Sentiment Analysis for Social Media comments

Christoph Emunds, Benedikt Heinrichs, Dominik Nerger, Richard Polzin

June 26, 2017

Abstract

Analyzing customer experience offers valuable data for Business Intelligence. Especially in e-commerce, where users write reviews about products and services, the analysis of the customers' sentiment towards these entities yields significant insights into potential strengths and weaknesses of the product or service.

Our work focuses on evaluating customer experience through the application of aspect-based sentiment analysis on social media posts. The data set contains a year's worth of Facebook comments on the pages of two supermarket chains (Tesco and Sainsbury). The goal is to extract as many triplets (e, a, s) as possible, where e is an entity (product or service), a is an aspect of this entity (performance, battery, politeness, etc.), and s is the sentiment polarity label (negative, neutral, positive).

To accomplish this task, many different subtasks need to be solved. After a rudimentary preprocessing routine, the posts need to be POS tagged, Named Entity Recognition (NER) must be applied and sentiment words need to be detected and evaluated. During these tasks we face many challenges, as a significant number of social media texts do not follow the grammatical rules or contain a lot of misspellings. With the completed analysis we identify potential products and services, their corresponding aspects and sentiment words, which are scored and aggregated into an opinion.

The results obtained from this analysis are visualized to get a better understanding of the customers' feelings towards these products, services, and their aspects. The visualization supports businesses in a wide range of decisions, such as product positioning and pricing. This can lead to better customer satisfaction, faster reactions to trends and overall revenue growth.

Contents

1 Introduction

In today's world, Social Media is a big part of the social life. On social networks like Facebook, besides personal accounts there are corporate accounts as well, with the possibility of customers/people interacting with companies, celebrities, sports teams etc. People can review companies or comment on their news feed, allowing interaction between the two entities. These comments or reviews are often filled with positive or negative sentiments because they mostly occur after a really positive or negative experience with the company. Through the comments, it should therefore be possible to extract sentiments about products or aspects of the company. The project presented in this document focuses on finding and analyzing customers' opinions on products and their aspects from social media posts of a supermarket chain. This report describes the required background knowledge and the related work that we build upon. We present how we used the available components to try and fulfill our task. For this task we review the previously defined research questions and evaluate their success or failure.

2 Related work

2.1 Word embeddings

Word embeddings are numeric representations of words where similar words have similar vectors. Today, word embeddings are used as input to a lot of natural language processing (NLP) tasks, which is the reason why the process of training these vectors is sometimes called pre-training.

Word embeddings are generally more concise and have a lower dimensionality than the usual *bag of words* model with one-hot encoded vectors. Depending on the amount of words in the vocabulary, these one-hot vectors have a huge dimensionality, while being sparse (i.e. only one component of each vector is set to 1 and the others to 0) and wasting a lot of space. Such simple models also fail to incorporate meaning of and similarities between words. For example in the sentences

> the cat got squashed in the garden on friday
>
> the dog got flattened in the yard on monday

simple language models do not capture the similarities between the word pairs *cat/dog*, *squashed/ flattened*, *garden/yard* and *friday/monday*.

A famous example for the expressive power of word embeddings is the following:

$$king - man + woman \approx queen$$

This equation shows that if one were to subtract the vector representation for the word *man* from the vector for *king* and add the vector for *woman*, this would result in a point near to the vector for the word *queen*.

The high dimensional vector representations of the words can be processed with the *t-SNE* dimensionality reduction algorithm to be able to plot them in a two dimensional vector space. Figure 1 shows a section of the vector space.

