Language Technology - EDAN20
Assignment 2 - Fall 2020

Language Models
n-gram statistics - Probability of a Sentence

Hicham Mohamad, hi8826mo-s
hsmo@kth.se

August 30, 2020

1 Objectives

The objectives of this assignment are to:

- Write a program to find n-gram statistics
- Compute the probability of a sentence
- Know what a language model is
- Write a short report of 1 to 2 pages on the assignment
- Optionally read a short article on the importance of corpora

2 Organization and location

The second lab session will take place on September 17 and 18.

There can be last minute changes. Please always check the official times here:

`https://cloud.timeedit.net/lu/web/lth1/ri1Q5006.html`

You can work alone or collaborate with another student:

- Each group will have to write Python programs to count unigrams, bigrams, and trigrams in a corpus of approximately one million words and to determine the probability of a sentence.
- You can test you regular expression using the `regex101.com` site
- Each student will have to write a short report of one to two pages and comment briefly the results. In your report, you must produce the tabulated results of your analysis as described below.

3 Programming

Collecting a corpus

1. Retrieve a corpus of novels by Selma Lagerlöf from this URL:

   `https://github.com/pnugues/ilppp/blob/master/programs/corpus/Selma.txt` . The text of these novels was extracted from Lagerlöf arkivet at Litteraturbanken.

2. Alternatively, you can collect a corpus of at least 750,000 words. You will check the number of words using the Unix command `wc -w`.

3. Run the concordance program to print the lines containing a specific word, for instance *Nils*.

4. Run the tokenization program on your corpus and count the words using the Unix `sort` and `uniq` commands.

Normalizing a corpus

1. Write a program to insert `<s>` and `</s>` tags to delimit sentences. You can start from the **tokenization** and modify it. Use a simple heuristics such as: a sentence starts with a capital letter and ends with a period. Estimate roughly the accuracy of your program.

2. Modify your program to remove the punctuation signs and set all the text in lower case letters.

3. The result should be a normalized text without punctuation signs where all the sentences are delimited with `<s>` and `</s>` tags.

4. The five last lines of the text should look like this:

<s> hon hade åftt östrre äkrlek av sina öäfrldrar än ångon annan han visste och åsdan äkrlek åmste ävndas i ävlsignelse </s>
<s> åd äprsten sade detta kom alla ämnniskor att se bort mot klara gulla och de öfrundrade sig över vad de åsg </s>
<s> äprstens ord tycktes redan ha ågtt i uppfyllelse </s>
<s> ädr stod klara fina gulleborg åifrn skrolycka hon som var uppkallad efter äsjlva solen vid sina öäfrldrars grav och lyste som en öfrklarad </s>
<s> hon var ålikas vacker som den ösndagen åd hon gick till kyrkan i den örda äklnningen om inte vackrare </s>

Counting unigrams and bigrams

1. Read and try programs to compute the frequency of unigrams and bigrams of the training set: [Program folder].

2. What is the possible number of bigrams and their real number? Explain why such a difference. What would be the possible number of 4-grams.

3. Propose a solution to cope with bigrams unseen in the corpus. This topic will be discussed during the lab session.

Computing the likelihood of a sentence

1. Write a program to compute a sentence's probability using unigrams. You may find useful the dictionaries that we saw in the mutual information program: [Program folder].

2. Write a program to compute the sentence probability using bigrams. Select five sentences in your test set and run

3. your programs on them.

4. Tabulate your results as in the examples below with the sentence *Det var en gång en katt som hette Nils*:

Unigram model

wi	C(wi)	#words	P(wi)
det	21108	1041631	0.0202643738521607
var	12090	1041631	0.01160679741674355
en	13514	1041631	0.012973884225795 89å
gng	1332	1041631	0.001278763784871994
en	13514	1041631	0.012973884225795 89

```
katt      16      1041631      1.5360525944408337e-05
som       16288     1041631      0.015637015411407686
hette     97      1041631      9.312318853797554e-05
nils      87      1041631      8.352285982272032e-05
</s>      59047     1041631      0.056687060964967444
```

```
Prob. unigrams:    5.361459667285409e-27
Geometric mean prob.: 0.0023600885848765307
Entropy rate:    8.726943273141258
Perplexity:    423.71290908655254
```

Bigram model

wi	wi+1	Ci,i+1	C(i)	P(wi+1|wi)
<s>	**det**	5672	59047	0.09605907158704083
det	var	3839	21108	0.1818741709304529
var	en	712	12090	0.058891645988420185
en å	gng	706	13514	0.052242119283705785å
gng	en	20	1332	0.015015015015015015
en	katt	6	13514	0.0004439840165754033
katt	som	2	16	0.125
som	hette	45	16288	0.002762770137524558
hette	nils	0	97	0.0 *backoff: 8.352285982272032e-05
nils	</s>	2	87	0.022988505747126436

```
Prob. bigrams:    2.376007803503683e-19
Geometric mean prob.: 0.013727289294133601
Entropy rate:    6.186809422848149
Perplexity:    72.84759420254609
```

4 Reading

As an application of n-grams, execute the Jupyter notebook by Peter Norvig here. Just run all the cells and be sure that you understand the code. You will find the data here.

In your report, you will also describe one experiment with a long string of words your will create yourself or copy from a text you like. You will remove all the punctuation and white spaces from this string. Set this string in lower-case letters.

You will just add a cell at the end of Sect. 7 in Norvig's notebook, where you will use your string and run the notebook cell with the `segment()` and `segment2()` functions.

You will comment the segmentation results you obtain with unigram and bigram models.

5 Complement

As a complement, you can read a paper by Church and Hanks, *Word Association Norms, Mutual Information, and Lexicography*, **Computational Linguistics**, 16(1):22-29, 1990, as well as another one on **Backoff** by Brants et al. (2007) *Large language models in machine translation*.

6 Appendix

References

[1] Pierre Nugues, *Language processing with Perl and Prolog*, 2nd edition, 2014, Springer.

[2] P. Nugues, Language Technology - EDAN20, Lectures notes: https://cs.lth.se/edan20/

[3] D. Jurafsky, J. Martin, *Speech and Language Processing*, 3rd edition. Online: https://web.stanford.edu/~jurafsky/slp3/

[4] Pierre Nugues, Assignment 2: https://github.com/pnugues/edan20/blob/master/notebooks/2-language_models.ipynb

[5] Peter Norvig, *How To Do Things With Words and Counters*, jupyter notebook: http://nbviewer.jupyter.org/url/norvig.com/ipython/How%20to%20Do%20Things%20with%20Words.ipynb

[6] VanderPlas, A Whirlwind Tour of Python. O'Reilly 2016. Online reading: https://jakevdp.github.io/WhirlwindTourOfPython/

[7] Geron, Jupyter notebook on linear algebra from Machine Learning and Deep Learning in Python using Scikit-Learn, Keras and TensorFlow: https://github.com/ageron/handson-ml2