Machine Learning - Homework 2

Tintrim Dwi Ary Widhianingsih
20185116

November 23, 2018

Problem 8. Experiment of Convolutional Neural Network (CNN) for fashion MNIST dataset [1].

Fashion MNIST consists 60,000 training and 10,000 testing set images. Each example is a 28×28 grayscale image, associated with a label from 10 classes (T-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, and ankle boot).

CNN is the one of developed method that has powerful capability in object processing. In this section, I implemented CNN to fashion MNIST database. By default, I use 3 convolutional and pooling layers, which is using 32, 64, and 128 filters with the size of 3×3 in each convolutional layer and using 2×2 size of max-pooling window. The strides that is used in both layer is 1. The activation function is activated during convolutional and classification process. By default, I used PReLU [2]. Initializer and optimizer in this experiment are He. (from normal distribution) and Adam [3]. For the fully-connected layer, I used 1 hidden layer with 128 nodes in the hidden layer. To decrease the number of the nodes when passing hidden layer, dropout penalization is implemented using 0.5 as the threshold. Finally, the activation function for getting probability of each category is calculating using Softmax.

For discussion, the default setting is change by some scenarios. The following sub-section, each different scenario result is shown and explained.

[1] https://arxiv.org/abs/1708.07747
[2] https://arxiv.org/abs/1502.01852
[3] https://arxiv.org/abs/1412.6980

a) Performing "Mean Subtraction" and "Normalization" on the input data.

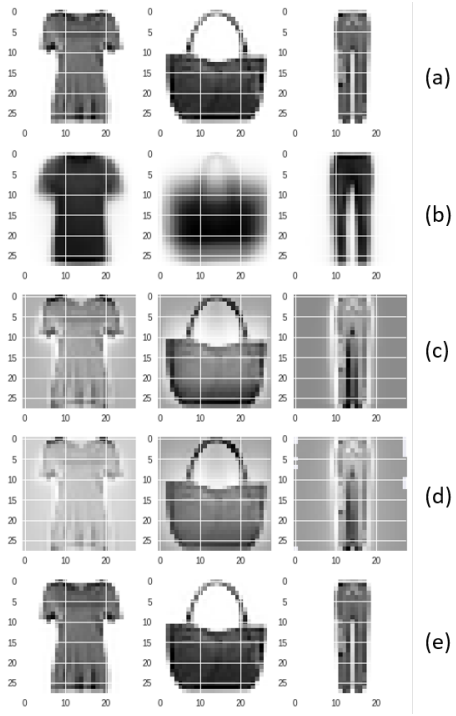

Figure 1: Visualization of 3 Categories: T-Shirt/Top, Bag, and Trouser for (a) Original Image, (b) Mean Value, (c) Mean-Subtraction, (d) Z Normalization, and (e) Grayscale Normalization

For the mean subtraction (Fig.1b), the mean value that is used to subtract the original image is calculated by each category. Then, for the normalization that is used in this experiment are centering (using z value normalization) and normalization of the grayscale value. Centering or z-normalization is calculated by $(x - mean(x))/std(x)$, where the mean and standard deviation of x is calculated by each category. The second method for normalization that I implemented in this experiment is by dividing each value inside the images by 255 (the maximum of grayscale value in the image). Both normalization seems showing different image as an input to our CNN. For the next experiment, I use grayscale normalization, because the image is clearer and more like the original one even if it has been converted to 0-1 boundary value.

b) Performing the result of using Xavier and He initialization.

In this scenario, the setting is to show and compare different initializer, which are He and Xavier. Using different type of both initializers, normal and uniform, we can see that from Fig.2 in the top left, in the training process from the first to about 20th epoch, He initializers (normal or uniform) outperform Xavier or Glorot, but in the last they have almost the same performance. But if we see in Fig.2 in the bottom left, the testing performance of the four different initializer is vary each other even if in the beginning He have the better initial performance.

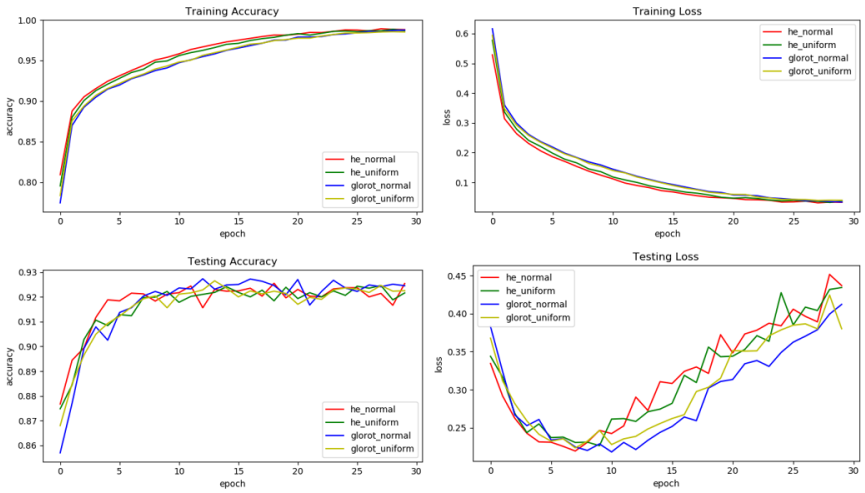

Figure 2: Experiment Result for Different Initialization Scenario

c) The result of the experiment using different setting of layers and hidden nodes.

