Middle East Technical University | Department of Computer Engineering

CENG 563

Computational Linguistics

Spring '2013-2014

Assignment 3 and Take home final

Due date: 6 June 2014, Friday, 17:30

1 Objectives

In this take home final and assignment combination, you are expected to implement a part of speech tagger for Turkish. You will experiment with three different tag sets of your choice. You will be asked to convert a given morphological analysis into a part of speech tag and tag a given sentence in Turkish. You are also expected to write a report on your part of speech tags and implementation details.

Keywords: *Part of Speech Tagging, Tag-set selection, Viterbi Algorithm, Hidden Markov Models*

2 Part of Speech Tagging

A part of speech of a word is the lexical category of the word (also word class, morphological class, lexical class, or lexical tag). A part of speech of a word provides information about a word and its neighbours, word's pronunciation, its morphological analysis or can be used in speech recognition, parsing, translation, information retrieval and question answering tasks. Many NLP applications may benefit from syntactically disambiguated text. Part of speech tagging is the process of assigning tags (class markers) to words. A word can be a member of multiple classes and main task of a tagger is to solve this ambiguity. Successful approaches can solve over 95% of these ambiguities. There are many machine learning algorithms applied to the tagging problem and they can achieve over 95% accuracy. However even a 1% accuracy matters. For example, a tagger with 98% overall accuracy, has 74% chance of tagging all the words correctly in a 15-word sentence, a tagger with 97% overall accuracy has only 63% chance. Your performance depends considerably on the amount of training data, your tag set, your dictionary and unknown words.

Please study existing part of speech tags. You have to build an appropriate part of speech tag set or modify an existing set according to your needs. There are several examples for different POS tag sets

sorted according to the level of information they contain in the examples, therefore, you also have to decide on your POS tags. Your report should discuss the methods you used and the assumptions you made.

2.1 Stochastic Part of Speech Tagging

Hidden Markov Models are used to tag sequence of words. There are several algorithms for learning and decoding for HMMs. Baum-Welch and Viterbi Algorithms are commonly used for these problems. HMMs model word sequences (sentences) as emissions of tags which depend only on the previous ones. These are called Markov chains. You have labelled data at your disposal, so choose the appropriate training algorithm.

3 Specifications

1. Name your modules pos_tagger_1, pos_tagger_2 and pos_tagger_3

2. You will implement two functions in each module.
   One will have single morphological analysis and return a (word,tag) pair.
   The other will have single sentence and return (word,tag) pairs of the sentence.

```
convert(analysis)
tag(sentence)
```

3. You are expected to train your system with the training file provided ans tune it using the development set. Another test file will be used in grading.

4. Your modules will be tested with an additional test.py file which imports your modules and calls the implemented functions.

   Test system:

```
>>> from pos_tagger_1 import convert , tag
>>> # Same format as in the development files .
... convert('okuduğum (oku)oku +Verb+Pos(+dHk)^DB+Adj+PastPart(+Hm)+P1sg') (↩
    'okuduğum','Verb_Adj')
>>> # Tokens will be separated by space .
 ... tag('Ve hüzün yüzündeki kırışıklıkları biraz daha oyarak derinleş↩
     tiriyor .')
[('Ve','Conj'),('hüzün','Noun_Nom'),('yüzündeki','Adj_Noun'),('kırışıklıklar↩
    ı','Noun_Acc'),('biraz','Adj'),('daha','Adv'),('oyarak','Adv_Verb'),('↩
    derinleştiriyor','Verb_Adj'),('.','Punc')]
```

5. For each one of your three experiments, you should clearly explain your choice of tags. Your report should include strengths and weaknesses of each tag set and their comparison. Try to explain the sequences that your models capture easily. Analyse your errors. Build a confusion matrix to show which tags are hard to catch.

4 Example Tags

Listing 1: Morphological analysis

```
filmin ( film ) film+Noun+A3sg+Pnon(+nHn )+Gen
en ( en ) en+Adverb
can ( can ) can+Noun+A3sg+Pnon+Nom
alıcı ( al ) al+Verb+Pos(+yHcH ) ^DB+Adj+Agt
noktası ( nokta ) nokta+Noun+A3sg(+sH )+P3sg+Nom
işlenmiş (iSle)işle+Verb(+Hn)^DB+Verb+Pass+Pos(+mHS)^DB+Adj+NarrPart
olan ( ol ) ol+Verb+Pos(+yAn ) ^DB+Adj+PresPart
konunun ( konu ) konu+Noun+A3sg+Pnon(+nHn )+Gen
işleniş (iSle)işle+Verb(+Hn)^DB+Verb+Pass+Pos(+yHS)^DB+Noun+Inf3+A3sg+Pnon+Nom
şekli ( Sekil ) şekil+Noun+A3sg(+sH )+P3sg+Nom
```

Listing 2: A simple tag set

```
filmin Noun
en Adverb
can Noun
alıcı Adj
noktası Noun
işlenmiş Adj
olan Adj
konunun Noun
işleniş Noun
şekli Noun
```

Listing 3: Another tag set that covers case information for nouns

```
filmin Noun+Gen
en Adverb
can Noun+Nom
alıcı Adj
noktası Noun+Nom
işlenmiş Adj
olan Adj
konunun Noun+Gen
işleniş Noun+Nom
şekli Noun+Nom
```

Listing 4: These tags also have stem information for explaining the sequence

```
filmin Noun+Gen
en Adverb
can Noun+Nom
alıcı Verb+Adj
noktası Noun+Nom
işlenmiş Verb+Adj
olan Verb+Adj
konunun Noun+Gen
işleniş Verb+Noun+Nom
şekli Noun+Nom
```

5 Regulations

1. **Programming Language:** You will use Natural Language Toolkit with Python language. HMM libraries are forbidden. You should implement necessary methods yourselves.

2. **Collaboration:** Not allowed. Standard code of honour applies

3. **Submission** will be done via COW.

4. **Evaluation** Your Part of Speech Taggers will be evaluated on a test set which contains different sentences from your development set. Your program will be evaluated automatically using 'black-box' technique so make sure to obey the specifications. Your taggers will be tested with an additional test.py file which imports your module and calls your implemented functions.

```
>>> from pos_tagger_3 import tag
>>> tag('Bunu başından beri biliyordum zaten .')
[('Bunu','Pron'),('başından','Noun_Abl'),('beri','Postp'),('biliyordum','↩
    Verb'),('zaten','Adv'),(' .','Punc')]
```

6 References

- Jurafsky, Daniel, and James H. Martin. 2009. Speech and Language Processing: An Introduction to Natural Language Processing, Speech Recognition, and Computational Linguistics. 2nd edition. Prentice-Hall.
- Christopher D. Manning and Hinrich Schütze. 1999. Foundations of Statistical Natural Language Processing. MIT Press, Cambridge, MA, USA.
- Steven Bird, Ewan Klein, and Edward Loper. 2009. Natural Language Processing with Python. 1st edition. O'Reilly Media, Inc.