

**LING 570: Hw1**  
**Due date: 11:45pm on Oct 7, 2009**

For this homework, you are going to write an English tokenizer and a tool that creates a vocabulary from the input text. All the sample files are under **~/dropbox/09-10/570/hw1/examples/**.

For this assignment, you can assume that a token will not cross the line boundary; that is, you can process the input text line-by-line.

**Q1 (40 points):** Implementing an English tokenizer, **eng\_tokenizer.sh**

- Format:
  - The command line is: `cat input_file | ./eng_tokenizer.sh {arguments} > output_file`
  - Your code can take additional arguments, such as a file with common abbreviations. In that case, in your hw1.\* note file, explain the content and format of the arguments.
  - The input and output files should have the same line number, and the i-th line in the input corresponds to the i-th line in the out file.
  - The tokens in the output lines should be separated by the whitespace.
  - A sample input file is “ex1”, and a sample output file is “ex1.tok”. The sample output file is meant to show you the format, NOT the correct answers.
- Note:
  - Your code should process each line independently of other lines.
  - Do not merge the tokens in the input text (e.g., the collocation expression such as “pick up”, “because of”, “Hong Kong” should not be merged into one token).
  - **Don't spend too much time trying to make your tokenizer perfect.**

**Q2 (20 points):** In your note file, write a short description of your tokenization algorithm, which includes the following:

- Explain your tokenization criteria: e.g., how are hyphenated words, abbreviation, telephone numbers and urls treated
- Discuss remaining problems (if any) in your tokenizer and how you believe they can be addressed.
- Does your tokenizer use any resource (e.g., word lists)? If so, what are they?

**Q3 (10 points):** Writing a tool, **make\_voc.sh**, that creates a vocabulary from the input text.

- The command line should be: `cat input_file | ./make_voc.sh > output_file`

- The tool reads in each line in the input, breaks it into tokens by whitespace, and output the frequencies of the tokens.
- Each line in the output file is a (token, frequency) pair. The lines are sorted by the frequency of the tokens in descending order.
- A sample input is “ex1”, and a sample output is “ex1.voc”.

**Q4 (10 points):** Run the code in Q1 and Q3

- Run the following commands:
  - `cat ex1 | ./eng_tokenizer.sh > ex1.tok`
  - `cat ex1.tok | ./make_voc.sh > ex1.tok.voc`
  - `cat ex1 | ./make_voc.sh > ex1.voc`
- In your note file, write down
  - the numbers of tokens in ex1 and ex1.tok
  - the sizes of ex1.voc and ex1.tok.voc

**Q5 (20 points):** If you bet a 7 on a roulette wheel, there is a probability of  $1/38$  of winning. Assume bets are placed on the number 7 in each of 500 different spins. What is the probability of winning exactly 13 times? Write down the formulas and the final answer.

- Use the binomial distribution to solve the problem
- Use the Poisson distribution to solve the problem

The submission should include:

- The answers to Q2, Q4, and Q5
- The source code: `eng_tokenizer.*`, `make_voc.*`
- The shell scripts: `eng_tokenizer.sh`, `make_voc.sh`
- If your tokenizer uses any additional files (e.g., word list), submit those files and write down the command line for running your tokenizer
- The files created by Q4: `ex1.voc`, `ex1.tok.voc`, and `ex1.voc`