American University of Armenia

College of Science and Engineering

Capstone Project

Real-time Sentiment Analysis on Social Media Data

Authors:

Nareg Barsoumian

Tigran Sedrakyan

Supervisor: Saro Deravanesian

Spring, 2019

”If it’s going to be a world with no time for sentiment, it’s not a world that I want to live in.”

– Christopher Isherwood

Abstract

Sentiment analysis or opinion mining is the process of extraction and classification of emotion from the text. The purpose of this paper is to research and explore different techniques and types of sentiment analysis, along with a fast, scalable, highly accurate and flexible sentiment analysis library VADER, used primarily for social media sentiment analysis, and describe methods used for further expansion, development and fine-tuning of the library. This paper addresses the problem of assigning sentiment intensity scores to texts from realtime data, as well as labeling them into “Positive”, “Negative”, and “Neutral” using a fast, rule-based sentiment classification method that is built on a highly developed lexicon.

Contents

1 Introduction

1.1 Problem Statement

Sentiment analysis is one of the branches of natural language processing (NLP) that involves processing of the language using computational techniques. It is the process of extraction of subjective information from the text using various methods. As such, it represents a big field and is of major importance for automated text processing. Our target is to use sentiment analysis techniques on data gathered from social media, primarily Twitter, which is a major source of information. For this purpose we aim to build/extend over the existing and openly available code, by tweaking and making it more suitable for the usage in the context of social media analysis.

1.2 Brief History

Even though natural language processing has an expansive history, taking its roots from 1950s and involving fields such as automatic machine translation or speech recognition, sentiment analysis in its modern sense is a relatively young branch, as it only became possible due to the spread of the textual information on the computers, through the invention and development of the Internet technologies, as well as technological advancement, that made possible the usage of machine learning techniques in natural language processing. Therefore, major interest in sentiment analysis became visible after year 2001, when number of paper publications on the topic exceeded 100, making it a quite popular topic.

1.3 Motivation

The motivation behind the work done within the boundaries of this project is the absence or incompleteness of openly available libraries, which would be fast, accurate and scalable enough to be able to perform sentiment analysis on a realtime data. A bit of research revealed that such library indeed exists, and, moreover, it is used for social media analysis. It is called VADER[1] (Valence Aware Dictionary and sEntiment Reasoner) and has a well developed lexicon. However, further research shows that despite a good lexicon, code-wise it is quite incomplete and may fail for several scenarios. Thus, we concentrated on improving, developing and modifying the existing solution in a way that would fit our needs better.

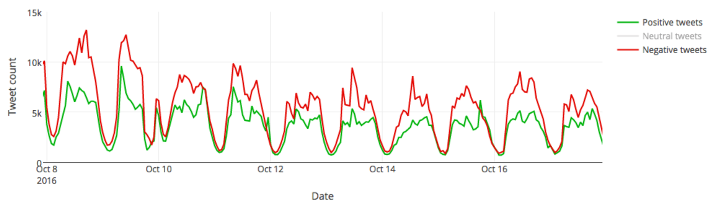
[1]https://github.com/cjhutto/vaderSentiment

1.3.1 Applications

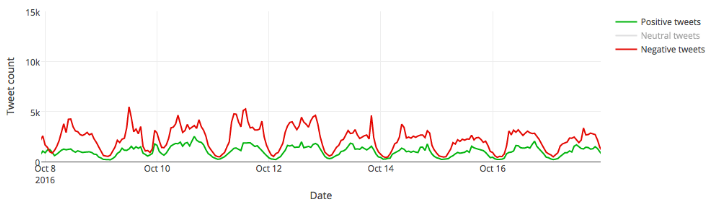
Yet another reason why we decided to concentrate on sentiment analysis is that it has multiple application over the data all across the Web.

Social media became a place where people not only express their ideas and opinions about events happening all around the world, but also use it for scholarly work, and often regard it as a primary source, making it a major supplier of textual information. Thus, being able to perform analysis of the text on the social media is of crucial importance for understanding not only the present, but also being able to evaluate future outcomes.