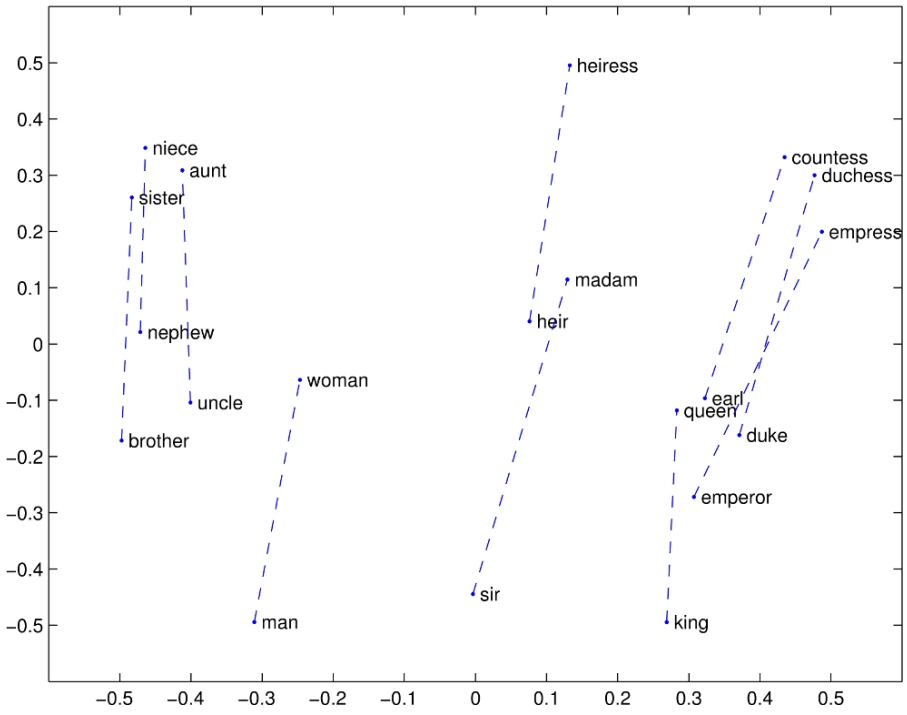

Figure 1: Representation of words relating to the differentiation between man and woman

2.2 POS-Tagging

POS-Tagging (part-of-speech tagging), also called grammatical tagging or word-category disambiguation, refers to the process of identifying particular parts of speech like nouns or verbs.

Identifying the role of a certain word within a sentence is important for the task of sentiment analysis, as identifying entities and their corresponding aspects can be handled much easier relying on certain assumptions about the part of speech of words.

Frequent nouns or noun phrases often describe aspects of products and the vocabulary that is used to describe those usually converges. With such an approach many important aspects can easily be found, but on a more general scheme grammatical based relationships can be used to extract important aspects.

For example words that express opinion on something, like 'great' or 'bad' can be incorporated in rules to extract aspects. To sum it up POS-Tagging is an important part of sentiment analysis.

While being important POS-Tagging is also a complex problem. Word-forms in natural language are often ambiguous. For example the word 'dogs' is usually thought of as a plural noun, but can be used as a verb as well:

> The sailor dogs the hatch.

Due to the complexity, machine learning techniques are often applied in POS-Taggers. Popular approaches such as the Viterbi algorithm, the Brill tagger or the Baum-Welch algorithm work with techniques such as dynamic programming, supervised learning or hidden Markov models.

2.3 NER-Tagging

Named-Entity recognition (NER) is a task that seeks to locate and classify specific information in text. This information is called a named entity and can refer to categories such as the names of

persons, locations, times or many others.

An annotated sentence could look like this :

> [Tim Cook]$_{[Person]}$ has a Net worth of [785 million USD]$_{[MonetaryValue]}$ as of [March 16. 2017]$_{[Time]}$.

Often the task of NER-Tagging is separated into two. In the first part names are detected and in the second part these names are classified.

While state-of-the-art NER-Taggers perform very well and produce near-human performance they are also brittle and do not perform well in domains they were not designed for.

2.4 Dependency parsing

Dependency parsing is a method of building up context in a sentence. It looks on the relations each word has with the other words and tries to create a meaning out of a sentence by that.

For example the sentence *I saw a girl with a telescope* has two times the combination of an object with the word *a*. The word *girl* is specified by the verb *saw* and the object *telescope* is specified by the word *with* with references to the verb *saw*. The verb *saw* relates then to the noun *I* as it is the subject. A visualization can be seen in the following. This is however just one method of analyzing this sentence. Another way of seeing this sentence would be to relate the word *with* with the object *girl* which is possible in the language and gives the whole sentence a different meaning.

