Data preprocessing:

- 使用 get_data() return train_vector, train_vector_label, vocabulary。
- get_data()裡面重要的 variable:

variable 例子(句子)

1. train_sentence: [['I','am','a','pig'],['HAHA', '<padding>', '<padding>', '<padding>'],['Yes','you','are', '<padding>']]
   - 這邊我預先看了一下 train,validation data 裡面最常句子為 41。
   - 我在訓練時將字母都設為小寫。
2. train_vector: [[0,1,2,5],[29,101,101,101],[33,8,100,101]]
3. train_sentence_label: [['B-person','O','O','B-other'],['B-other', 'O', 'O', 'O'],['O','B-person','O', 'O']]
4. train_vector_label: [[0,20,20,18],[18,20,20,20],[20,18,20,20]]
5. vocabulary: set('I','am','a',.....,'are', '<padding>', '<url>', '<user>', '<unk>')
   - 這邊我在將原始資料中@.....轉為'<user>'，http:轉為'<url>'。
   - 這邊我把 train+validationa 資料中的單字都加到 vocabulary。
   - 最後總共大小為 7530
6. idx_to_word: sorted(list(vocabulary))
7. word_to_idx: {'I':0,'am':1, 'a':2,.....,'are':7}
8. idx_to_label: ['B-person', 'I-person',.....,'O']
9. label_to_idx: {'B-person':0, 'I-person':1,.....,'O':20}

- 此 function 最後是要用來得到 train_vector, train_vector_label。

Model Design:

- input data x: train_vector
- input data label y: train_vector_label
- architecture: x ---> Embedding layer ---> Lstm ---> linear layer ---> softamx
  1. Embedding layer: num_embeddings= 7530, embedding_dim= 100
  2. Lstm: 2 layer, hidden_size= 128, bidirectional, dropout rate=0.5。
  3. Linear layer:輸出轉為要分類的大小(21 類)

Training Process:

- batch_size = 128
- learning rate=0.01
- epochs=1000
- loss function: CrossEntropyLoss(reduction='none')
  1. 特別設了 none 因為想說原本資料 O 的太多，想要再算 Loss 時把他的權重設低一點，每一個 O 算出來的 loss 我都把它減一半。

```
model_pred = model(x)     # model_pred.shape: batch*seq_len*n_class
# 讓 O label的loss比較小
loss = loss_fun(model_pred.permute(0, 2, 1), y)
for i in range(len(y)):
    for j in range(41):
        if y[i][j]==20:weight_loss+=(0.5*loss[i][j])
        else:weight_loss+=loss[i][j]
weight_loss/=(256*41)
temp.append(weight_loss)
weight_loss.backward()
optimizer.step()
```

- optimizer: Adam
- validation:每 train 完一個 epoch 都存在 validation set 上 f1 score 較高的那個模型，最後留下在 validation set 上 f1 score 最高的那個。

evaluation scores:使用 sklearn.metric.f1_score 在 validation set 上測出來的結果:

- B-person: 0.273
- I-person: 0.255
- B-geo-loc: 0.326
- I-geo-loc: 0.231
- B-company: 0.296
- I-company:0
- B-facility: 0.267
- I-facility: 0.290
- B-product:0
- I-product:0
- B-musicartist: 0.323
- I-musicartist: 0.143
- B-movie: 0.027
- I-movie: 0.235
- B-sportsteam:0
- I-sportsteam: 0.237
- B- tvshow: 0.137
- I-tvshow: validation data 沒有這類
- Other: 0.987