MiniProject3

Group 20: Amelia Cui, Evelyn Cao, Zhengxin Chen

March 29, 2022

1 Abstract

In this project, we investigated multi-layer perceptrons (MLP) and Convolutional neural networks (CNN) on the fashion-MNIST dataset. We first delved into the details of MLP implementation. We explored the architecture of MLP including depth, width, choice of activation function and regularization strategy. We also conducted experiments to find best hyper-parameters for MLP. On the other hand, we built a CNN model to compare its performance with MLP. Generally, the main result of our experiment is that with appropriate architecture MLP can ahieve high test accuracy which is close to CNN.

2 Introduction

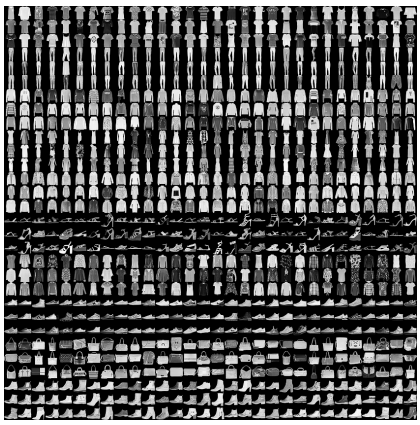
Nowadays, image recognition plays an important role in the machine learning field due to its large number of applications in the real world industry. In this project, we aim to analyse factors that impact the performance of MLP in image recognition for the fashion-MNIST dataset, and compare it with the CNN model. With default conditions, CNN performs better than MLP. But, we could improve the performance of MLP by proper methods, including increasing the network depth, choosing an appropriate activation function and using regularization strategy. Finally, MLP could reach a high accuracy closing to the CNN model.

The fashion-MNIST data set is a dataset of Zalando's article images which consists collection of 70,000 images of clothes and bags It is widely used by many researchers in machine learning and deep learning-related field. We read the paper about the performance of MLP on fashion-MNIST dataset. Its finally accuracy about this dataset is 85.6% which close to our MLP with tanh activation function.

3 Datasets

Fashion-MNIST is a dataset of Zalando's article images—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.The dataset is open-source on GitHub.Below is an example of the image data and the relationship between label and fashion items.

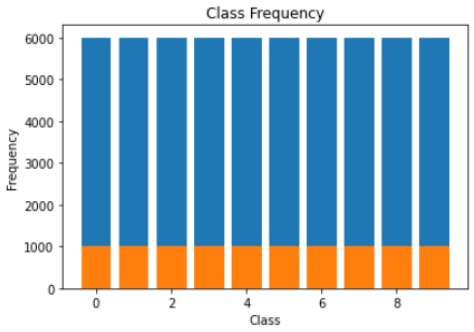
As the part of the data preparation, we change the dimension of the data, reformatted the y labels in order to fit the data for the one-hot coding structure. And we apply vectorization and normalization to the data in an attempt to obtain a evenly scaled set of data as well as enhanced performance.



(a) Sample

Label	Description
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

(b) Labels



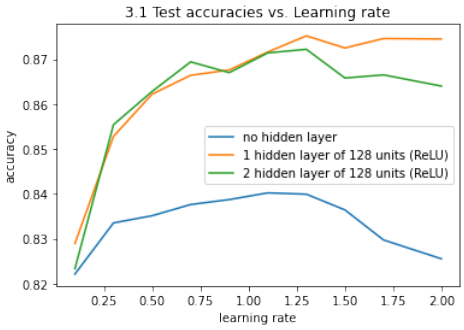
(c) Class Distribution

Figure 1: Data visualization

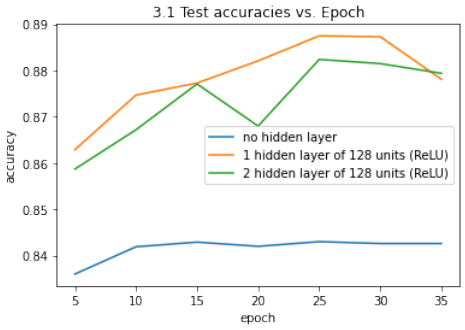
4 Results

4.1 Impact of depth and activation function

Non-linear activation function allows the stacking of multiple layers of neurons. Our experiment showed that the 1-hidden-layer model achieved the highest test accuracy 88.8% with learning rate 1.3, significantly higher than the 0-hidden-layer and slightly higher than the 2-hidden-layer model. Generally, deeper network is more expressive thus has better performance, but it also has the potential of over-fitting. Also, deeper network is more complex and needs higher learning rate and more training epoch to converge.



