1. Model description

Rnn

```
model.add(Masking(mask_value=0, input_shape=(mfcc_train.shape[1], mfcc_train.shape[2])))
model.add(LSTM(CELL_SIZE,
               batch_input_shape=(None, mfcc_train.shape[1], mfcc_train.shape[2]),
               stateful=False,
               dropout=0.2,
               return_sequences=True))
model.add(BatchNormalization())
model.add(Dense(units=OUTPUT_SIZE, activation='softmax'))
```

在 Rnn model 我選擇 LSTM (長短期記憶層)。一開始我選擇對資料做 padding，訓練效果表面很好，但實際丟 kaggle 分數卻很低(Overfitting)。我想是因為 padding 加入過多假的 training data (all features =0 and label = 49) 造成訓練很吻合這些特徵全為 0 的 training data。 在 LSTM 層前加入 masking layer，避免此情況發生。最後在 kaggle 上的分數為：

Cnn + Rnn

```
model = Sequential()
model.add(Conv1D(CELL_SIZE,
                 kernel_size=8,
                 input_shape=(mfcc_train.shape[1], mfcc_train.shape[2]),
                 padding='causal',
                 activation='relu'))
model.add(BatchNormalization())
model.add(Masking(mask_value=0, input_shape=(mfcc_train.shape[1], mfcc_train.shape[2])))
model.add(LSTM(CELL_SIZE,
               stateful=False,
               dropout=0.2,
               return_sequences=True))
model.add(BatchNormalization())
model.add(Dense(units=OUTPUT_SIZE, activation='softmax'))

optimizer = RMSprop(lr=0.0005)
model.compile(loss='categorical_crossentropy',
              optimizer=optimizer,
              metrics=['accuracy'])
```

在 Cnn model 我選用一維的卷積層，因為我有做 padding，所以每一段音頻被我擴展成長度 777 的 frame，而每個 frame 中有 39 維 feature (我使用 mfcc)

接下來也是照 Rnn model 作法，也需要先放一層 Masking layer 避免訓訓練會太吻合假 training data。

2. How to improve your performance

a. 使用 Padding 技巧

b. 在處理資料時，使用 Padding 的方式擴展每一段的音頻的 frame 個數到一模一樣的值，在這裡我將每個音頻的frame都擴展到有777個（因為某個音頻的frame有 777 個，為最長的 frame 數，所以以他為上限做 padding ，並且使用 Rnn model，把 Time Step設計為 777 ，別且將 return_sequence 設為 True，讓每個 step 都會輸出結果。

c. 資料不做任何處理的話丟進去 Rnn 做 training，由於有 Time step 的參數，但每個音頻的 frame 數都不同，則音頻仍會去參考其他音頻的訓練資料，一開始我對資料沒有做這些處理，訓練效果就滿差的，丟上去 kaggle 都只有20幾。因此我把每一段音頻都擴增到相同長度，這樣才能使他們不跨音頻做 training 效果就好了很多。

3. Experimental results and settings

在這次作業中，我深刻體會到 network 架構設計的重要性，我一開始嘗試好幾次調整 batch size 或是 epoch 數量，但是分數總是在 20 分左右徘徊，後來開始認真思考架構後才有突飛猛進的改進。另外，處理 data 也是很重要，讓我體會了適時 padding data 的重要性。以下列出幾次我的架構與資料處理對於結果的影響之實驗：

- 一開始未做 Padding:

- 做了 Padding 但沒有加入 masking layer:

最後加上 Padding 與 masking layer 就能得到至少過 baseline 的分數。