CSCI 544 HW6

Qifan Wang

9369314001

1.1.

I implemented a bunch of features, first let me explain some basic improvement.

```
##    if word.lower() in Nwordlist:
##        ftrs.append("IS_COMMON")
    ftrs.append("PTAG_" + ptags[i][1])

    if len(word) == 4 and word.strip(':-,().').isnumeric():
        ftrs.append("MAYBE_YEAR")
    if len(word) == 0:
        ftrs.append("IS_SPACE")
    if len(word) > 8:
        ftrs.append("IS_LONGER.")
    if len(word) > 0 and word[0] == '#':
        ftrs.append("IS_HASHTAG")
    if len(word) > 0 and word[0] == '@':
        ftrs.append("IS_TWITTERTAG")
    if len(word) > 0 and word[0].isupper():
        ftrs.append("IS_HEAD")

    ftrs.append("FIRST_TRI" + word[:3])
    ftrs.append("LAST_TRI" + word[-3:])

ptags = nltk.pos_tag(sent)
for i in xrange(len(sent)):
    print sent[i], ":", token2features(sent, i, ptags)
```

here are some minor improvements I have implemented,:

year identify

space identify

longer word identify

hashtag, twitter-tag, http identify (Since a lot of twitter use

these tags to post something and some of them are links)

head of a sentence identify

some partial word letter identify

POS tag identify

```
QifandeMacBook-Pro:hw6_og AlwaysBeBetter$ perl data/conlleval.pl -d \\t < twitter_dev_test.ner.pred
processed 11308 tokens with 644 phrases; found: 170 phrases; correct: 55.
accuracy:  91.02%; precision:  32.35%; recall:   8.54%; FB1:  13.51
          company: precision:  72.73%; recall:   7.34%; FB1:  13.33  11
         facility: precision:   4.00%; recall:   2.17%; FB1:   2.82  25
          geo-loc: precision:  58.21%; recall:  24.53%; FB1:  34.51  67
            movie: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
      musicartist: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
            other: precision:   0.00%; recall:   0.00%; FB1:   0.00  17
           person: precision:  14.29%; recall:   7.29%; FB1:   9.66  49
          product: precision:   0.00%; recall:   0.00%; FB1:   0.00  1
       sportsteam: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
           tvshow: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
```

The original logistic regression on the test file gives about

13.51 score.

```
QifandeMBP:Homework-6 AlwaysBeBetter$ perl data/conlleval.pl -d \\t < twitter_dev_test.ner.pred
processed 11308 tokens with 644 phrases; found: 271 phrases; correct: 76.
accuracy:  91.24%; precision:  28.04%; recall:  11.80%; FB1:  16.61
          company: precision:  62.50%; recall:   9.17%; FB1:  16.00  16
         facility: precision:   8.57%; recall:   6.52%; FB1:   7.41  35
          geo-loc: precision:  56.25%; recall:  28.30%; FB1:  37.66  80
            movie: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
      musicartist: precision:   0.00%; recall:   0.00%; FB1:   0.00  1
            other: precision:   0.00%; recall:   0.00%; FB1:   0.00  42
           person: precision:  19.15%; recall:  18.75%; FB1:  18.95  94
          product: precision:   0.00%; recall:   0.00%; FB1:   0.00  3
       sportsteam: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
           tvshow: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
```

And after I add these feature, it improve up to 16.61,

especially when I added word[:3] and word[-3:], it goes up

quite a bit.

Also, the a very important part lexicons joined after I

implement these basic feature improvement.

```
def preprocess_corpus(train_sents):
    """Use the sentences to do whatever preprocessing you t
    such as counts, keeping track of rare features/words to
    loading files, and so on. Avoid doing any of this in to
    that will be called on every token of every sentence.

    Of course, this is an optional function.

    Note that you can also call token2features here to aggr
    """
    path = "./data/lexicon"
    for fn in os.listdir(path):
        with open(path + '/' + fn) as fnn:
            fName[fn] = set(line.strip() for line in fnn)

if add_neighs:
    for f, values in fName.iteritems():
        if word in values:
            ftrs.append("TYPE_" + f)
```

first of all, I just achieve some very simple goals, which are

searching words inside the lexicons dictionary, if it exist,

then tag added.

And here is what the result improvement looks like.

```
QifandeMBP:Homework-6 AlwaysBeBetter$ perl data/conlleval.pl -d \\t < twitter_dev_test.ner.pred
processed 11308 tokens with 644 phrases; found: 331 phrases; correct: 106.
accuracy:  91.66%; precision:  32.02%; recall:  16.46%; FB1:  21.74
          company: precision:  80.00%; recall:  11.01%; FB1:  19.35  15
         facility: precision:  10.81%; recall:   8.70%; FB1:   9.64  37
          geo-loc: precision:  67.39%; recall:  38.99%; FB1:  49.40  92
            movie: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
      musicartist: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
            other: precision:   2.38%; recall:   0.85%; FB1:   1.25  42
           person: precision:  19.57%; recall:  28.12%; FB1:  23.08  138
          product: precision:   0.00%; recall:   0.00%; FB1:   0.00  7
       sportsteam: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
            tvshow: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
```

from 16 up to 21 almost 22.

However, I realize that there it might not be so simple to use lexicons, it would be too rough. So I choose to use some of them, such as last-name.5k, sport team and so on. But the result turned out is worse. After few tries, I chose to not modify them.