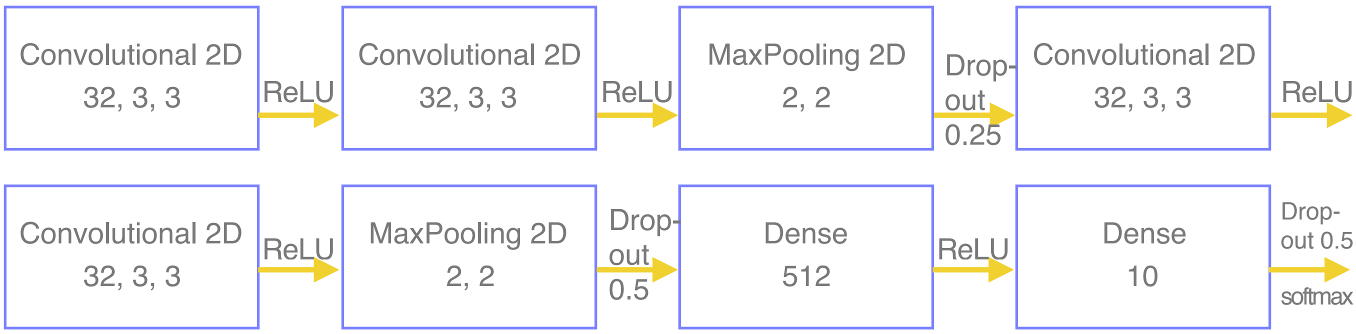
Machine Learning HW3

b03901057 詹晉誠

Q1. Supervised Learning

The network structure used in this problem is described below:



The dropout technique is used to in order to learn a more generalised model. Also, the activation function of the hidden layers were chosen to be ReLU since sigmoid tends to cause the vanishing gradient problem. In this method, only the labeled data (5000 samples) were used for training, and the performance of this method is shown below :

Validation Accuracy	Kaggle Score
0.6873	0.539

Notice that, 500 randomly chosen samples out of the 5000 labeled data were used as the validation set. The Kaggle score is merely above 50%, which is a little bit higher than the Kaggle public set baseline.

Q2. Semi-supervised Learning Method1

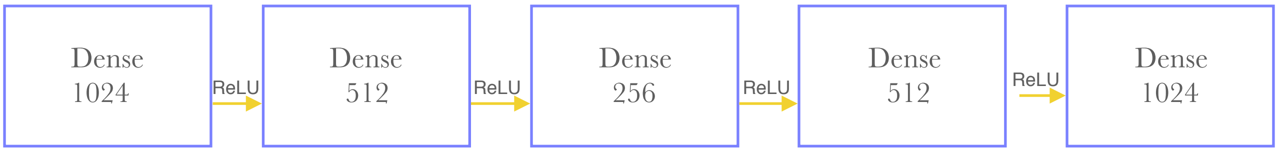
In this method, I implemented self-training. More specifically, 1) I first train the CNN model, whose architecture will be discussed later, with the labeled data. Later, 2) I repetitively use the trained model to make predictions on the unlabelled data. 3) The most confident ones (with probability >0.98) are added to the labeled data. Then I repeat step 2) and 3) until either all the unlabelled data (excluding test data) are added or the number of iteration reaches a predefined value. The architecture of the CNN is similar to that in Q1, except that two Convolutional2D Layers with parameter (128, 3, 3) were inserted after the second max-pooling layer. Also, data generation technique is employed to increase the prediction accuracy. I simply use image random shifting and flipping as the data generation technique. The performance of this method is briefly listed below:

Training Accuracy	Validation Accuracy	Kaggle Score
0.94	0.64	0.630

Q3. Semi-supervised Learning Method 2

In method 2, I implemented an autoencoder to extract features (code) from the data and then use fully-connected layers to predict which class an image belongs to according to the extracted feature (code). The structures are displayed below:

Autoencoder:



The neural network for prediction is rather simple. It consists of a Dense 1024 layer followed by a Dense 10 layer. Activation functions are ReLU and sigmoid respectively. The Kaggle score of this method is only 0.27. The reason could be that deep autoencoder is used here instead of convolutional autoencoder. For tasks like image classification, a better way is to use convolutional autoencoder.

Q4. Compare and analyse the results:

1. **Supervised vs semi-supervised:**

There is an about 10% accuracy gain after adopting the semi-supervised method, since we leverage the unlabelled data in semi-supervised learning. In supervised learning, I train the model for 50 epochs, while in semi-supervised learning I trained the model with the labeled data for only 10 epochs. The reason is that our labelled data may be biased, so instead of learning a model that explains the unbiased data very well, we want a more generalised model to make predictions on the unlabelled data. According to my experiments, reducing the number of epochs of the training on labelled data from 50 to 10 increases the performance.

2. **Overfitting problem of semi-supervised learning method 1:**

Observing the Training Accuracy and the Kaggle Score, we notice a huge difference between the two numbers. It means that there is a severe overfitting problem in my model. The overfitting problem could be caused by spending too many epochs to train the model on the labelled data. However, 10 epochs were used in my experiment, so this might not be the true reason. It is also possible that huge differences between these two values is a common problem in semi-supervised learning since now we only have a small fraction of data being labelled. To alleviate the problem, the data generation technique may be used, yet, the gain in accuracy is not phenomenal in my experiments.

3. **Accuracy of semi-supervised learning method 1:**

The reported accuracy of method 1 is about 0.63, which is not very good. Some possible improvement of the method includes early stopping and more sophisticated data generation techniques. For example, variational autoencoder is a DNN-based technique, and is widely used for the data generation purpose. Better data generation methods may add some variations to the data and thus prevent overfitting.