CSI5138 Assignment2

Name: Jiacheng Hou
Student Number: 300125708

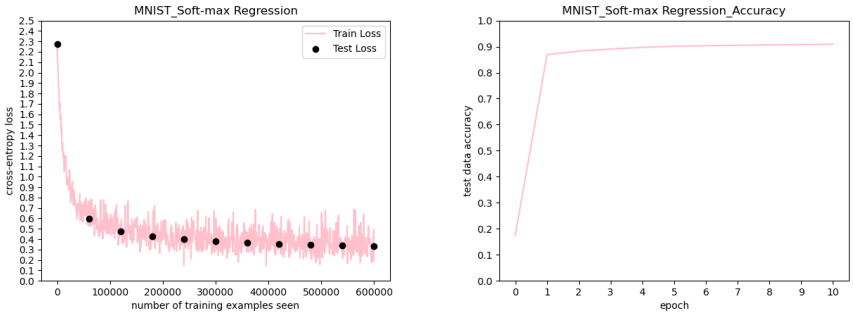
In the assignment, I aim to design and train Soft-max regression, MLP and CNN models for MNIST and CIFAR-10 dataset respectively.

1 MNIST dataset

In this section, Soft-max regression, MLP and CNN models for the MNIST dataset are introduced.

1.1 Soft-max regression model

I instantiated a logistic regression model with input dimension $28 * 28$ and an output dimension of 10. Cross-entropy loss is used to calculate the training dataset and testing dataset loss. I used the SGD optimizer to calculate the parameters' gradients and update them subsequently. With 10 epochs of training, I managed to achieve 91% accuracy on the testing dataset. Figure 1a shows the training dataset and testing dataset loss change over time. Figure 1b shows the testing dataset accuracy changes over time.



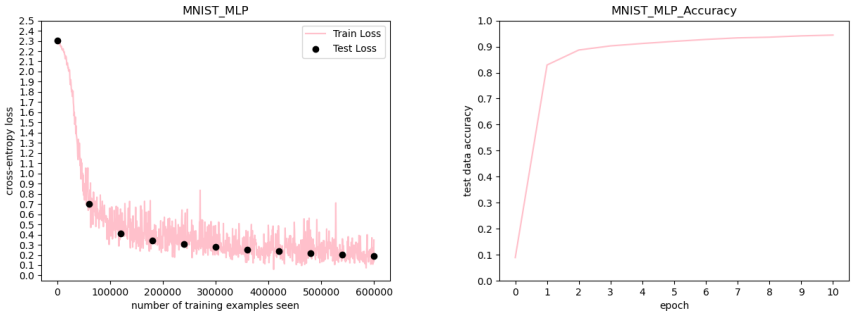
(a) Training dataset loss and testing dataset loss for 10 epochs

(b) Testing dataset accuracy after each epoch

Figure 1: Soft-max regression model on MNIST dataset

1.2 MLP model

I defined an MLP network architecture with a $28 * 28$ dimension input, 2 hidden layers, a dropout layer and an output layer. The first hidden layer has 512 neurons and the second hidden layer has 320 neurons. The output layer has 10 neurons. I used the cross-entropy loss to calculate the loss and the SGD optimizer to update parameters and then trained the model for 10 epochs. Figure 2a shows the training loss and the testing loss change over time. Figure 2b shows the testing data accuracy changes over time. The testing data accuracy achieved around 94% at the end of 10 training epochs.



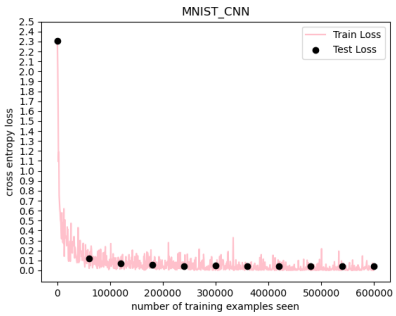
(a) Training dataset loss and testing dataset loss for 10 epochs

(b) Testing dataset accuracy after each epoch

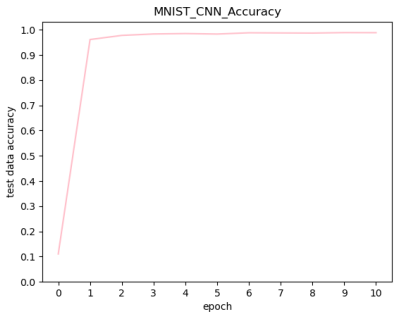
Figure 2: MLP model on MNIST dataset

1.3 CNN model

I defined a CNN architecture with a $28 * 28$ dimension input, two convolution layers, each with 5*5 kernels. After each convolution layer, I used a max-pooling layer with a stride of 2. Then I flattened the tensors and put them into a dense layer, pass through an MLP to carry out the task of classification of 10 categories. I chose Adam as the optimizer and trained the model with 10 epochs. I managed to achieve 99% accuracy on the testing dataset after 10 epochs of training. Figure 3a shows the training loss and the testing loss change over time. Figure 3b shows the testing data accuracy changes over time.



