DALARNA UNIVERSITY

Business Intelligence - DT3018

Final Project Report – 22 May 2017

Natural Language Processing
Definition and Application for Analyzing YouTube Comments

Author: Marina Ferreira Uchoa

h16mferr@du.se

1 Introduction

In this report I will discuss what is Natural Language Processing (NLP), as well as its objectives, levels of analysis and approaches. NLP is a key aspect of linguistic analysis and text mining, allowing for computerized knowledge extraction from text and speech [1]. Automatically processing text documents is a critical necessity of contemporary society. There exists several underused textual databases from which valuable information can be obtained to promote advances in both public and private companies [2].

An area in which NLP has been widely used for is customer relationship management, more specifically for analyzing the customer perception over a specific product, service or company/brand. In this field, Sentiment Analysis, also called Opinion Mining, is a technique for assessing positive and negative opinions to better understand and address the customer's needs.

Even though sentiment analysis is not the mean issue undertaken in this project, it was used for demonstration of an actual and useful application of NLP. The idea was to process comments from an arbitrarily selected video from BuzzFeed, an American digital media company focused on producing viral content. They are responsible for video series such as Tasty and Ladylike and have 12.458 million subscribers on their main channel and of 21.468 million on their secondary channels on YouTube, summing up to a total of almost 34 million followers.

The video selected is from a series named Worth It. It is a somewhat new BuzzFeed programme, with the first video having been released on September 2016 and with a total of 24 videos available on YouTube. The Worth It videos compare the same foods, services or experiences at three different price-points, which they name "$", "$$" and "$$$", to select the one that is worth the most.

In the next section, I will provide a brief discussion over NLP. Then in the two following parts, the methodology and data collection processes will be addressed. The fifth section will cover data analysis and visualization. Finally, the last component of this report is a discussion over the project work carried out.

2 Natural Language Processing

NLP can be defined as a set of mathematical and computational techniques used for analyzing, representing and/or producing written or oral textual content in one or more human languages [1, 3, 4]. Early NLP systems were developed to be used in systems that perform machine translation. However, there are several other tasks it may be useful for, such as information retrieval, information extraction, question-answering, summarization and dialogue [5, 3].

This approach attempts to mimic human language processing and is usually considered a means for attaining a desired output, instead of an end in itself. It performs analysis in one or more of these seven levels: phonetic, morphological, lexical, syntactic, semantic, discourse and pragmatic [3]. Each of these contribute with the understanding of the meaning transmitted through text/speech. The author advocates that a "full" NLP system should take all levels into consideration.

The phonology level deals with pronunciation and interpretation of oral communication. In this level, individual word and word sequences pronunciations are analyzed by employing phonetic and phonemic rules, while stress and intonation, by prosodic rules. The morphological level analyzes the constituents of each word, e.g. roots, preffixes and suffixes. The lexical level covers the functions of individual words in a sentence. Where more than one function is possible, a statistical analysis is performed and the most likely part-of-speech is assigned. This contributes to understanding the meaning of the sentence by elucidating order and dependency - which word is subject, object, and so on.

The syntactic level deals with the grammatical structure of the sentences and evince the dependency between words. The semantic level covers the analysis of word interactions to come up with the possible meanings of a sentence. In this level, semantic disambiguation is performed by assigning the most likely meaning to a polysemous word given the context.

Discourse analysis complies analyzing the text as a whole, with each sentence contributing to part of it's overall meaning. In this level, pronouns can be replaced by the nouns they replace in order to evince its meaning in the context. The last level (pragmatic) analyzes meaning that might be embedded in the text by the real world context without being encoded into words.

Performing all of these levels of analysis enables the NLP to mimic human language processing and, therefore, to manipulate natural language in order to perform the tasks needed by the user [3, 5]. It is important to notice, however, that NLP is not the only, nor the first approach to attempt to extract valuable information from textual databases[2].

