# **Mini Project 2:** Unsupervised learning of a polarity lexicon

## 1 Goal

The goal of this project is to automatically predict the polarity of words. Consider the two sentences

1. *In a few words, the cd is excellent and is a musical treasure for any salsa lover.*
2. *This is a shamefully poor piece.*
3. *This piece is excellent, an absolute treasure.*

Pretend that you do not know the polarity of any English words but *excellent* and *poor*. In that case, you would not know that the word *treasure* is positive (sentence 1) or that *shamefully* is negative (sentence 2). However, you may hypothesize that words that frequently occur close to *excellent* are positive, and words frequently close to *poor* are negative. In this project, you will implement an algorithm that exploits this information.

## 2 Method

We follow the basic idea by Turney (2002). He proposes to measure the degree of association between words by using pointwise mutual information (PMI), as discussed in class. Turney uses a search engine to determine the following counts (on the web):

1. hits(*poor*): number of times the word *poor* occurs
2. hits(*excellent*): number of times the *excellent* occurs
3. hits($X$ NEAR *poor*): number of times the word $X$ occurs close to *poor*
4. hits($X$ NEAR *excellent*): number of times the word $X$ occurs close to *excellent*

Turney then calculates the overall polarity of a word as

$$\mathrm{SO}(X) = \log \frac{\mathrm{hits}(X \text{ NEAR } \textit{excellent}) \cdot \mathrm{hits}(\textit{poor})}{\mathrm{hits}(X \text{ NEAR } \textit{poor}) \cdot \mathrm{hits}(\textit{excellent})}$$

This value will be larger than 0 if the word is positive, and smaller than 0 if it is negative. As zero hit counts are problematic, Turney adds 0.01 to all counts before calculating the fraction.

Unfortunately, autmatically sending such queries to a search engine is not easily done anymore today. However, with a large enough corpus, we can simulate the experiment offline. Given such a corpus, we can easily obtain the counts for 1. and 2. by checking how often the words *poor* and *excellent* occur in it. To obtain counts 3. and 4., we have to check the surroundings of each of the aforementioned occurrences. We simply look to the left and to the right of each *excellent* and *poor* within a certain window (say, 5 words) and count for these words how often they occurred in such a window.

As an example, if our corpus consisted only of the three sentences above, *treasure* would occur twice with *excellent* and zero times with *poor*. We also find that *poor* occurs once while *excellent* occurs twice. Given these counts, we could then calculate the PMI score

$$\mathrm{SO}(\textit{treasure}) = \log \frac{2.01 \cdot 1.01}{0.01 \cdot 2.01} = \log 101 \approx 2.0$$

As the score is positive, our prediction is that *treasure* is positive.

# 3 Your task

You will write a program that predicts the polarity for a collection of words as shown above. We will use the one by Hu and Liu (2004), available at

`http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar`

This dataset consists of two files, `positive-words.txt` and `negative-words.txt`, containing several thousand positive and negative words, respectively. The files also contain some header information that you have to remove or ignore when processing.

You will now write a program that first computes the four types of counts shown above. For that, you need a large corpus. You can use the one provided on Ilias (`sample.preprocessed.txt`). This is a corpus of random Amazon reviews. Each line contains a single review. The text is already tokenized and lowercased.

Next, your program needs to read the polarity lexicon and calculate the SO score for each word. Check if the prediction is correct and calculate accuracy. You can expect an accuracy of 65% or more.

# 4 Submission

For this project, you need to submit to me the following:

- Your working code
- Your predictions (i.e., the new polarity lexicon)
- A short report (1-2 pages) describing the results you obtained
  - Overall accuracy
  - Error analysis of two words where your program made a mistake – show why it went wrong. You have to look into the corpus for this.

# References

Hu, M., Liu, B., 2004. Mining and summarizing customer reviews. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp. 168–177.

Turney, P., July 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In: Proceedings of 40th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pp. 417–424.
URL `http://www.aclweb.org/anthology/P02-1053`