**Report of Natural Language Processing, hw2**

109550027紀竺均

1. Describe how you build your model ? How did you do to preprocess your data from dataset ? The distribution of the emotion is unbalance, what did you do to improve the accuracy on those emotion which are in small scale?(30%)

Ans:

- Model:

Use Pytorch to build a multi-layer long short-term memory (LSTM) RNN. Parameters of LSTM:

dimension\_model = 128

num\_layers = 5

hidden\_size = 60

linear\_hidden\_size = 30

dropout = 0.2

- Data preprocess:

(1) Use collections.Counter and nltk.tokenize.word\_tokenize to calculate the frequency of each word in training data set.

(2) Remove low frequency word ( exist = 1 )

(3) Encode each utterance into a vector of indexes that represent the words.

- Deal with unbalance data:

Use K-fold cross validation with k=5 to repeatedly resampling the data and introduced randomness to the dataset.

def cross\_val(i):

    batch\_size = 32

    train\_x = total\_x[kfold\_data[i][0]]

    train\_y = total\_y[kfold\_data[i][0]]

    dev\_x = total\_x[kfold\_data[i][1]]

    dev\_y = total\_y[kfold\_data[i][1]]

    train\_ds = TensorDataset(torch.from\_numpy(train\_x), torch.from\_numpy(train\_y))

    dev\_ds = TensorDataset(torch.from\_numpy(dev\_x), torch.from\_numpy(dev\_y))

    train\_dl = DataLoader(train\_ds, shuffle=True, batch\_size=batch\_size, drop\_last=True)

    dev\_dl = DataLoader(dev\_ds, shuffle=True, batch\_size=batch\_size, drop\_last=True)

    return train\_dl, dev\_dl

2. Have you tried pretrain word embedding?( e.g. Glove or Word2vec).What is the influence of the result after you using them?(30%)

3. Have you tried attention on your model? What is the influence of the result after you using them? Which text your model attention on when it predict the emotion?(30%)

4.Have you used other information form dataset to improve your model performance?(e.g. Speaker) What is the influence of the result after you using them?(10%)

Ans: Yes, I add speaker into