Question 4: Evaluation Metrics

Let me compare the output of **conlleval.pl** and **main.py** for **dev** for the **CRF tagger** on the **base features**.

Output from Main.py

```
### evaluation of data/twitter_dev.ner; writing to ./twitter_dev.ner.pred
Token-wise accuracy 95.7701308832
Token-wise F1 (macro) 29.5648858833
Token-wise F1 (micro) 95.7701308832
Sentence-wise accuracy 68.6440677966
              precision    recall  f1-score   support

    B-company       0.88      0.39      0.54        36
   B-facility       0.57      0.43      0.49        28
    B-geo-loc       0.70      0.36      0.48        77
      B-movie       0.00      0.00      0.00         7
B-musicartist       0.50      0.08      0.13        13
      B-other       0.62      0.13      0.21        63
     B-person       0.62      0.35      0.45       108
    B-product       0.60      0.16      0.25        19
 B-sportsteam       0.50      0.09      0.15        11
     B-tvshow       0.50      0.18      0.27        11
    I-company       0.00      0.00      0.00         7
   I-facility       0.65      0.45      0.53        29
    I-geo-loc       0.50      0.14      0.22        14
      I-movie       0.00      0.00      0.00        11
I-musicartist       1.00      0.07      0.12        15
      I-other       0.65      0.16      0.26        81
     I-person       0.58      0.34      0.43        61
    I-product       0.80      0.25      0.38        16
 I-sportsteam       0.00      0.00      0.00         4
     I-tvshow       0.67      0.20      0.31        10
            O       0.96      1.00      0.98     10916

  avg / total       0.95      0.96      0.95     11537
```

Output from conlleval.pl

```
guest-wireless-207-151-035-009:Homework-6 RishabhTyagi$ data/conlleval.pl -d \\t < twitter_dev.ner.pred
processed 11537 tokens with 373 phrases; found: 165 phrases; correct: 100.
accuracy:  95.77%; precision:  60.61%; recall:  26.81%; FB1:  37.17
          company: precision:  87.50%; recall:  38.89%; FB1:  53.85  16
         facility: precision:  47.62%; recall:  35.71%; FB1:  40.82  21
          geo-loc: precision:  70.00%; recall:  36.36%; FB1:  47.86  40
            movie: precision:   0.00%; recall:   0.00%; FB1:   0.00  1
      musicartist: precision:   0.00%; recall:   0.00%; FB1:   0.00  2
            other: precision:  53.85%; recall:  11.11%; FB1:  18.42  13
           person: precision:  57.38%; recall:  32.41%; FB1:  41.42  61
          product: precision:  60.00%; recall:  15.79%; FB1:  25.00  5
       sportsteam: precision:  50.00%; recall:   9.09%; FB1:  15.38  2
           tvshow: precision:  50.00%; recall:  18.18%; FB1:  26.67  4
```

Comparing both the outputs, first thing we see is that, main.py evaluates the precision, recall and f1 score for both the beginning and inside tags for each category.
On the other hand, conlleval.pl evaluates precision, recall and f1 score for categories without considering the subcategories B-<category_name> and I-<category_name>. This means that if can find the named entity of beginning and inside of the phrase correctly, only then the phrase is marked correct for f1 scores and accuracies. While main.py predicts the phrase as correct or incorrect based on subcategories i.e. like beginning of person and inside of person.
Therefore, inorder to see if our features are working correctly or not, we should use main.py for evaluation as it provides us with detailed evaluation on sub categories like B-<category_name> and I-<category_name> and we can figure out whether it was predicted correctly or not.
But to find the overall performance we should use **conlleval.pl** because it tells us whether we have named an entity/complete phrase correctly or not

Also, keeping a thought for conlleval.py, if we need to check for overall performance of the system, we need to find how well our features and tagger performed to perform named entity recognition for the entire token.
Therefore, conlleval.pl should be considered as better metric in Named entity recognition problem.
e.g. In the case of Sherlock Holmes, Correct prediction is Sherlock (B-person) Holmes(I-person). Let's us say, our model predicts Sherlock correctly but predicts Holmes incorrectly, then by evaluating on main.py, we can find this out that Sherlock is correct and Holmes is incorrect. Overall accuracy in main.py takes Sherlock being predicted correctly and Holmes being predicted incorrectly, but it is not right as overall NER is wrong for the word Sherlock Holmes. But in the case of conlleval – the evaluation says this is incorrectly named entity. This way it is difficult to check for the entire BIO encoding unless it is completely right.
But if it is wrong, it should not contribute to the accuracy, therefore conlleval is better performance indicator.