A simpler way of processing textual input is the bag-of-words approach. Is is distinct from NLP because it performs the analysis only at the word level. It disregards word order and grammatical relations between the elements in sentences, paragraphs and entire texts [2]. This approach compares the individual contents of the text with a predefined collection of words assumed to be related to an issue of interest, e.g. being spam or not, being positive or negative. Grammar and word order are not considered in this approach.

NLP on the other hand, attempts to take the complex relations between textual elements into account to better approximate the meaning embodied in the text. It has a better outcome than the bag-of-words method, however at a higher cost [2]. The simpler the model, the easier it is to implement and maintain. Thus the broader the analysis, most likely the more expensive it will become.

For the purpose of developing, testing and validating systems that address one or more parts of NLP, large-scale grammars and lexicons are needed [1]. In this context, WordNet is an online lexical database of English words developed by Stanford University's NLP lab [2, 5]. In this large lexicon that contains words gathered into synonym groups as well as semantic relations between these sets.

A lexicon is similar to a dictionary of words, their synonyms and meaning. As Sharda, Delen and Turban point out, WordNet is a very commonly used general-purpose lexicon [2]. Several extensions to WordNet have been developed to accomplish specific tasks.

NLP faces several challenges, some of which are performing accurate part-of-speech tagging, text segmentation, word sense and syntactic disambiguation, imperfect/irregular input and the analysis of speech acts. Some of these issues are common to all text analysis, such as part-of-speech tagging and disambiguation. However, some are more characteristic to the informal writing, namely segmentation and imperfect input.

Due to the nature of online opinion sharing, problems atypical of the language may arise. One example is that several internet posts contain typographic errors that not only hinder machine applications, but may also impose the need of segmenting words. The same problem derives from the advent of hashtags, which made key aspects of online communication be contained in space-less strings beginning with a #.

Segmentation is a very important aspect of processing languages such as Korean and Japanese, however, it is not usually a main problem in English - words are separated from each other with a space. For this reason, not many applications have been developed to deal with such issues.

2.1 Sentiment Analysis

Sentiment analysis is the branch of computational linguistics dedicated to assessing the opinion of people with regard to a specific topic. It may be used to check for consumer satisfaction with a product or service, to identify need for improvement in certain areas of customer service or how citizens regard a specific politician, for example. Having a large lexicon database is useful for the purpose of sentiment analysis as a source of knowledge about emotions transmitted by the words in the text.

There are two main approaches to sentiment analysis. The first is sometimes called polarity identification and intends to classify textual and speech content into positive, negative and neutral classes. Several distinct emotions may conform what is within these classes, however, the grouping of texts is based only on their overall positiveness or not. The second approach attempts to identify predominant emotions within textual exerts. These may be anger, joy or fear for example.

In this project I do not advocate for one approach or the other and will use both polarity and sentiment classifications for the analysis of YouTube comments. For this purpose, the WordNet-Affect by Strapparava and Valitutti [6], which is a WordNet extension, will be used to classify the comments into sentiments of anger, disgust, fear, joy, sadness and surprise. To complement this analysis, Wilson, Wiebe and Hoffmann's subjectivity lexicon will be used for assessing polarity by classifying the comments into positive, negative or neutral [7].

3 Methodology

As highlighted by Sharda, Delen and Turban [2], text mining is performed by following three main steps. The first is to gather data and build a corpus. The second is to create the term-document matrix (TDM) with word frequencies across documents. The third step is to actually extract knowledge from the data. NLP is a primary component of the first and second steps, providing information for the knowledge extraction.

Establishing the corpus is dependent on the scope of the project and the area of interest. In this present study, the textual data is retrieved from a single YouTube video. The channel and specific video selection was arbitrary, I chose a video I found interesting from the Worth It series: $8 Salmon Vs. $56 Salmon.

The extraction of comments from the YouTube site was performed with a web scraper specifically designed for this task. The characteristics of the scraper and a discussion on how to use it will be presented in the next section. For this part, it is sufficient to know that raw comments were retrieved using R [8] and the YouTube API (Application Programming Interface).

