Natural Language Processing
Emoticon Prediction

By: Gornishka, Loor, Qodari, Xumara

November 22, 2014

Contents

Introduction

This report contains the implementation of an emoticon predictor, based on text. This is a project within Natural Language Processing (NLP), meaning many resources within the NLP community were consulted. This report discusses two different ways of prediction emotions. In the first case, emoticons are predicted based on a short message. This lead to the use of an average multiclass perceptron. In the second case, chat data is used to predict emoticons, based on the message as well as the previous post. This lead to the use of the Viterbi algorithm.

Assignment

As stated in the introduction, this report discusses emoticon prediction based on a short message and based on the previous message. In order to find small messages, Twitter was used. For the second part of the assignment, Ubuntu chat data was used.

Twitter emoticon prediction

In this part of the project Twitter data was used in order to train a classifier for predicting emoticons. Three basic classes were used in order to classify the data:

- **positive** - data containing 'positive' emoticons (expressing positive emotions) such as ':)', ':]', '=)', or ':D'
- **neutral** - data containing no emoticons or neutral emoticons like ':|' or 'o_o'
- **negative** - data containing 'negative' emoticons (expressing negative emotions) such as ':(', ':[', '=(' or ';('

wijzers

Data

In order to collect data from Twitter, a crawler was used. The Twitter crawler searched for training and test data, based on emoticons. These messages ('tweets') contain much information - the name of the author, text, hyperlinks, hashtags (sequences of the form '#word_sequence'). Since there are no official restrictions, messages often contain plenty of intentional or - more often - unintentional spelling and grammar errors. Messages also sometimes contain words from other languages, which makes preprocessing and analyzing the data even harder.

Preprocessing

Before any of the data can be used, the messages need to be preprocessed. This process includes the following steps:

- **Splitting the message** into different components - text, hashtags and emoticons.
- **Spelling check and correction**
- **Grammar check and correction**

Model

A multi-class perceptron was chosen as most appropriate for the purpose of this project.

Features

The following features were extracted from the data and used for classification:

- **words** - total number of words in the text
- **positive words** - total number of positive words in the text
- **negative words** - total number of negative words in the text
- **positive words hashtags** - total number of positive words in the hashtags
- **negative words hashtags** - total number of negative words in the hashtags
- **uppercase words** - number of words containing only uppercase letters
- **special punctuation** - number of special punctuation marks such as '!' and '?'
- **adjectives** - number of adjectives in the text

Extracting features such as **positive words** and **negative words** requires the usage of a predefined lists of 'positive' and 'negative' words. Both these lists contain .....

Results

Ubuntu emoticon prediction

Data

Preprocessing

Before any of the data can be used, the messages need to be preprocessed.

Model

Features

Results

Conclusion

Files attached

- Code

Sources

- Some paper