Prof. Dr. Andreas Hotho,
M.Sc. Janna Omeliyanenko
Lecture Chair X for Data Science, Universität Würzburg

1. Project task for "Sprachverarbeitung und Text Mining"

05.11.2021

General rules

Over the course of the semester, you will be given three different project assignments, each with its own issue and submission deadline. You must pass the required control tests for at least two of the three assignments in order to receive a grade bonus of a full grade step (e.g. from 1.7 to 1.3) for the written exam at the end of the semester. The grade bonus only applies if you pass the written exam. What is necessary to pass each assignment is announced individually for each assignment.
All tasks must be solved in **Python 3**. Other programming languages are not allowed. Solutions will be submitted via WueCampus2.
Collaboration of a maximum of two people is allowed and encouraged. In this case, please include a .txt file with the names and matriculation numbers of both participants in the .zip file to be submitted (see below). Participation in the project tasks is not compulsory, but highly recommended.

Note: For programming exercises, you have free access to the computer science CIP pools A001 and A002 in the computer science building M2.

Project 1: Multiword expressions through significantly frequent tri-grams

Within the PUNKT sentence splitting algorithm in lecture 02, you learned about the likelihood-ratio test to determine statistically significant frequent bigrams of a text[1]. Extend this method to tri-grams, to detect multiword expressions consisting of three words within a given text corpus. Multiword expressions denote combinations of words that build a joint meaning beyond word boundaries, (e.g. 'fish and chips', 'by the numbers', ...). To accomplish your goal, proceed as follows:

1. Download the files `static_idioms.txt`, `formal_idioms.txt` and `stopwords.txt` from the WueCampus2 course.
2. Implement a python program, which reads a .txt file, transfers the Likelihood-Ratio test from the lecture from bi-grams to tri-grams, and returns the 20 most statistically significant frequent tri-grams as console output.

   Use a tokeniser that removes words which contain language-dependent special characters such as such as ß or é. All words should be converted to lower case. Punctuation marks, as well as all words that are in the given stopword list `stopwords.txt`, should be removed.[2]

   Note: Assume $log(0) = 0$ for the implementation.

   Your application must be implemented in such a way that it can be started with only one command line command, which only receives the path to a .txt file as an input parameter (e.g. `python3 foo.py static_idioms.txt`. You may assume that the file `stopwords.txt` is in the execution directory of the script.

   You can use the file `static_idioms.txt` to test your application. The sample solution for this file is printed in Listing 1 for comparison. Your output can of course be formatted differently, but should roughly follow the scheme `(tri-gram, log-likelihood, counts)`.
3. Upload your solution to WueCampus2 until **26.11.2021 at 12pm CET**. Pack everything that belongs to your submission into a single .zip file.

For the implementation of the statistical procedure, **no** frameworks that offer this functionality may be used. This is exactly the part you are supposed to practise! However, frameworks are allowed for tokenisation and counting the bi-grams and tri-grams. Keep

[1] Ted Dunning, Accurate Methods for the Statistics of Surprise and Coincidence, `http://aclweb.org/anthology/J93-1003`

[2] E.g. the sentences "I told my fiancé to go stand in the corner when she is cold. Its usually about 90°." should be tokenized to: ['told', 'go', 'stand', 'corner', 'when', 'cold', 'usually']

in mind that your submission must run ”out of the box”. You can only assume that the Python module `nltk` is available on the target machine.

Your submitted program will be compared against a reference implementation on a test dataset that we do not publish. If the result matches (a certain tolerance is taken into account here), the task is rated as ”passed”.

Obligatory note:
Detected plagiarism will be assessed as ”failed” and, in the worst case, can lead directly to the exclusion from the exam bonus.

Listing 1: Sample solution static_idioms.txt

```
((tri-gram), -2log(lambda), counts)
(('behind', 'closed', 'doors'), (1786.4336871514977, 109, 131, 108))
(('new', 'lease', 'life'), (1447.065491334064, 116, 621, 116))
(('dream', 'come', 'true'), (1331.6474551958834, 90, 199, 89))
(('level', 'playing', 'field'), (1084.1241907127226, 69, 128, 68))
(('nine', 'times', 'ten'), (1016.5943763877704, 73, 181, 69))
(('breath', 'fresh', 'air'), (886.5670528244891, 65, 347, 65))
(('tour', 'de', 'force'), (875.2450608671977, 57, 115, 55))
(('easier', 'said', 'done'), (740.1051493823943, 51, 229, 51))
(('find', 'way', 'around'), (618.2756174133447, 56, 597, 52))
(('difference', 'life', 'death'), (594.840526659164, 40, 190, 40))
(('something', 'along', 'lines'), (575.1996518437709, 44, 439, 44))
(('well', 'never', 'mind'), (535.470165717792, 46, 696, 45))
(('back', 'drawing', 'board'), (505.138911056315, 38, 294, 37))
(('given', 'new', 'lease'), (418.16447654096373, 28, 120, 27))
(('well', 'fair', 'enough'), (406.1820311931333, 32, 520, 32))
(('clean', 'bill', 'health'), (390.7978555317336, 24, 96, 24))
(('enough', 'fair', 'enough'), (367.92861626315516, 29, 520, 29))
(('oh', 'never', 'mind'), (362.73147184220215, 30, 696, 30))
(('right', 'across', 'board'), (360.27725120259674, 33, 294, 28))
(('ill', 'gotten', 'gains'), (354.78391685014384, 18, 25, 18))
```