Exercise 4: N-grams

You can earn up to 19 points on this exercise.
You may work as a group of up to 3 people, but please submit your own version.
You may use any programming language you wish, but any submission that we cannot run on our computers without installing things must be presented to the class.

Please email your solution to `mittul.singh@lsv.uni-saarland.de` *or submit before the tutorial by* **10 am, November 25, 2015**. If you wish to submit your assignment by email, please name the file as Ex4_ <*your name*>.pdf . Some of the following tasks have been blatantly copied from Statistical Machine Translation book available here: `http://www.statmt.org/book/`

Task 1

The training data is given in Table 0.1 and 0.2.

Count	Count of counts
1	5000
2	1600
3	800
4	500
5	300

Table 0.1: Count of counts statistics

Count	Bigram
4	beer drinker
4	beer lover
2	beer glass

Table 0.2: The word beer occurs as history in three bigrams in the data

1. What are the adjusted counts under Good-Turing discounting for the three given bigrams? (3 points)
2. The left over probability mass after summing the adjusted prediction model is given to a back-off unigram model. Using such a back-off model, what are the probabilities for the following bigrams? (3 points)
   i p(drinker|beer)
   ii p(glass|beer)
   iii p(mug|beer)

Note: p(mug) = 0.01. State any assumptions that you make.

Task 2

This exercise is to get you familiar with using a popular LM toolkit. Download and install the SRILM[1] . SRILM is a great free language modelling toolkit for doing interpolated modified Kneser-Ney-smoothed n-gram language models. It has other smoothing techniques with it but today we will only use Modified Kneser-Ney. (10 points)

[1] `http://www.speech.sri.com/projects/srilm/download.html`

1. Download the "English News Crawl (articles from 2007)" corpus from: `http://www.statmt.org/wmt14/training-monolingual-news-crawl/news.2007.en.shuffled.gz`

2. Use only the first 100,000 lines, and tokenize the resulting smaller corpus, using a simple tokenizer found at `https://github.com/jonsafari/tok-tok`.

3. Convert the text to lowercase. You can use the script `http://jon.dehdari.org/corpus_tools/lowercase.pl`. You can use the following Unix commands to do the above instructions:

   ```
   zcat corpus.gz | head -n 100000 | ./tok-tok.pl | ./lowercase.pl > en.txt
   ```

   If you don't understand the above Unix command, ask someone who does. Report the number of words in the corpus en.txt. (3 points)

4. Split the resulting subcorpus into training/development/test sets, with the ratio 18:1:1 respectively, using the script `http://jon.dehdari.org/corpus_tools/generate_splits.pl` . Use the –help argument for usage info.

5. Train SRILM interpolated modified kenser-ney model (use flags $-interpolate$ and $-kndiscount$ on the training set. Report training times and submit the language model file generated. In Unix environments, you can get time info by prepending "time" before a command. (2 points)

6. Report both probabilities and $log10$ probabilities of each of the first 8 words in the first sentence of the dev set, up to "candidates". You can get word-level info for SRILM by adding the "-debug 2" argument. How does each LM software handle unseen (OOV) words, as well as </s> ? Briefly discuss. (3 points)

7. Report perplexity (PPL) on the entire test set, including OOV words (use $-unk$ flag to train the model) and not including them. (2 points)

Bonus

Given that we add α to each n-gram count c as described by *Add-α Smoothing*, derive the probability of a bigram under this smoothing scheme:

$$p(w_2|w_1) = \frac{c + \alpha}{N + \alpha v^2}$$

where, N is the number of bigrams in the corpus and v is the vocabulary size. (3 points)