The retrieved information included the actual comments, the user who posted the comment, the numbers of likes and replies the comment had, the date and time when the comment was published and the comment identification number. It was stored in a data frame named `comments`.

For the analysis undertaken in this work, only the textual comments were analyzed. A suggestion for future studies would be to use like and reply counts as weights for the relevance of a comment over the others.

The next task for building the corpus was to clean the comments to eliminate the characters which were not letters, spaces or dollar signs. I opted to keep dollar signs because of the name of the series and of how they opt to name their videos ($ Vs. $). Apart from that, all emoticons and other non-alphabetical characters were removed.

As a result of the cleansing, 18 comments became empty and were also removed from the data set. Then, all comments were transformed into lowercase letters only.

The final step for the comment's global analysis was to remove the English stopwords from it. This process eliminates words that do not contribute with meaning and whose presence may distract the viewer from actually meaningful words. The output was then used to assess word frequencies across documents. Package `tm` performs this without the need to convert the data vector into a `TextDocument`, which would be the corpus [9].

Complementary to the overview of all the comments, sentiment and polarity were assessed by using the `sentiment` package [10]. This package employs a naive Bayes classifier trained on the WordNet-Affect and subjectivity lexicons to assign the most likely sentiment and polarity to each comment. For this purpose, two main corpora were created, one for sentiments and the other for polarity by constructing two distinct TDM.

Each corpus was constructed as to hold aggregated data related to each sentiment/polarity. This allowed for visualization of word frequencies in terms of how frequent they were for each emotion, as well as for each polarity category.

The function for sentiment analysis evaluates six different emotions, namely anger, disgust, fear, joy, sadness and surprise and returns the most likely sentiment or "NA" if no specific sentiment was found to be predominant. The function for polarity, on the other hand, assigns either "positive", "negative" or "neutral" to all comments that are passed to it. Based on the outcomes of these functions, a sentiment database was constructed by combining comments, sentiment and polarity in the same data frame.

The third step for text mining is knowledge extraction. For the purpose of this work a basic wordcloud approach will be adopted. It serves the purpose of demonstrating the capabilities and challenges for text mining. A wordcloud visually displays the words in a document or set of documents with the word sizes varying according to their frequency.

4 Data Collection

Data collection was performed by creating a new class of objects named `yt_scraper`. The main code for this class was posted by user *nrussel* on StackOverflow and then adapted to fulfill my own necessities.

This class has fields for: the base API URL; the code for the next page to be scraped;

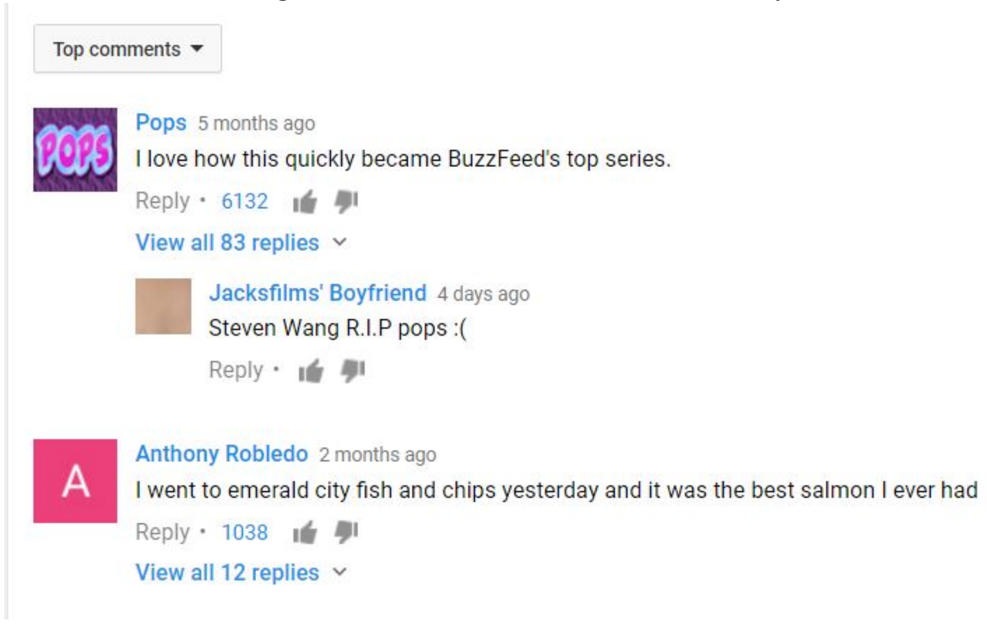
the data retrieved as a list; the counting of how many entries were retrieved; a logical for checking whether the work is over or not; and a data frame for saving only the features of interest, namely the comments, the user, the numbers of likes and replies the comment had, the publication date and time and the comment identification number.