(a) Training dataset loss and testing dataset loss for 10 epochs



(b) Testing dataset accuracy after each epoch

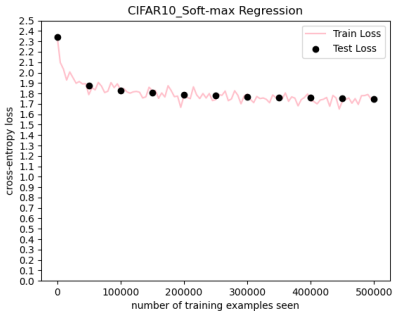
Figure 3: CNN model on MNIST dataset

2 CIFAR-10 dataset

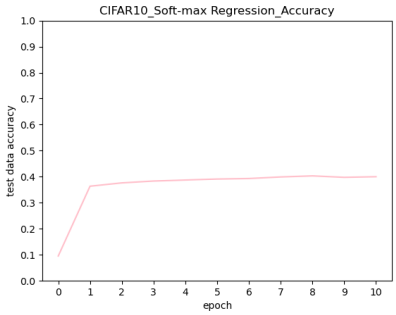
In this section, Soft-max regression, MLP and two CNN models for the CIFAR-10 dataset are introduced.

2.1 Soft-max regression model

I instantiated a logistic regression model with input dimension $32 * 32 * 3$ and an output dimension of 10. Cross-entropy loss is used to calculate the training dataset and testing dataset loss. I used the SGD optimizer to calculate the parameters' gradients and update them subsequently. With 10 epochs of training, I managed to achieve 40% accuracy on the testing dataset. Figure 4a shows the training dataset and testing dataset loss change over time. Figure 4b shows the testing dataset accuracy changes over time.



(a) Training dataset loss and testing dataset loss for 10 epochs

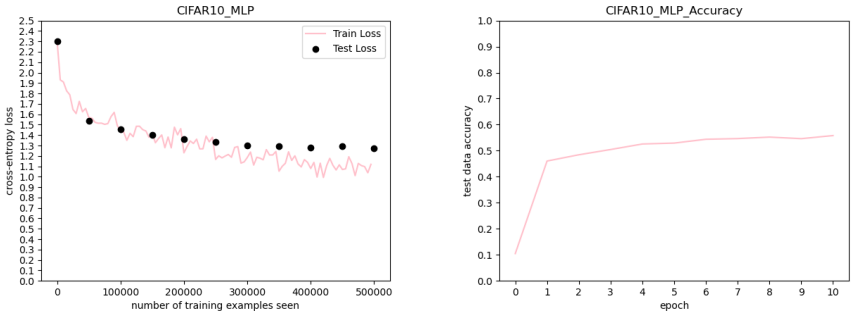


(b) Testing dataset accuracy after each epoch

Figure 4: Soft-max regression model on CIFAR-10 dataset

2.2 MLP model

I defined an MLP network architecture with a $32 * 32 * 3$ dimension input, 3 hidden layers, a dropout layer and an output layer. Three hidden layers include 1024, 512, 320 neurons respectively. After each hidden layer, there is a dropout layer to avoid overfitting. The output layer has 10 neurons. I used the cross-entropy loss to calculate the loss and chose Adam as the optimizer. I trained the model for 10 epochs. Figure 5a shows the training loss and the testing loss change over time. Figure 5b shows the testing data accuracy changes over time. The testing data accuracy achieved around 55% at the end of 10 training epochs.



(a) Training dataset loss and testing dataset loss for 10 epochs

(b) Testing dataset accuracy after each epoch

Figure 5: MLP model on CIFAR-10 dataset

2.3 CNN model

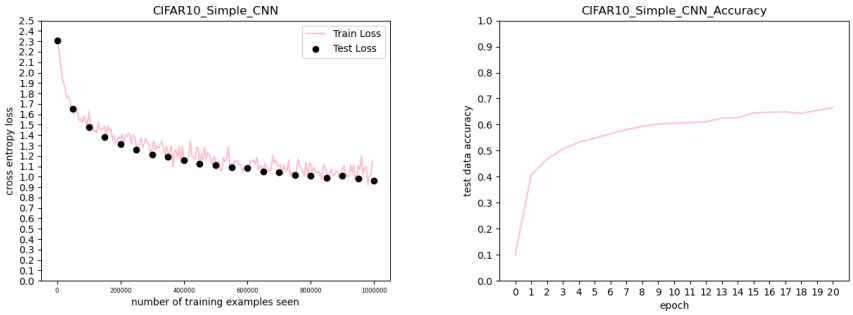
In this section, I built two CNN models to train the CIFAR-10 dataset.

