ML HW3 Report

B03902082 資工三 江懿友

1. Supervised Learning

I use Keras' ImageDataGenerator to augment the labeled data by randomly mirror flip / rotate / jitter the labeled images, and use the augmented labeled data to train my CNN. The structure of my CNN is:

Input(dim = 3x32x32) -> Conv(32x3x3) -> relu -> Conv(32x3x3) -> relu -> MaxPool(2x2) -> Dropout(0.25) -> Conv(64x3x3) -> relu -> MaxPool(2x2) -> Dropout(0.25) -> Flatten() -> FullyConnected(256) -> relu -> FullyConnected(256) -> relu -> Dropout(0.25) -> FullyConnected(10) -> Softmax -> Output(dim = 10)

The result on public set (accuracy): 0.65240

2. Semi-supervised Learning

I use the trained CNN from "Supervised Learning" to predict the unlabeled data set, then for each data I have a probability distribution of it. I calculate the probability distribution's entropy and filter out the data whose entropy is lower than a threshold. These are the data that I have enough "confidence" with. Thus I add these filtered data with their predicted label into the training set, and retrain the CNN.

The result: 0.70900

3. Semi-supervised Learning Method 2

First I use the labeled and unlabeled image data to train an autoencoder, then I encode the labeled data and use the encoded data and their label to train a SVM. Next I use the trained SVM to classify the unlabeled data and use their classified result as their label. Then I use all the data to train the previous CNN.

The result: 0.29400

4. Compare

(2) performs slightly better than (1), because (2) has much more data to train the CNN than (1). But the improvement isn't huge because the initial CNN is not good enough and it has many mispredictions, so the training data may have some wrong labels in them.

(1) and (2) outperforms (3) because the autoencoder's performance is bad. Thus the SVM classifier aren't working well, too. So there must be a lot of wrongly labeled data in the training set.