The way the class was coded, its constructor does not take any argument. Thus, the identification number of the video to collect data from, as well as the Google developer key and type of data to retrieve are all assigned inside the class code as default values. No method was implemented to alter these default values. Therefore, whenever a new video is chosen for analysis, the user must manually alter the class constructor.

I find this rather annoying, but it is not a problem for the purpose of this work. Since there were time constraints and I managed to get the YouTube content I needed, I opted to leave the class as it was. However, I acknowledge that this is detrimental to further usage of the class and plan to make the needed changes.

With regard to the actual data collected, YouTube differentiates main comment threads from its replies, providing specific API request formats for each kind of collection. For the purpose of this work, only main comments were analyzed - those that are posted directly as a comment of a video, instead of a reply to a prior comment. To allow a better visualization of the hierarchical display of YouTube comments, I snipped the two top comments of the video to be analyzed (Figure 1).

Figure 1: YouTube Comment Hierarchy



Pops 5 months ago
I love how this quickly became BuzzFeed's top series.

Jacksfilms' Boyfriend 4 days ago
Steven Wang R.I.P pops :(

Anthony Robledo 2 months ago
I went to emerald city fish and chips yesterday and it was the best salmon I ever had

Snipped from YouTube on 2017/05/22 at 1:49 p.m.

As of may 2017, YouTube does not allow for replies of replies, even though they express the possibility of a change in the near future in their API documentation. Current data contain only two hierarchy levels. The first are the comments posted to a video, while the second are comments posted as a reply to a comment posted to a video.

For also retrieving replies, the main change needed in the `yt_scraper` class is to specify `part = "snippet,replies"` instead of only `"snippet"` in the constructor. However, the replies will not be displayed on the data frame with the results unless an additional change is made. The way the data set will be structured to represent the dependencies depends on the needs of the researcher and should be considered prior to making changes to the class.

To access data from any Google API, two authorization formats may be available: OAuth and developer key. Both function as identifiers of the person requesting data from the API and allow Google to limit the number of requests a user can make each day, as well as monitor who is requesting what.

I preferred to work with a developer key, since it seemed like a simpler approach. To set a key and allow API requests, one must sign in with a Google account to GoogleAPIs, create or select an existing project, go to the API Manager on the left side menu and select `Credentials >> Create credentials >> API key`. To authorize the YouTube API, then from the API Manager, select `Library >> YouTube Data API >> ENABLE`. Once this whole process is complete, then all that is needed is to input the developer key into the constructor of `yt_scraper`.

Once the developer key is set and put into the constructor, all that is needed to run the scraper is: (1) run the code creating the class; (2) call the `httr` package [11]; (3) create an instance of the class; (4) call the `scrape_all` method; and (5) save or view the data frame containing comments and the information related to them - number of likes and replies, identification and author.

