tanh activation

3.2 CNN Model

Best CNN Model

By utilizing the tensor-flow relative libraries, we constructed our CNN model as required and we tried various number of approaches to obtain the best performance model based on test accuracy. First of all, we determined the best batch size by running multiple experiments in a number of 30 epochs and find the highest test accuracy. The result is shown below in table 6. As the table indicates,128 batch size and 64 batch size are both good for our CNN model relative to 32 batch size. However, model fitting with 128 batch sizes runs more efficiently than the 64 batch size one. Therefore, we decided to choose the 128 batch size for fitting our model and running the future experiments.

	Test accuracy
128 batch size	91.78%
64 batch size	91.72%
32 batch size	91.40%

Table 6. CNN different batch size with same epochs

After deciding the appropriate batch size, we now test for different number of epochs in order to get the best performance model for our required CNN. We selected from 10-50 number of epochs to fit the model and the following is a simple comparison based on the testing accuracy.

	Test accuracy
epochs=10	91.13%
epochs=20	92.03%
epochs=30	91.20%
epochs=40	91.42%
epochs=50	90.96%

Table 7. CNN different number of epochs with 128 batch size

From the result shown in the table 7, it is clear that 20 epochs to train our model is the best choice since it is cheap and has a higher testing accuracy. Furthermore, after we determined the optimal value of these two hyperparameters, we are now having a regular CNN model with two convolution layers, then 2 fully connected layers and a final layer with softmax. In order to improve our model's performance, we also introduced the dropout regularization with a dropout rate of 25% to the layers. After training the CNN on the dataset by adding this dropout regularization, we successfully obtained a better testing accuracy which was 92.64%. We then save this model as the control group for future experiments on different hyperparameter comparison when constructing the convolution layers. In summary, we decided to fit our CNN model with batch size equals to 128 and the number of epochs equals to 20 with a dropout percentage of 0.25(25%).The highest test accuracy is obtained from this saved model which is 92.64%.

Impact of Convolution Layer Hyperparameter

In our experiment, we decided to explore more on how the CNN's convolution layer hyperparameter kernel size influence our model performance. After doing some research on the internet. We discovered that it is better to choose a kernel size with odd numbers rather than even numbers.[2] Therefore, we started to train our CNN model with a 5*5 kernel size and a 7*7 kernel size for the convolution layer and then run the evaluation function on these models to see the performance compares to the original one with 3*3 kernel size. Here is the result in table.

	Test accuracy
kernel size=(3*3)	92.64%
kernel size=(5*5)	92.15%
kernel size=(7*7)	92.18%

Table 8. Different kernel size influence the Test accuracy

It is obvious that choosing a 3*3 kernel size for our convolution layer is the best choice.

Impact of Dropout Regularization

In the mean time, we also tested the effect of dropout percentage on the layers for the CNN model. In our experiment, we compared the difference between the test accuracy for the three distinct dropout percentages and the result is presented in the following table:

	Test accuracy
dropout rate= 25%	92.64%
dropout rate= 50%	92.05%
dropout rate= 75%	91.87%

Table 9. Different Dropout rate influence the Test accuracy

Similar to the result of the influence of Dropout regularization on MLP models, as we increase the dropout percentage, the test accuracy is decreasing in general. And for our current CNN model, by adding a dropout regularization with a dropout percentage equals to 0.25(25%), the model performs the best.

3.3 Comparison

	Test accuracy
Best MLP Model	82.36%
Best CNN Model	92.64%

Table 10. Best Model Comparison of MLP and CNN

In summary, as Table 10 indicated, the test accuracy of the best model we determined of CNN model is higher than the one of MLP model with approximately 10%. Additionally, during the training process, we also found that the training accuracy of CNN model is generally higher than the one of MLP model. Sometimes, the training accuracy of CNN model would even converge to 99% when having a large number of epochs.Therefore, for our fashion MINST dataset, we concluded that it is better to use CNN model for the image classification rather than MLP.

4. Discussion and Conclusion

In conclusion, we learned how to implement MLP from scratch and how to use our own model to solve images classification. Although our MLP model didn't achieve very high train and test accuracy, we found that the cost are extremely high at the beginning of iterations. After the stochastic gradient descent, it's hard for SGD to decrease the cost close to zero which means gradient descent may find a local optima. We would like to investigate more approach such as using momentum or adagrad to solve this kind of problem in the future. Furthermore, by comparing the results obtained from the two different models, in general, CNN model has a higher training and testing accuracy than the MLP model we implemented. That is the reason why CNN now is a current favourite of computer vision algorithms and it wins multiple ImageNet competitions.[3] Due to limit amount of time, we only investigated partial fields of the convolution layer hyperparameters(kernel size). Since we have already constructed a good performance model. In the future, we are able to run more experiments on a variety categories of convolution layer hyperparameters and compare the final testing accuracy so we can get a better and better result in testing accuracy to improve our CNN model performance.

5. State of Contributions

Xiaoman Sun: Acquire, preprocess and analyze the data, contribution on writing report.
Yubai Zhang: Implementation of the CNN, contribution on writing report.
Chuqi Wang: Implementation of the MLP, contribution on writing report.

References

[1] *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. 2017. URL: http://arxiv.org/abs/1708.07747.

[2] Swarnima Pandey. *How to choose the size of the convolution filter or kernel size for CNN?* July 2020. URL: https://medium.com/analytics-vidhya/how-to-choose-the-size-of-the-convolution-filter-or-kernel-size-for-cnn-86a55a1e2d15.

[3] Uniqtech. *Multilayer perceptron (MLP) vs Convolutional Neural Network in deep learning*. June 2019. URL: https://medium.com/data-science-bootcamp/multilayer-perceptron-mlp-vs-convolutional-neural-network-in-deep-learning-c890f487a8f1.