UNIVERSITÄT DES SAARLANDES

Statistical Natural Language Processing

Summer Semester 2019
Prof. Dr. Dietrich Klakow

Exercise Sheet V

Submission Deadline: May 31th, 23:59

1 Long-Range Dependencies

Exercise 1.1: Correlation Function (10 points)

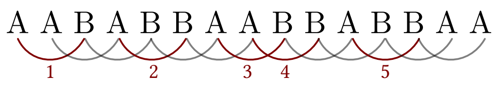
Words in natural languages exhibit both short-range local dependencies as well as long-range global dependencies to various degrees. To analyse these dependencies, the temporal correlation function can be used. The correlation between two words w_1 and w_2 is defined as

$$Corr_D(w_1, w_2) = \frac{\mathbf{P}_D(w_1, w_2)}{\mathbf{P}(w_1)\mathbf{P}(w_2)}$$

where $\mathbf{P}_D(w_1, w_2)$ is the probability of observing w_2 after w_1 separated by a distance D, estimated from a (continuous) text corpus, and $\mathbf{P}(w_1)$ and $\mathbf{P}(w_2)$ are the unigram probabilities of w_1 and w_2, respectively, estimated from the same text corpus. The term $\mathbf{P}_D(w_1, w_2)$ can be defined as

$$\mathbf{P}_D(w_1, w_2) = \frac{N_D(w_1, w_2)}{N - D}$$

where $N_D(w1, w2)$ is the total number of times w_2 was observed after w_1 at distance D and N is the total number of word tokens in the text corpus. A simplified example is illustrated in the figure below.



$$\mathbf{P}_2(w_1 = A, w_2 = B) = \frac{N_2(A, B)}{N - 2} = \frac{5}{13}$$

In this exercise, you will implement and use the correlation function to analyze word dependencies in an English corpus. To this end, extract the text file `continuous.corpus.en` from `exercise5_corpora`, ignore sentence boundaries and apply whitespace-based tokenization using the regex-based tokenizer provided in a previous exercise sheet. Proceed as follows

(a) If the occurrence of word w_2 is always independent of preceding occurrences of word w_1, what is value of $Corr_D(w_1, w_2)$ in this case?

(b) Implement a function `correlation` that takes two words, w_1 and w_2, and distance D as inputs and returns as an output the correlation value.

(c) Using the `correlation` function of part (b), and for each $D \in \{1, 2, 3, \ldots, 100\}$, compute the correlation values of each of the following word pairs

$w_1, w_2 =$ `'you', 'your'`	$w_1, w_2 =$ `'he', 'his'`	$w_1, w_2 =$ `'he', 'her'`
$w_1, w_2 =$ `'she', 'her'`	$w_1, w_2 =$ `'they', 'their'`	$w_1, w_2 =$ `'she', 'his'`

(d) Apply moving average (a.k.a running mean) with window size 5 on the data points you obtained in part (c). Generate a linear curve where the x-axis represents the distance D

and the y-axis represents the value of the correlation. Use a legend to identify each word pair on the graph. In addition, your graph should show the expected behavior if word occurrences were statistically independent. Explain your findings.

(e) Repeat part (d) but with the x-axis in a logarithmic scale.

Your solution should include two figures; part (d) and part (e), a sufficient explanation of the findings in this exercise, as well as the source code to reproduce the results. Moreover, feel free to include other interesting word pairs in your analysis.

2 Backing-off Language Models

Exercise 2.1: LM Smoothing with Absolute Discounting (10 points)

In this exercise, you will implement absolute discounting smoothing for a bigram language model with a back-off strategy. Backing-off LMs alleviate the sparsity problem with natural language word sequences by interpolating higher- and lower-order n-gram models. When the higher-order n-gram model is a bigram LM, absolute discounting is defined as

$$\mathbf{P}_{abs}(w_i|w_{i-1}) = \frac{\max\{N(w_{i-1}w_i) - d, 0\}}{\sum_{w' \in \mathcal{V}} N(w_{i-1}w')} + \lambda(w_{i-1})\mathbf{P}_{abs}(w_i)$$

$$\mathbf{P}_{abs}(w_i) = \frac{\max\{N(w_i) - d, 0\}}{\sum_{w' \in \mathcal{V}} N(w')} + \lambda(.)\mathbf{P}_{unif}(w_i)$$