Assuming the key is set and the code creating the class has already been run, all that is needed is:

```
library(httr) # step 2
scraper_object <- yt_scraper() # step 3
scraper_object$scrape_all() # step 4
comment_data_frame <- scraper_object$core_df # step 5
```

5 Data Analysis

After running the `yt_scraper`'s `scrape_all` method, I had obtained 11155 comments. As discussed in the previous section, a few of those ended up empty after cleaning. These were comments that contained only emoticons, numbers or other non-alphabetical and non-dollar-sign symbols. After this process, there were still 11137 comments.

To allow the analysis of such large corpus, I have created wordclouds for all comments at once, comments split into polarity classification and comments split into emotion classification (Figures 2, 3 and 4). Additionally, to better address the predominance or not of one or more emotions or of positive, negative or neutral comments, Table 1 displays how many comments have been classified within each category. It is worth noting that, for emotion classification, approximately 70% of the comments were not assigned to any of the classes. This may suggest that additional levels of analysis or even a better classifier implementation could improve the results obtained.

From the comments that were tagged to an emotion, the majority (76%) was identified as joyous comments. With regard to polarity, 68% of the comments were identified as

Table 1: Summary of Sentiment and Polarity Classifications

Emotion	# of Comments	Polarity	# of Comments
Anger	220	Negative	2438
Disgust	34	Neutral	1157
Fear	71	Positive	7542
Joy	2690		
Sadness	376		
Surprise	131		
NA	7615		
TOTAL	11137	TOTAL	11137

positive. Both these facts may indicate that BuzzFeed is indeed doing a good job with relation to consumer satisfaction.

An initial overview of the comments' contents can be visualized in the wordcloud depicting the most frequent words across all comments: Figure 2. As can be seen in the Figure, the most common word was "salmon". It makes sense, since the episode was about eating salmon prepared at different restaurants. Nonetheless, an analysis of the emotion associated with that word can help in extracting more information.

From the polarity wordcloud (Figure 3) we can see that salmon appears in the pool of positive comments. Checking the data set for positive comments, salmon appears in several comments regarding how people love/like salmon. However, by taking a look at the emotion wordcloud (Figure 4), we can verify that salmon has been identified as a predominant word within comments expressing disgust.

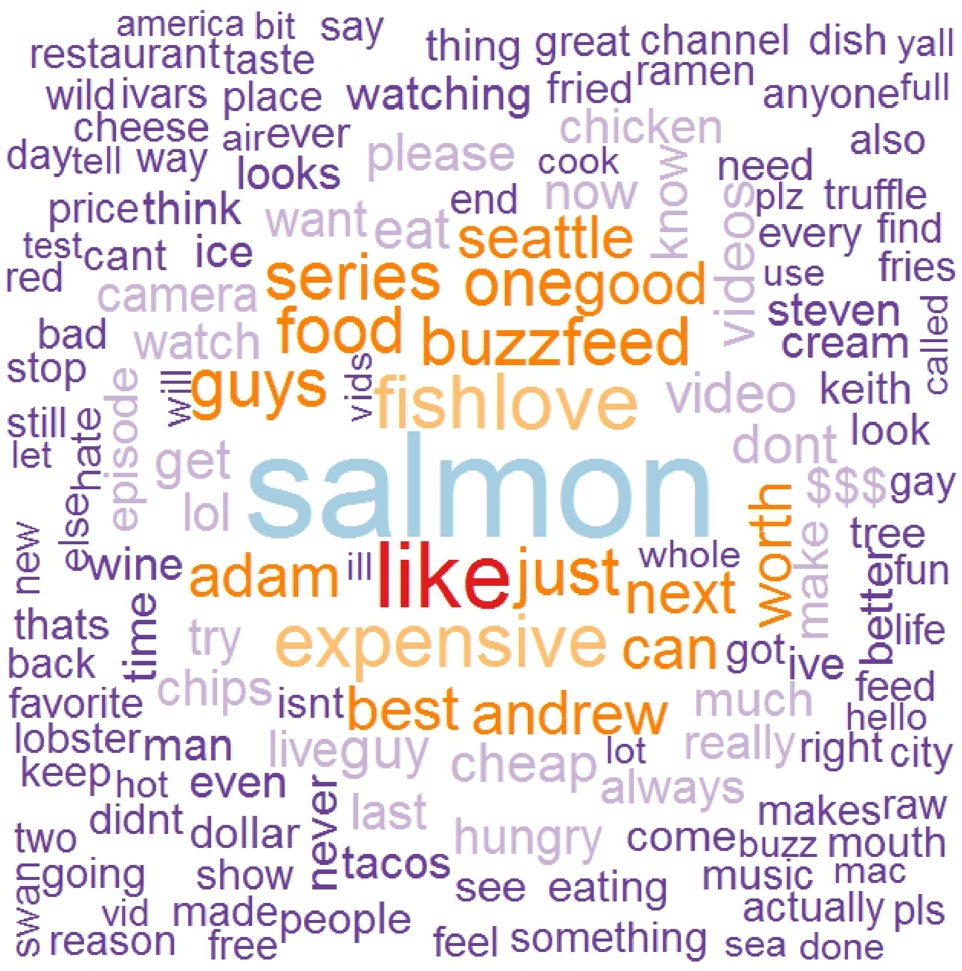
The latter fact may seem intriguing, since one would expect salmon to be associated with joyous feeling and not with disgust. However, by searching the comment data base, we can verify that 12 out of the total 34 comments expressing disgust actually comment on how they find salmon disgusting. There are two main reasons, the first being that they dislike it. The second is due to the coloring fed to salmon so it becomes pink. It is important to notice that, even though salmon is very frequent within the disgust class, it only has 34 comments, while the positive polarity has 7542 (see Table 1). Therefore, we can dismiss salmon as a major problem with regard to the YouTube comments analyzed.