In this scenario, we want to know how the different configuration influence the performance of our CNN model. The configuration setting that is implemented in this experiment are using different number of convolutional layers, in this case I use 2, 3, 4, and 5 layers. In the first configuration, we use 32 and 64 filters. In the second configuration, I use 32, 64, and 128 filters. In the third one, I use 32, 64, 128, and 512 filters. And the last configuration is using 32, 32, 64, 128, and 512 filters.

In the top left of Fig.3, we can see that using 2 layers of convolutional

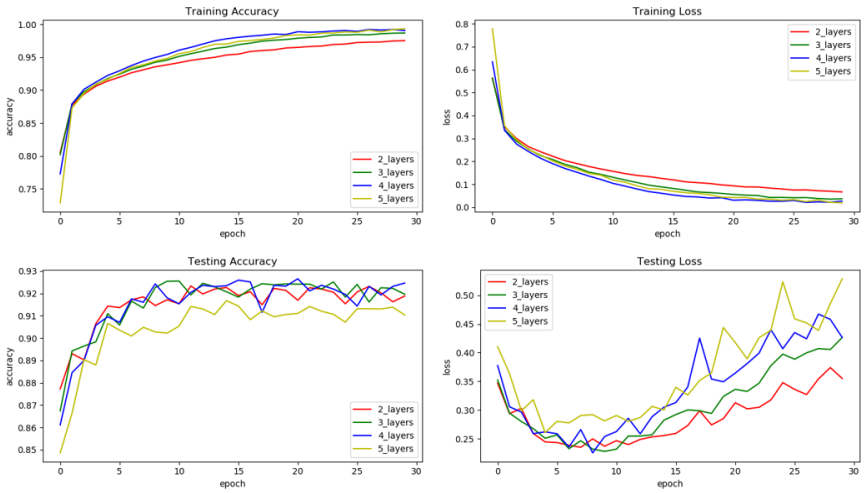

Figure 3: Experiment Result for Different Configuration Scenario

layers the training accuracy is not as better as the other setting that using the more number of layers. But when we compare it to testing accuracy in bottom left of Fig.3, we can see that the performance of using 2 layers is same with using 3 and 4 layers. In the other hand, using 5 convolutional layers obtains the worse testing accuracy among them. This can be stated that using the more convolutional layers is higher possibility to obtain the overfitting model. So, in this scenario, based on this experiment, we can choose 2 convolutional layer configuration as the best scenario.

d) The result of the experiment using different gradient optimization techniques.

In this scenario, we compare different type of the optimizer that is implemented to our CNN model. The optimizers that I used to the experiment are Adam, Adamax, Adadelta, and Adagrad. Using those different type of optimizers, we can see in Fig.4 that Adagrad obtains the worse performance compare to the others. But if we look at the testing accuracy result, we can see that all of the optimizers have almost the same performance. Looking further to the comparison of training and testing accuracy, we can find that the difference of the performance in both dataset from Adagrad is not as big as the other optimizers.

However, Adam, Adamax, and Adadelta obtains almost 100% in the training result. So, based on this experiment, we can say that Adagrad is the most effective to use our CNN model.