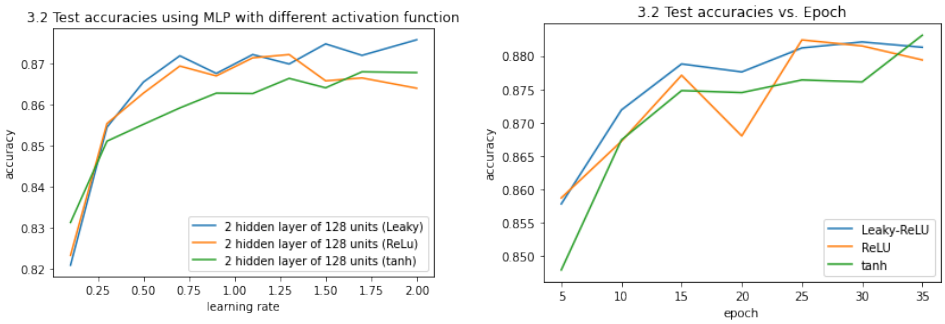
(a) Accuracy vs. Learning rate



(b) Accuracy vs. Epoch

Figure 2: Impact of depth

Regarding activation functions, we tried three functions with the 2-hidden-layer model. ReLu gave higher test accuracy than tanh since tanh had the vanishing gradient problem. Leaky-ReLU performed slightly better than ReLU when training epoch was low, but they tended to have similar performance when epoch increased. Note that there were a few exceptions when plotting Accuracy vs. Epoch, we think that was because some randomization process in the algorithm and did not contradict our general conclusion.



(a) Accuracy vs. Learning rate

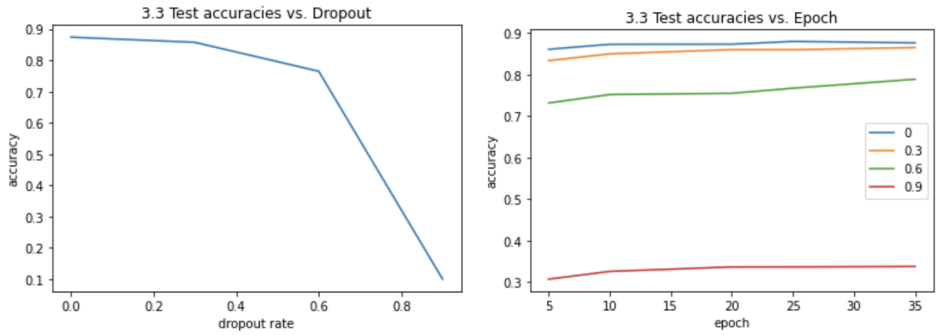
(b) Accuracy vs. Epoch

Figure 3: Impact of activation function

4.2 Impact of dropout regularization and normalization

To investigate the impact of dropout regularization, we added the option of filtering nodes at random with certain probability (dropout rate) at the forward propogation stage. Then we compared with the performance in previous section, using MLP with 2 hidden layers each having 128 units with ReLU activations (learning rate=1.3, training epoch=25). As shown below, when p is high, the neurons left to update is low, thus results in lower accuracy. However, when p is relatively low (0.5), the accuracy does not change much.

To see the impact of unnormalized data, we compared the performance of MLP for normalized and unnormalized data. The experiment showed that unnormalized data had significantly lower accuracy on test set, which was around 10%. This may due to the dying of gradient. When the gradient gets closer to 0, it is unable to adjust the weights anymore. Consequently, the parameters are not adjusted, so the model is unable to improve its accuracy.



(a) Accuracy vs. Dropout rate

(b) Dropout vs. Epoch

Figure 4: Impact of regularization and normalization

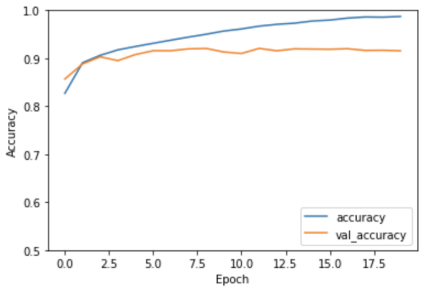
4.3 Compare MLP and CNN

Using TensorFlow library, we created a convolutional neural network (CNN) with 2 convolutional and 2 fully connected layers with 128 units in the fully connected layers and ReLU as activation function, the parameters are shown in Figure5(a). In summary, we added 2 convolution layer, along with pooling. In the final layer, we used softmax classification. CNN with 20 epoch achieved accuracy over 90%, compared to 89% of MLP.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 28, 28, 32)	832
activation (Activation)	(None, 28, 28, 32)	0
max_pooling2d (MaxPooling2D)	(None, 14, 14, 32)	0
conv2d_1 (Conv2D)	(None, 14, 14, 64)	51264
activation_1 (Activation)	(None, 14, 14, 64)	0
max_pooling2d_1 (MaxPooling2D)	(None, 7, 7, 64)	0
flatten (Flatten)	(None, 3136)	0
dense (Dense)	(None, 128)	401536
activation_2 (Activation)	(None, 128)	0
dense_1 (Dense)	(None, 10)	1290
activation_3 (Activation)	(None, 10)	0