The other words I would like to comment on are "camera" and the sets of words "ice"-"cream" and "please"-"plz"-"pls". Starting with camera, it is a relatively common word in general and it also appears in the positive words cluster. However, what could be the meaning behind this word. What does camera mean in the context of this video? This answer is not readily given by just looking at the figures. Taking a look at the original comments is the key to address this issue.

From the data base, one can verify that camera actually refers to the cameraman, Adam. By taking this into consideration, we can see that both camera and Adam refer to the same individual and should have been grouped as synonyms. If we take into account the number of times both words appear in a comment, we would have Adam appearing 437 times and camera, 258. Together, they would sum up to 695. This is approximately equivalent to one third of the appearance of the word salmon, which amounted to 1903.

Assessing that Adam and camera refer to the same person would require analysis also in the pragmatic level, which is very costly and complex but would increase the accuracy.

Figure 2: Wordcloud with All Comments



Additionally, an intelligent spelling checker could assess that the bigram “camera”-“man” is actually a typographic error of “cameraman” and substitute those entries with its correct spelling. Hence grouping “cameraman”, “camera” and “man” into a single term.

I tried to find an implementation of a spelling checker and found Peter Norvig’s suggestion, which was implemented for R by Rasmus Bååth. However, by not taking the context into consideration, this implementation ended up not helping with the task I was aiming and I removed it from my code. A more complete application, taking other levels into consideration would be very much appreciated, however, I did not find any good implementation yet.

The next aspect I want to discuss is the fact that the basic implementation carried out here does not account for n-grams. This can be visualized by taking the example of the “ice-cream word set. Even though they are not considered a single concept, by analyzing the data we can verify that all 131 comments containing “ice” refer to ice cream. Tokenizing in n-grams is implemented in R, I did not use it due to time constraints.

The final word set I would like to discuss is please-plz-pls. All three appear in the wordclouds for all words, as well as for positive ones. Even though they do are not similar to the look ar to a machine, in informal texts they are the same word. Other slang abbreviations include “idk” for “I don’t know” and “omg” for “oh my God”. Treating these irregular inputs is another challenge for NLP applications. One way to minimize this issue would be to develop a dictionary of substitutions. However, new abbreviations will emerge and an intelligent self-enhancing system would be a much better approach.