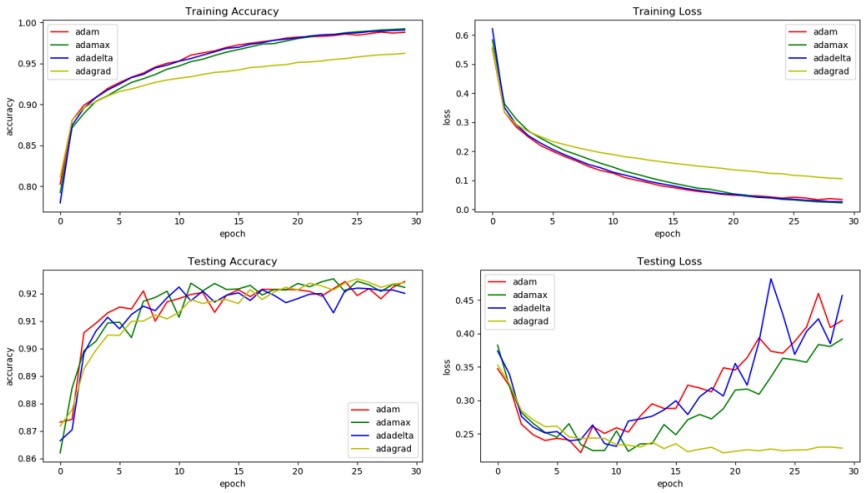

Figure 4: Experiment Result for Different Optimization Scenario

e) The result of the experiment using different activation function

In this section, we want to see the effect of different activation function. Three activation functions that we use in this experiment is PReLU, Leaky ReLU, and ELU. When in the training process, the performance of those activation functions is nearly same each other and increasing almost smoothly until reaching above 97.5% accuracy, see Fig.5. The difference of the effect can be seen in the testing accuracy. We can see that from epoch 15, the graphic of PReLU performance is being farther above the other. In the other hand, ELU has the better performance compare with Leaky ReLU, but the difference is not as big as PReLU. In this configuration, we can easily choose PReLU as the best activation function in our model.

f) The result of the experiment using different type of regularization method

In this implementation, we want to know how the different type of regularization influence the performance of CNN. From the experiment,

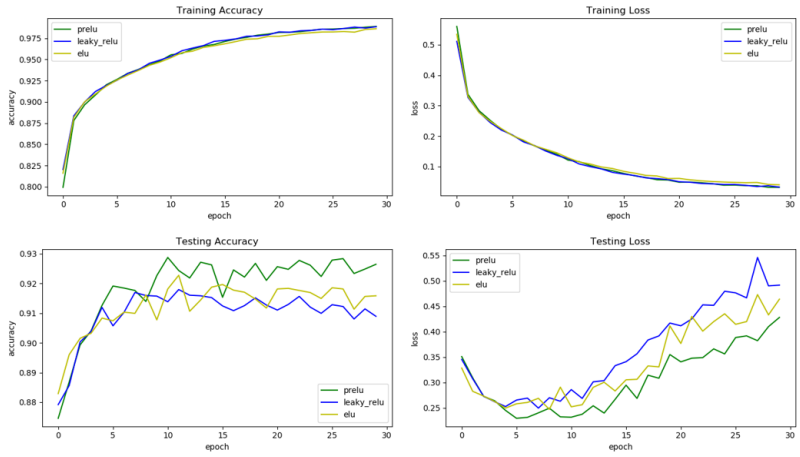

Figure 5: Experiment Result for Different Activation Function Scenario

we can see that the performance of dropout from is mostly better than L1 norm in testing process. But the performance of L2 norm regularizer has tendency to vary between a particular range.

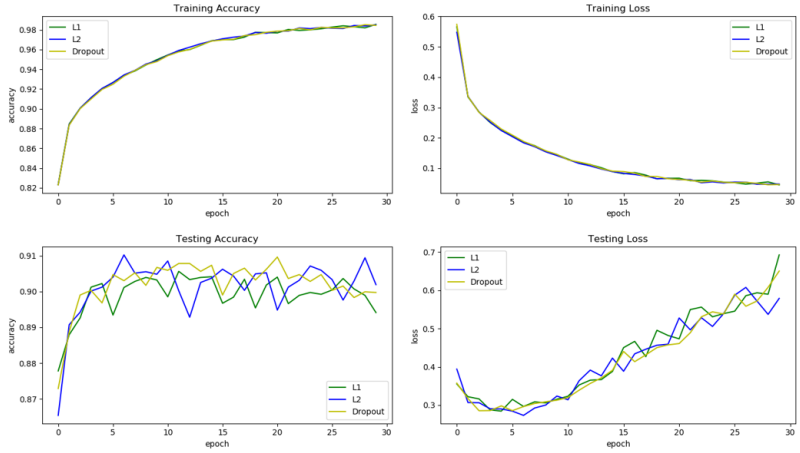

Figure 6: Experiment Result for Different Regularization Scenario

Problem 9. Experiment of Generative Adversarial Nets (GAN) on fashion MNIST dataset.

In this experiment, I applied Conditional GAN. The loss function from the generator in Fig.7 is always greater than discriminator. Both network reaching the convergence value after 800 epoch.

