Assignment 4: Natural Language Processing

Objectives

The objectives of this assignment are to:

- Get acquainted with an annotated corpus
- Extract bigram statistics from this corpus
- Implement a baseline part-of-speech tagger
- Implement the Viterbi algorithm
- Implement a part-of-speech tagger using hidden Markov models

Organization

Each student will have to:

- Write a baseline part-of-speech tagger
- Write a part-of-speech tagger using hidden Markov models
- Evaluate the results on a corpus and comment them briefly

Programming

This assignment will use a corpus from the shared task of the conference on computational natural language learning, CoNLL. CoNLL corpora consist of columns to store the words and their syntactic and semantic properties.

The assignment corpus follows the CoNLL 2009 variant. Read a description of it here. In our corpus, two columns contain parts-of-speech tags: POS and PPOS. POS is the part of speech given manually by the annotators while PPOS is the part of speech predicted by an automatic POS tagger. This PPOS column is given by the CoNLL organizers to produce the subsequent columns of the file. They are out of the scope of the assignment. Your tagger will produce such a PPOS column.

Corpus processing

1. Write a program to read the corpus and extract the list of distinct words and their frequencies (number of occurrences) from the training set.
2. Extract the list of distinct parts of speech (POS) and their frequencies (number of occurrences) from the training set.

Evaluation program

1. Write an evaluation program that computes the per-word accuracy of a tagger.
2. Apply your evaluation program to compute the accuracy of the tags in the PPOS column: The ratio of predicted parts of speech (PPOS column) matching the manually assigned parts of speech (POS column) divided by the number of POS. You will use the development set to compute this accuracy.
3. Use the POS and PPOS columns to compute the confusion matrix of a POS tagger.

Baseline tagger

1. Write a program that for each word extracts its most frequent part of speech. You will extract these pairs, (word, part of speech), from the training corpus (Form and POS columns). Use the pairs to write a tagger
2. Apply your tagger to the development set.
3. Evaluate the performance of your tagger using your evaluation program.

POS tagger using hidden Markov models

1. Extract all the POS bigrams and estimate $P(t_i|t_{i-1})$.
2. For all the relevant pairs, extract and estimate $P(w_i|t_i)$.
3. Write a tagger to apply the noisy channel model to a sentence of length *n*.
4. Apply your tagger to the development set with *n* as a parameter. This means that you will process the sentences that have a length less than *n* using the naïve version of the tagger. What is the maximal value of *n* for which you can get results on your machine?
5. Implement the Viterbi algorithm.
6. Evaluate the performance of your tagger on the development set using your evaluation program.
7. Apply your tagger to the test set.

Remarks

Deadline

The report must be handed in for evaluation before 23.59 on Monday, March 11, 2013. The report should be e-mailed to tai @ cs.lth.se with the subject line Assignment X by usernames.

Problems

In case of problems, send an e-mail to Pierre.Nugues@cs.lth.se.

Report

The assignment must be documented in a report, which should contain the following:

- The name of the authors, the title of the assignment, and any relevant information on the front page.
- A presentation of the assignment.
- A presentation of your implementation and how to run the executable.
- A print-out of the results on the development set.
- Comments on the results you have achieved.

You need also to hand in the tagged version of the test set.

Please, typeset and format your report consistently using Latex for instance.

Programming Language and Environment

You may use these languages: Java, C, C++, Prolog, Perl, or Python, to develop your program. Should you want to use another language, contact me before.

Your final program must be available and runnable on the LINUX computers at the *.student.lth.se address (e.g. login.student.lth.se). Remember to make your programs and all the directories in their path read and execute accessible to 'others' (chmod 705 filename). Remember also to quote where does your solution reside and how should it be run (kind of "User's Guide").

The resulting programs should remain in your directory until you have been notified of the result, e.g. on the notice board and/or web or by e-mail. You may expect that your report and implementation will be examined within two weeks. If your report or implementation is unsatisfactory you will be given one chance to make the corrections and then to hand it in within a week after you have been notified (on the notice board and/or web or by e-mail).

Last updated: 2013-12-13