where $0 < d < 1$ is the discounting parameter, $\mathbf{P}_{unif}$ is a uniform distribution over the vocabulary $\mathcal{V}$, $\lambda(w_{i-1})$ and $\lambda(.)$ are the backing-off factors defined as

$$\lambda(w_{i-1}) = \frac{d}{\sum_{w' \in \mathcal{V}} N(w_{i-1}w')} N_{1+}(w_{i-1} \bullet) \quad \text{where} \quad N_{1+}(w_{i-1} \bullet) = |\{w : N(w_{i-1}w) > 0\}|$$

and

$$\lambda(.) = \frac{d}{\sum_{w' \in \mathcal{V}} N(w')} N_{1+} \quad \text{where} \quad N_{1+} = |\{w : N(w) > 0\}|$$

(a) With the notations used for the bigram LM above, write a generic recursive definition of an absolute discounting LM of any n-gram level. Explain what would be the base case of the recursion and the backing-off factor.

(b) Given the equations above, implement absolute discounting smoothing and build a bigram LM using the `corpus.sent.en.train` corpus. Your implementation of this exercise can be an extension of the `ngram_LM` class, or your could design your own code from scratch if that is more convenient for you. If you choose to extend the existing code, the main adaptation should be on the `estimate_smoothed_prob` function. Make sure your implementation is tested with suitable test cases.

(c) Compute perplexity of the two test corpora `simple.test` and `wiki.test` with $d = 0.5$. Compare the perplexity values of the absolute discounting LM with those observed by the best-performing bigram LM in the previous exercise sheet. Briefly explain your findings.

(d) Investigate the effect of the discounting parameter d on the perplexity of the two test corpora. For each $d \in \{0.1, 0.2, \ldots, 0.9\}$ compute the perplexity and plot a line chart where the x-axis corresponds to the discounting parameter and the y-axis corresponds to the perplexity value.

(e) How does the absolute discounting-based LM differ from the Lidstone smoothing-based LM developed in the previous exercise sheet? How would you explain the difference in the performance of the language models built with those two techniques?

Your solution should include a clear/clean mathematical definition of part (a), a table of part (c), one/two graphs of part (d), as well as the source code to reproduce the results.

Exercise 2.2: Out-of-vocabulary (OOV) Words (Extra credits)

In this exercise you will use the multilingual corpus in `exercise1_corpora` to investigate the relation between the vocabulary and the OOV rate in different languages. For each text corpus, lower-case the text, remove non-alphabetical characters, and apply whitespace-based tokenization.

(a) Partition each corpus into a training corpus (80% of the word tokens) and a test corpus (20% of the word tokens).

(b) Construct a vocabulary for each language by taking the most frequent $15k$ word types in the training corpus. The vocabulary set should be ranked by frequency from the most frequent to the least frequent.

(c) Compute OOV rate (percentage) on the test corpus as the vocabulary grows by $1k$ words.

(d) For each language, plot a logarithmic curve where the x-axis represents the size of the vocabulary and the y-axis represents the OOV rate. Each curve should be plotted on the same figure with a legend to identify the language of the text corpus (similar to SNLP chapter 5 slides, page 8). Write your observations.

(e) Which of the languages in this exercise would be challenging for statistical language modeling? Justify your answer.

Your solution should include the graph of part (d) and the source code to reproduce the results. Depending on the clarity and the novelty of your presentation, you may earn up to 5 points for this question.

Submission Instructions

The following instructions are mandatory. Please read them carefully. If you do not follow these instructions, the tutors can decide not to correct your exercise solutions.

- You have to submit the solutions of this exercise sheet as a team of 2-3 students.
- NLTK modules are not allowed, and not necessary, for this assignment.
- You do not need to include the distributed corpora within your submission.
- Make a single `ZIP` archive file of your solution with the following structure
  - A `source_code` directory that contains your well-documented source code and a `README` file with instructions to run the code and reproduce the results.
  - A `PDF` report with your solutions, figures, and discussions on the questions that you would like to include.
  - A `README` file with group member names, matriculation numbers and emails.
- Rename your `ZIP` submission file in the format

  `exercise05_id#1_id#2_id#3.zip`

  where `id#n` is the matriculation number of every member in the team.

- Your exercise solution must be uploaded by only one of your team members to the course management system (CMS). Once the grading is done, your assignment grade will be distributed to each team member by one of the tutors.
- If you have any problems with the submission, contact `babdullah@lsv.uni-saarland.de` before the deadline.