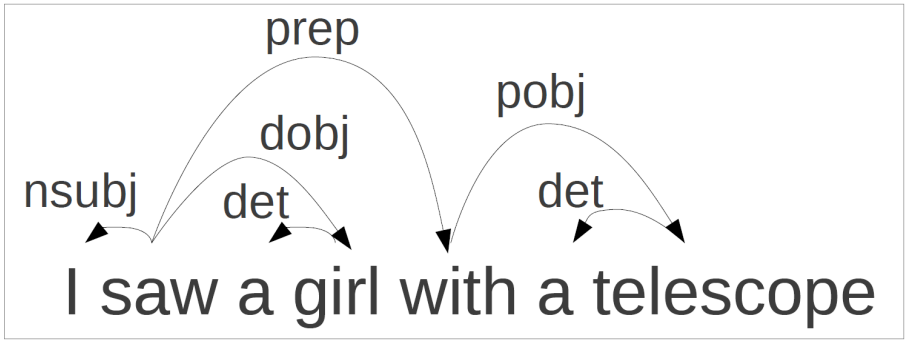

Figure 2: The dependency parsing example

This example has been taken from (?, ?).

3 Preprocessing

We pre-processed the data by extracting the actual text of the posts into separate files, which can be processed by SyntaxNet. We decided to exclude posts that are less than 20 characters long or include images. This is due to the fact that posts which are too short do not yield much information most of the time. Furthermore, posts that include images often refer to objects in the image, which makes it hard to understand the author's sentiment without analyzing the image content.

4 Co-Reference Resolution

5 Extracting entities and aspects

To be able to determine the customers' opinions on certain entities and their aspects, we first need to extract possible entities and aspects from the given dataset. For this task, we will use a method proposed by (?, ?), which consists of the following two steps:

First, we identify frequent nouns and noun phrases with a POS-Tagger. Since people commenting on aspects of a product usually use a limited vocabulary, this process should yield a first set of possible entities and aspects. The frequency threshold needs to be chosen experimentally.

However, since the first step can miss quite a lot of important aspects, the next step tries to find some of them by exploiting the relationships between aspects and opinion words.

6 Determining sentiment polarities

For determining the sentiment polarities, we use a lexicon based approach that is executed with dictionaries to rate the sentiments that are expressed.

6.1 Lexicon-based approach

An approach to determine sentiment polarities is the lexicon-based sentiment analysis.

We applied a method by (?, ?), which can be broken down to the following four steps:

In the first step, all sentiment words and phrases in the text will be marked with the help of a sentiment lexicon. This is done by assigning positive or negative values to those sentiments, e.g. [+1] for a positive or [-1] for a negative sentiment.

In the second step, sentiment shifters will be applied. A sentiment shifter can be described as a word that negates the value that has been applied in the first step, changing its value from positive to negative or the other way around. Typically, these words are negation words like *not*.

In the third step, but-clauses and further contrary words are handled. With these contrary words, sentences can be split up into two parts. Depending on the value of the sentiments mentioned in the first part, the sentiments in the second part receive the opposite value.

In the fourth and last step, the opinions need to be aggregated according to the following function:

$$score(a_i, s) = \sum_{ow_j \in s} \frac{sw_j.so}{dist(sw_j, a_i)}$$

where sw_j is a sentiment word in s, $dist(sw_j, a_i)$ is the distance between aspect a_i and sentiment word sw_j in s. $sw_j.so$ is the sentiment score of sw_i. Depending on the score, it can be determined whether the opinion on the aspect a_i in s is positive or negative. If the final score neither positive or negative, it is a neutral aspect.

This method can be made more effective by applying dependency parsing to the sentences, such that the scope of each individual sentiment word can be determined more accurately. This allows us to discover the sentiment orientation of context dependent words (?, ?). Due to limited time however this approach was not followed.

7 Results

We built and tested the syntax parser *Parsey McParseface* (aka SyntaxNet) by Google, which we used as a POS-Tagger for extracting entities and aspects and as dependency parser for several other tasks. SyntaxNet is easy to use and yields state-of-the art results. Moreover, it is based on Google's Deep Learning Framework TensorFlow and therefore integrates nicely into the rest of our implementation. The implemented models are explained in detail in (?, ?).