Figure 3: Polarity Wordcloud

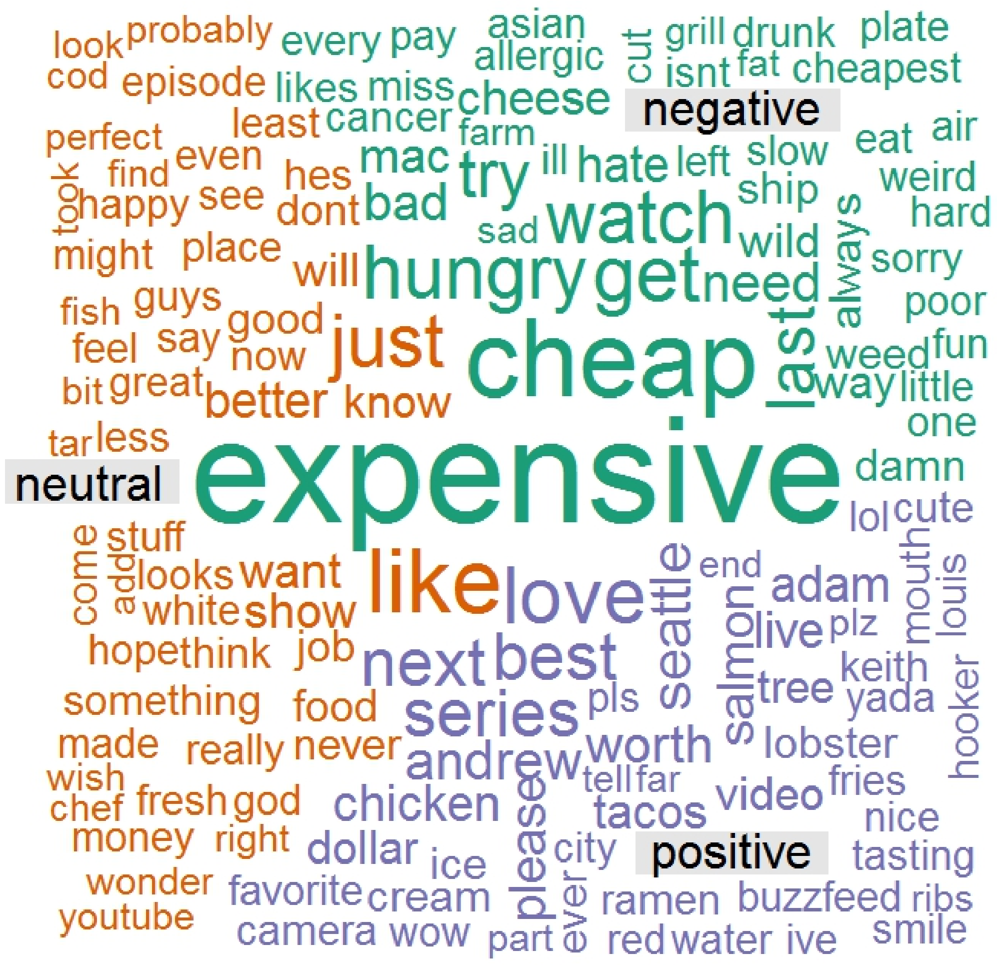
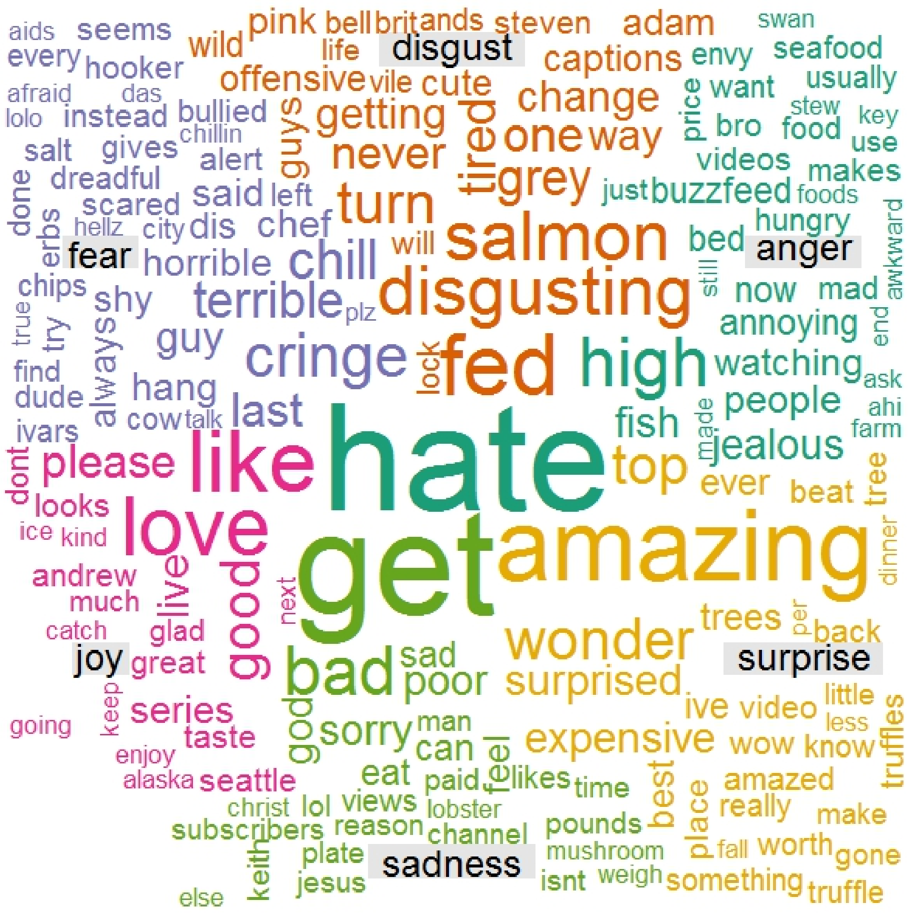


