

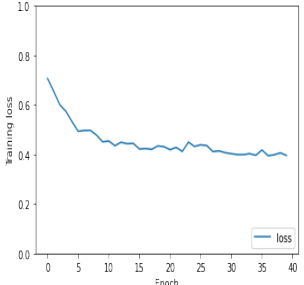
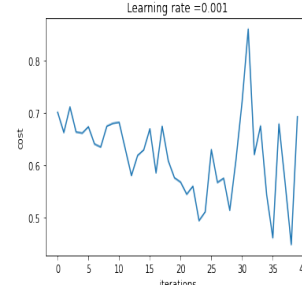
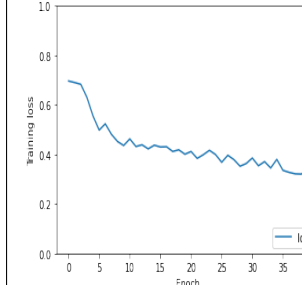
Convolutional Neural Network

Tsung-Yen Yang, 108022138

December 14, 2022

1 Comparison

The table below shows the performance and relative parameters of three models, which are linear model, my CNN model and Tensorflow CNN model respectively.

	linear model	my CNN model	Tensorflow CNN model
epochs	40	40	40
training time	14.68 sec	39 min	59.08 sec
training accuracy	0.8431	0.7764	0.8706
validation accuracy	0.7778	0.8000	0.8000
num of parameters	31,391	4,817	278,289
batch size	24	24	24
training loss curve			

Based on the table, Tensorflow CNN model reaches the highest accuracy on both training and validation set and also takes less time on training (similar with linear model). However, my CNN model is not only time consuming but also has a less stable training loss curve compared to the other two models.

2 Tensorflow CNN Model

The model architecture I designed is shown below in form of code.

```
2 model = models.Sequential()
3 model.add(layers.Conv2D(16, kernel_size=(3, 3), padding='same', activation='relu', input_shape=(32, 32, 1)))
4 model.add(layers.MaxPool2D(pool_size=(2, 2)))
5 model.add(layers.Conv2D(16, kernel_size=(3, 3), padding='same', activation='relu'))
6 model.add(layers.MaxPool2D(pool_size=(2, 2)))
7 model.add(layers.Conv2D(64, kernel_size=(3, 3), padding='same', activation='relu'))
8 model.add(layers.MaxPool2D(pool_size=(2, 2)))
9 model.add(layers.Dropout(0.25))
10
11 model.add(layers.Flatten())
12 model.add(layers.Dense(256, activation='relu'))
13 model.add(layers.Dense(16, activation='relu'))
14 model.add(layers.Dropout(0.25))
15 model.add(layers.Dense(1, activation='sigmoid'))
```

Figure 1: The model is mainly composed of three 3x3 convolution layers, 2x2 max pooling layers and fully connected neural network with two hidden layers. The activation functions used are all relu function except for the last layer. Besides, there are also two drop out layers with drop out ratio 0.25 to deal with potential overfitting issue.

As for optimizer and loss function, I respectively choose Adam as optimizer and binary cross entropy as loss function. For my model, Adam could really help accelerate the whole training process and improve the validation accuracy compared to the other optimizers such as SGD and Adagrade. Besides, I also import the early stopping callback function from keras to monitor the validation loss to prevent overfitting.