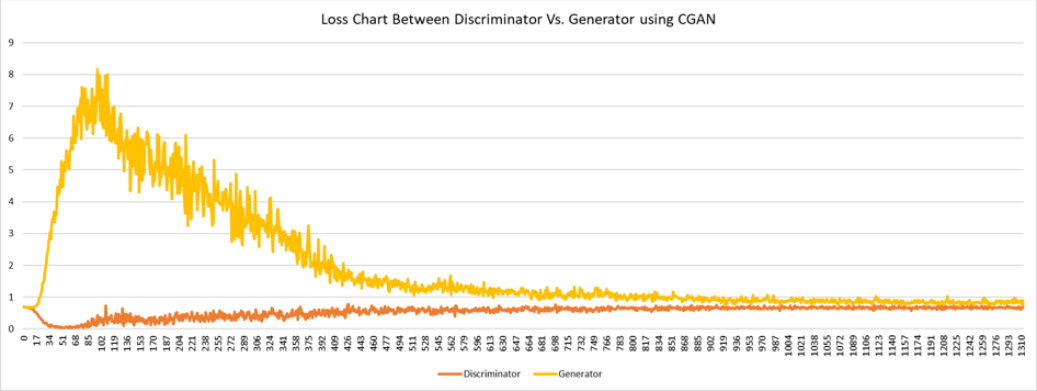

Figure 7: CGAN Experiment Loss Result on Fashion MNIST

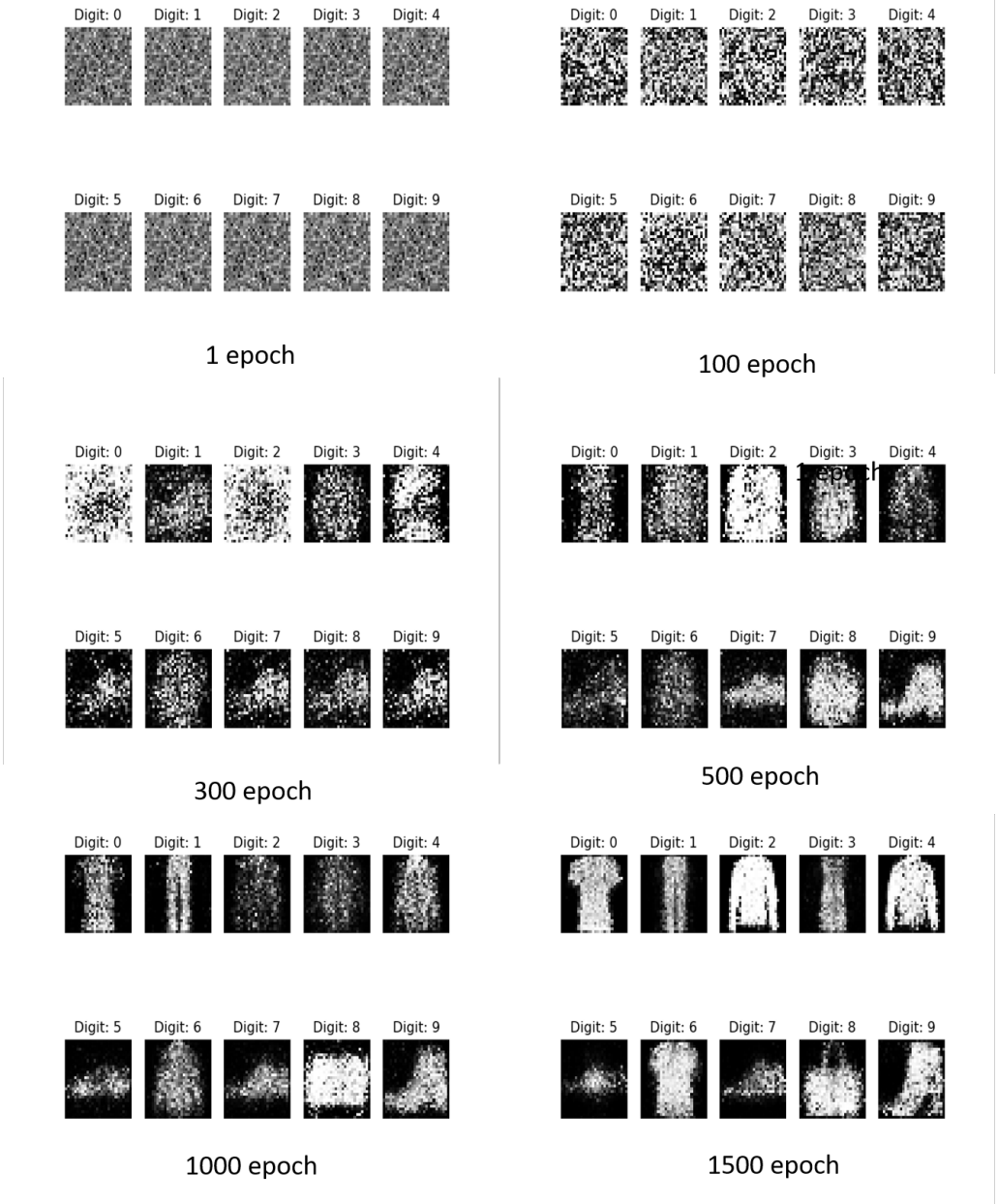

Figure 8: CGAN Experiment Image Result on Fashion MNIST