Figure 4: Emotion Wordcloud



6 Final Considerations

The present project work aimed to discuss NLP, its application and challenges with a study of an arbitrarily selected BuzzFeed video. The analysis has shown that even though many theoretical developments have been made, readily available applications are still needed. It has also indicated that NLP's approach of multi-level analysis have a huge potential of improving textual analysis. The work carried out serves as a primary approach to NLP and intends to provide unfamiliar readers with encouragement and basic knowledge to pursue their own research in the field.

References

[1] James F. Allen. Natural language processing. In *Encyclopedia of Computer Science*, pages 1218–1222. John Wiley and Sons Ltd., Chichester, UK, 2003.

[2] Ramesh Sharda, Dursun Delen, and Efraim Turban. *Business Intelligence and Analytics: Systems for Decision Support*. Pearson, 10th edition, 2014.

[3] Elizabeth D. Liddy. Natural language processing. In *Encyclopedia of Library and Information Science*. Marcel Decker Inc, NY, 2nd edition, 2001.

[4] Aravind K. Joshi. Natural language processing. *Science*, 253(5025):1242–1249, 1991.

[5] Gobinda G. Chowdhury. Natural language processing. *Annual Review of Information Science and Technology*, 37(1):51–89, 2003.

[6] Carlo Strapparava and Alessandro Valitutti. Wordnet-affect: an affective extension of wordnet. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, pages 1083–1086, Lisbon, May 2004.

[7] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[8] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016.

[9] Ingo Feinerer and Kurt Hornik. *tm: Text Mining Package*, 2017. R package version 0.7-1.

[10] Timothy P. Jurka. *sentiment: Tools for Sentiment Analysis*, 2012. R package version 0.2.

[11] Hadley Wickham. *httr: Tools for Working with URLs and HTTP*, 2016. R package version 1.2.1.