2.3.1 A simple CNN model

This simple CNN architecture has an input layer with a $32*32*3$ dimension and two convolution layers. The first convolutional layer expects 3 input channels and convolves 6 filters each of size $5 * 5 * 3$. Then I used max-pooling with a $2 * 2$ kernel and a stride of 2. I added a dropout layer after that. The second convolutional layer expects 6 input channels and convolves 16 filters each of size $5 * 5 * 6$, following a max-pooling layer with a $2 * 2$ kernels and a stride of 2. After that, I added a dropout layer.

Then I flattened the tensors which have $5 * 5 * 16 = 400$ features, and put them into a dense layer, pass through an MLP to carry out the task of classification of 10 categories. I chose Adam as the optimizer and set the weight decay as 0.0005. I trained the model for 20 epochs and achieved around 67% testing

data accuracy at the end. Figure 6a shows the training loss and the testing loss change over time. Figure 6b shows the testing data accuracy changes over time.



(a) Training dataset loss and testing dataset loss for 20 epochs

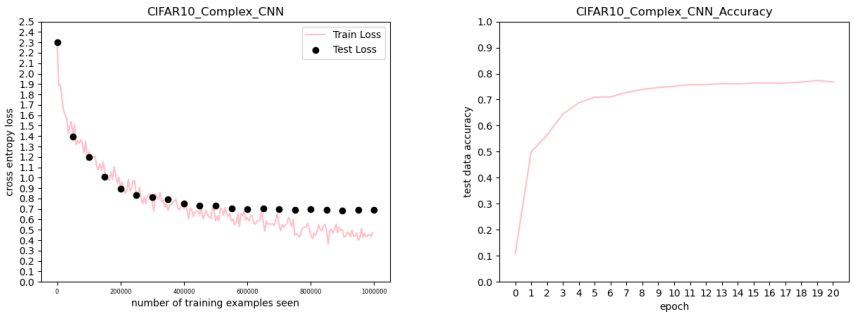
(b) Testing dataset accuracy after each epoch

Figure 6: Simple CNN model on CIFAR-10 dataset

2.3.2 A complicated CNN model

This complicated CNN architecture has an input layer with a $32 * 32 * 3$ dimension and five convolution layers. The first convolutional layer expects 3 input channels and convolves 32 filters each of size $3*3*3$. Then I applied a batch normalization and a dropout layer. The second convolutional layer expects 32 input channels and convolves 48 filters each of size $3 * 3 * 32$, following a batch normalization and a dropout layer. The third convolutional layer expects 48 input channels and convolves 64 filters each of size $3 * 3 * 48$, following a batch normalization and a dropout layer. The fourth convolutional layer expects 64 input channels and convolves 128 filters each of size $3 * 3 * 64$. After that, I applied batch normalization, a max-pooling layer with window size $2 * 2$ and a stride of 2 and a dropout layer. The fifth convolutional layer expects 128 input channels and convolves 256 filters each of size $3 * 3 * 128$. After that, I applied batch normalization, a max-pooling layer with window size $2 * 2$ and a stride of 2 and a dropout layer.

Then I flattened the tensors which have $5*5*256 = 6400$ features, and put them into a dense layer, pass through an MLP to carry out the task of classification of 10 categories. I chose Adam as the optimizer and set the weight decay as 0.0005. I trained the model for 20 epochs and achieved around 77% testing data accuracy at the end. Figure 7a shows the training loss and the testing loss change over time. Figure 7b shows the testing data accuracy changes over time.



(a) Training dataset loss and testing dataset loss for 20 epochs

(b) Testing dataset accuracy after each epoch

Figure 7: Complex CNN model on CIFAR-10 dataset

3 Conclusion

In conclusion, I believe CNN is the best model to classify images compared with Soft-max regression and MLP. Besides, the MNIST dataset is only $28 * 28$ px in size, I believe a simple CNN works well. However, a complicated CNN is necessary to train the CIFAR-10 dataset. In this assignment, I built two CNN models for the CIFAR-10 dataset. The simple CNN model can achieve around 67% accuracy for the testing dataset and the complicated CNN model can achieve around 77% accuracy for the testing dataset. It shows that the testing dataset accuracy improves. However, overfitting is a problem for the complicated CNN model even though I tried adding weight decay and dropout layer to avoid it.