Total params: 454,922
Trainable params: 454,922
Non-trainable params: 0

(a) Parameter used for CNN



313/313 - 6s - loss: 0.3990 - accuracy: 0.9156 - 6s/epoch - 21ms/step

(b) Accuracy for CNN vs epoch

Figure 5: performance of CNN

4.4 Best MLP Architecture

Trying to get a better architecture, we tried strategies including increasing network depth, decreasing width and using dropout. The best MLP model we find was the 2-hidden-layer with ReLU activation, while a 1-hidden-layer model worked equally well. The 89% accuracy is close to the 91% of CNN. In details,

1. We selected 2 hidden layer since it was more expressive and flexible and could achieve high accuracy in the meanwhile.
2. We set the width of hidden layer as [128, 80]. As we showed below, decreasing the layer width was a effective way to avoid over-fitting.
3. We did not use dropout regularization since our model was not so deep, there was no much different with small level of much difference when the dropout rate is small.
4. We used learning rate 1.7 and training epoch 25 since the model was more likely to overfit when the epoch number was large, and underfit when the epoch number was small.
5. We used normalized data before training to achieve higher accuracy.

4.5 Other experiments

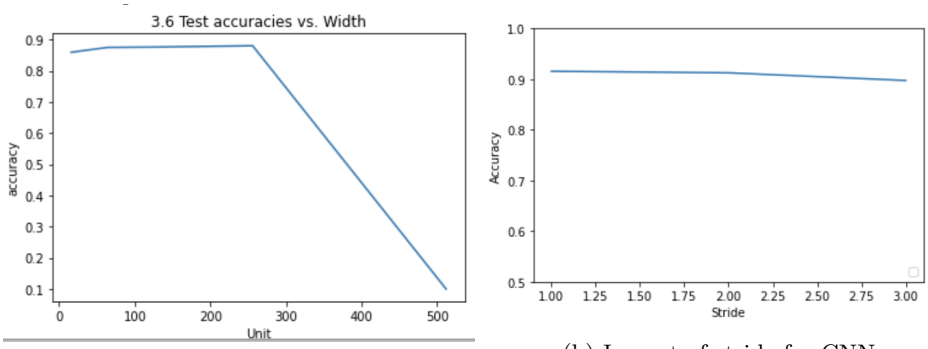
In addition to required experiments, we also investigated into the impact of width (number of units in the hidden layers) of the MLP, the stride hyperparameter of CNN, and training set size. The analysis are all based on test accuracy.

4.5.1 Impact of Width

We used an MLP with 2 hidden layers with ReLU activations, and compared the impact of different units on the network. In detail, we chose 16, 64, 128, 256, 512 units. We found the best accuracy is achieved when width = 128/256. When the number of units is low, it is likely to underfit; while when there are too many units, the model is likely to overfit.

4.5.2 Impact of Stride in CNN

Stride is the number of pixels shifts over the input matrix in CNN. We tested the performance using stride = 1,2,3 on the CNN model, the result is shown in the figure5 below. We observed that when stride is 1 or 2, the performance is similar, where we are able to get 0.91 accuracy. However, when stride is 3, the accuracy is lower, which is around 0.89. This may because when we move the filters 3 pixels at a time, the image is down-sampled too much.



(a) Impact of width for MLP

(b) Impact of stride for CNN

Figure 6: CNN vs MLP

4.5.3 Impact of number of training data in CNN and MLP

In this experiment, we used datasize = 1, 10, 100, 1000, 10000, and using all of the training data. The result is shown in the figure below. For CNN, we used the same model in section 4.3, with stride = 1; for MLP, we used 2-hidden-layer with ReLU activation. We used 20 epoch for both methods. As shown in the figure7, CNN has a better performance when the number of training data is low, it is more stable. For MLP, the accuracy stay low (around 0.1) when number of data is low, the accuracy increases until we increase training points are around 10,000. In the end, when there are enough data, they have similar accuracy.

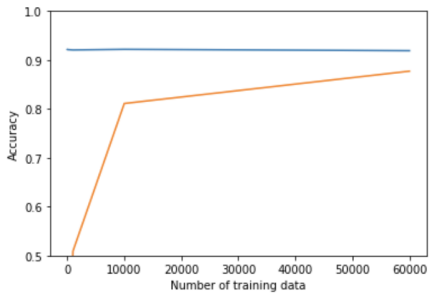


Figure 7: Impact of number of training data

5 Discussion and Conclusion

In this project, we investigated many different aspect of MLP, including width and depth of network, number of epoches, different activation functions, dropout regularization, data normalization. We also investigated into different parameters for CNN, including number of epochs, stride. Additionally, we compared the performance of both model as the number of training data increases. An extension of this study would be weight initialization tuning in the MLP model, and impact of kernel size or padding in the CNN model. By carefully tuning these parameters, we believe that the accuracy can still be increased.

6 Statement of Contributions

Everyone contributes to writing the write-up and coding.

7 Reference

1. Abien Fred Agarap, Arnulfo P. Azcarraga, "Improving k-Means Clustering Performance with Disentangled Internal Representations"

2. Zalando research https://github.com/zalandoresearch/fashion-mnist