We will use the sentiment lexicon by (?, ?), which includes mis-spellings, morphological variants, slang and social-media mark-up of 2006 positive and 4783 negative words. Moreover, we will use pre-trained word embeddings from the GloVe project (?, ?).

7.1 Visualization

Because it is time consuming to take the results and derive a meaning from them by hand, we decided to visualize the results in a web application. On the server side, we use Python with ... . On the client side, we use the JavaScript libraries d3.js, vue.js and jQuery.

For the visualization, we use the *JSON* file containing the aggregated opinions that are the result of executing the pipeline as well as a *JSON* containing the positive and negative sentiment words that are provided by the lexicon-based sentiment analysis.

The data is visualized with each product having its own page that is loaded from a dropdown menu and each aspect of the product being visualized with a bar chart as well as the specific posts being shown below the bar chart. The bar chart can be seen in Figure 3. It shows the amount of negative, neutral and positive sentiments regarding the combination of product and aspect.

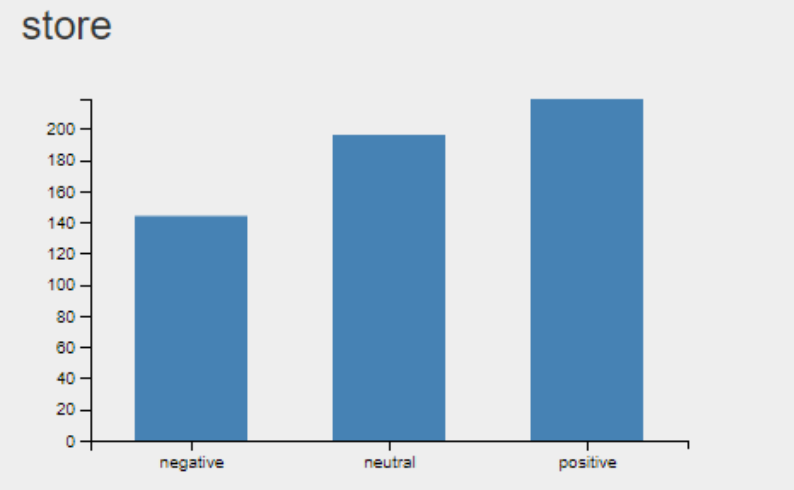

Figure 3: Bar chart displaying the sentiments regarding the customer service with aspect 'store'

An excerpt of posts regarding the combination of product and aspect is shown below each bar chart, with a maximum of 10 posts per aspect. In Figure 4, the highlighted posts can be seen. Positive and negative sentiments are highlighted in colors green and red, respectively. Mentions of the product are highlighted in orange and the aspect is shown with a cyan highlighting.

Figure 4: Highlighted posts regarding the customer service with aspect 'store'

8 Previously defined goals and their fulfillment

The overall goal is to extract as many quintuples $(e_i, a_{ij}, s_{ijkl}, h_k, t_l)$ as possible, where e_i is the i'th entity and a_{ij} is the j'th aspect of entity i the opinion is expressed on. s_{ijkl} is the sentiment polarity, which can take on the values *positive* or *negative*. h_k describes the opinion holder and t_l the time at which the opinion was expressed.

We build a database with opinions and statistics on several products and their aspects. For a visualization of the results we build a simple dashboard with state-of-the-art web technologies.

Based on these goals the following research questions were defined:

- Do state of the art techniques in sentiment analysis provide valuable results when applied to Facebook posts?
- How do supervised (LSTM-Network) and unsupervised (Lexicon-based) techniques for sentiment analysis compare?
  - Is it possible to apply the LSTM-Network approach proposed by (?, ?) on an aspect level, rather than document level?
  - Which technique offers better results, how does it compare to the effort required to use that technique?
  - How does a combination of the techniques affect performance?

9 The code pipeline

To reproduce our work, the provided code has to be executed in a certain sequence. The following will talk about the needed steps.

10 Conclusion and Future Work