WS 2017 Project 2: Stance Detection in Tweets
Due 17-03-2017, 23h55

Mareike Hartmann

1 Stance Detection

Stance detection is the task of automatically determining if an author is in favor or against a given target. See the following examples example given by Mohammad et al. 2016:

> Target: legalization of abortion
> Tweet: *A foetus has rights too! Make your voice heard.*

Here, we can infer that the author's stance is against the target (against the legalization of abortion). A stance classifier should classify this tweet as AGAINST.

> Target: Hillary Clinton
> Tweet: *Hillary Clinton has some strengths and some weaknesses. I could vote either way come election day.*

Here, the author has a neutral stance towards the target. The classifier should classify this tweet as NONE.

1.1 Academic Code of Conduct

You are welcome to discuss the project with other students. However, any sharing of code and/or text is not permitted, and each person must submit their own project. Plagiarism tools will be used in your submissions. **If you have questions regarding the project, post them on the discussion forum.**

1.2 Preliminaries

You may use any programming language, any libraries such as NumPy or Apache IO and external programs, etc., to solve this project. Make sure that we can run your code with these dependencies when you submit.

2 Project Task

Given a set of tweets that are annotated with stance information towards various targets, you will have to train a classifier that automatically predicts stance in tweets. You have to assess the performance of the classifier on a test set and carry out error analysis on incorrectly classified tweets.

2.1 Data

The data is part of the SemEval-2016 Shared Task on Stance Detection in Tweets (Mohammad et al. 2016) and you can find more information on the creation of the dataset at http://alt.qcri.org/semeval2016/task6/, as well as an interactive visualization of the data at http://www.saifmohammad.com/WebPages/StanceDataset.htm.

The data files contain one tweet and corresponding annotations per line. The file is tab separated and each line has the tweet ID, the tweet text, the target and the stance label (either AGAINST, FAVOR or NONE). Note that the data contains stance annotations for five different targets ('Atheism', 'Climate Change is a Real Concern', 'Feminist Movement', 'Hillary Clinton', 'Legalization of Abortion'). For the following experiments, choose one of the targets and only consider tweets that are annotated for that specific target.

3 Project Tasks

Step 1: Feature Extraction Compute a feature representation for each tweet in your train and test set. For this project, we use only bigram counts as features. In order to compute the features, first tokenize the tweets, i.e. split the tweet text into separate tokens. Then, compute a vocabulary containing all the unique bigrams (two subsequent tokens) that occur in any of the tweets. Then, for each tweet, compute how often any bigram in the vocabulary occurs in the tweet text.

By the end of this step, each tweet should be represented by a feature vector that has n dimensions, where n is the size of the bigram vocabulary. If you want to include any additional features into your feature representations, you are encouraged to do so.

Step 2: Classification and Evaluation of Classifier Performance Train a statistical classifier of your choice on the feature representations of tweets in the training set. You could for example use any of the classifiers in the python sklearn library.[1] Make predictions for the feature representations of tweets in the test set. Evaluate the performance of the classifier on the test set by computing the F1-score for each of the three categories.

Step 3: Error Analysis Pick at least five tweets that are incorrectly classified as FAVOR and five tweets that are incorrectly classified as AGAINST by the classifier. Inspect these misclassified tweets and for each of them, report if you can reconstruct why the classifier made a mistake for this tweet.

Design Decisions When designing your prediction system you need to consider any preprocessing steps you may wish to perform, in text analysis preprocessing of the data can affect the performance of your models. Such things as stopword removal, normalisation of user names, and normalisation of URLs can impact results. Take these into consideration when designing the project. Whatever design decisions

[1] http://scikit-learn.org/stable/documentation.html

you take with regards to preprocessing, cross validation, or statistical model choice justify them in the report with reasons that supports your design.

4 Submitting your Project

4.1 What to hand in

You have to submit **a single tar.gz file** that contains:

1. Your report in pdf. You must also include the latex sources or the original Word Document also. Please name the file with your KU ID in the following manner **XXXXXX_p2_report.pdf**

2. The source code that you developed including any appendices describing how to run your code

Do not include the datasets or any subset of this. Any shortcomings in your report such as results reported using an arbitrary subset of the data, or choices that are not justified will affect your grade.

4.2 How to Submit

The tar.gz or zip file must be uploaded to Absalon before the deadline. Please upload a copy of your pdf directly to Absalon alongside the tar or zip file. **The name of your file must be your KU username and the project identifier P2.** For example, your submission might look like abc123-P1.zip or pkn877- P2.tar.gz. No late submissions are accepted and will count as a used attempt at completing the exam. If Absalon is down, send your submission to your TA.

References

Mohammad, Saif M. et al. (2016). “Semeval-2016 Task 6: Detecting Stance in Tweets”. In: *Proceedings of the International Workshop on Semantic Evaluation.* SemEval ’16. San Diego, California.