ANU 2016

COMP4650/6490: Document Analysis

Information Extraction (IE) Assignment

Main Details

Maximum marks: 10

Programming language: Java (only)

Assignment questions: Post to the Wattle Discussion forum

Deadline:

1. Programming part: Graded in ***your lab*** of the week of Mon 10 Oct 2016 lab (automatic 0 for this part if fail to attend/demonstrate in the lab)
2. Written part: Thu 13 Oct, 23:59 on Wattle

Marking Scheme:

- Programming part: Full marks given for working, readable, reasonably efficient, commented code that performs well on the test case given in lab.

- Written part: Full marks given for a formulation that provides a well-reasoned and succinct response to the question that addresses all requested points. There may be more than one answer for each question that achieves full marks.

- Academic Misconduct Policy: All submitted written work and code must be your own (except for any provided Java starter code, of course) – submitting work other than your own will lead to both a failure on the assignment and a referral of the case to the ANU academic misconduct review procedures: ANU Academic Misconduct Procedures

Electronic Submission (only):

- Written part: All answers should be in a PDF file, titled ANSWERS.pdf. MS Word or other document formats are not accepted. LATEX formatting is preferred.
- Please submit ANSWERS.pdf and the files requested in Q1 zipped into a single ZIP archive IEASSIGNMENT_YourlastnameYourfirstname.zip (e.g., IEASSIGNMENT_SuominenHanna.zip) with a README.txt explaining directories and documents you have included in your submission.

Information Extraction: Programming Part

Q1 [5 pts]. CFF++ for NER

The task is to code a Named Entity Recognizer (NER) application in Java using the CRF++ tool and Conll2002 data sets. The CRF++ software and Conll2002 data sets are in Topic 7, Information Extraction, Lab and Assignment resources posted to Wattle.

First, build a NER model on the esp.train set as instructed by Steps 1-6 on the lecture slides (IE Lecture 1). Remember to include your CRF++ template as template.txt document in your submission for the written part together with a document called template_explanation.txt that explains what your template makes the CRF++ to do.

Second, write a Java application that uses your NER model to label a new test set as instructed by Steps 7-9 on the lecture slides (IE Lecture 1). Remember the requirement to have the F1 > 70 % for each entity class, except for the MISC class on the esp.testb set. You can use the esp.testa set to tune your classifier (e.g., to set a good *c* value).

Third, write a Java NER application that uses your NER model. Prepare your application for texting on a new test set (we will provide this in the grading lab). Your NER application must include an extractor that displays all unique recognized named entities (one entity per line) and their frequencies after 1) organising them alphabetically first with respect to the NER tag and then with respect to the extracted text, 2) counting the frequency of each unique NER-text line, and 3) removing duplicates (see the example/NE-ExtractorFormat.txt document).

Submit all your processing code (Java) as part of the written part.

Grading: Java code (1 pt), template and its explanation (2pt), the F1 performance on the esp.testb set (1 pt), and the correctness of the output format on the new test set provided in the grading lab (1 pt)

Information Extraction: Written Part

For this section, simply submit your answers in your ANSWERS.pdf document.

Q2 [1 pt]. Effect of the training size on test set

Measure the *effect of training size* on the NER performance on the test set by using 1/4, 2/4, 3/4, and 4/4 of the training sentences classifier using the conll-eval.pl script and the training and test sets of the programming part. Please plot the evaluation results as a *learning curve* and use it to derive a conclusion.

Q3 [1 pt]. Re-substitution performance

Explain what is *resubstitution performance* and use it to supplement your NER performance evaluation. What additional insight have you gained about the *c* value of the CRF++?

Q4 [3 pt]. HMM for POS tagging

Imagine you wanted to develop a POS tagger using a HMM and Finnish text, where each word has been associated with precisely one out of the following six labels: *noun*, *adjective*, *pronoun*, *numeral*, *particle*, *verb*, and *other*. What are the hidden states of your HMM and what could you use as observations (1 pt)? What do the emission probabilities refer to in this case (1 pt)?

In Finnish, adjectives that define a noun tend to occur before the noun and in Portuguese, the opposite is usually true. Would you expect to see a smaller transition probability from an adjective to a noun in Finnish than in Portuguese, and why? (1 pt)