A simple example of application of sentiment analysis on social media, is its usage with the purpose of analyzing sentiment and opinions about electoral candidates for presidential elections, to be able to compare their probabilities of becoming a president. A similar experiment[2] has already been performed for the United States 2016 presidential elections and results are shown in the Figure 1.



(a) Tweets about Trump



(b) Tweets about Clinton

Figure 1: Positive and negative tweet count for each of the candidates

The two candidates in the final phase of 2016 presidential elections were Donald Trump (Republican Party) and Hillary Clinton (Democratic Party). Figure 1 shows that positive to negative ratio for Trump is higher. Also there are more tweets about Trump. As a result, Trump won the elections in 2016.

Although the primary source of our textual information is social media, VADER is easily extensible to other domains, such as critical reviews on the review aggregators all over the internet. For example, analyzing the text of reviews of movies, in addition to the review ratings can help movie authors get a better understanding of how their work has been received by the audience, and giving them more insight on how

[2]https://monkeylearn.com/blog/trump-vs-hillary-sentiment-analysis-twitter-mentions/

to improve, without necessarily reading all the reviews one by one.

Another example of application of social media sentiment analysis is shown in the Case Study 1 of this paper. It involves analyzing Twitter data to understand happiness level in countries all around the world.

2 Theoretical Background

In this section we will be discussing some of the sentiment analysis types and different approaches and techniques which are widely applied in the field of sentiment analysis.

2.1 Sentiment Analysis Types

2.1.1 By purpose

By the purpose, sentiment analysis can be divided into two categories - polarity classification and subjectivity classification.

Polarity classification primarily aims to classify the sentiment of a text into generally three categories - ”Positive”, ”Negative” and ”Neutral”, or on more advanced level, labeling text by emotions, such as ”happy” or ”angry”. Subjectivity classification is used as a tool for determining objectivity/subjectivity of the text, by classifying it as more objective or more subjective.

2.1.2 By scope

By the scope, sentiment analysis can be divided on two main categories - coarse-grained and fine-grained.

The first one, in turn, is divided into two types - document level and sentence level. These are relatively simple, as they seek to determine the polarity of the whole document or sentence, respectively, by scanning for positive/negative words and using a certain technique or formula.

Fine-grained sentiment analysis, also sometimes called feature/aspect level sentiment analysis[3], is a more complicated approach. It targets to isolate the entity in the text, and find the feature(s) or aspect(s) regarding it. For example, a sentence like ”I love Russia, but the weather here is unbearable”, has a generally positive tone, where the entity is ”Russia” and the aspect is the ”weather”, however it is stated as ”unbearable” bringing in the negative side. The feature/aspect level approach tends to deal with cases like this, making it more sensitive.

The feature/aspect level approach is mainly used for analyzing sentiments of product reviews, movie reviews, etc. The sentence level and document level approaches are more general and can be used in many contexts.

2.2 Sentiment Analysis Approaches

Sentiment analysis is a massive field with many open-ended problems such as comprehending figurative-speech or being able to detect sarcasm in a text. Figurative language as opposed to the literal language, uses

[3]https://pdfs.semanticscholar.org/ee6c/726b55c66d4c222556cfae62a4eb69aa86b7.pdf

words that have unrelated meanings such as ”The best I can say about the course is that it was interesting”. Here ”interesting” does not necessarily imply positive sentiment. Taking into account all the difficulties and problems in sentiment analysis, all the solutions are narrowed down to two main approaches, lexicon(-based) approach, and machine learning (ML) approach.

2.2.1 Lexicon Approach

Lexicon approach aims at building a lexicon by assessing words and mapping them to a dictionary. Lexicon is a dictionary which assigns a fixed polarity score to each of the words. Sentiment of phrases and sentences can be evaluated by using the sentiment polarity of each word in the lexicon. Lexicons can be categorical, categorizing words on a discrete scale from the set of ”Positive”, ”Negative”, ”Neutral” labels or numerical which evaluate words on a continuous scale, usually from -1 to 1, where the most negative polarity corresponds to -1 and increases to neutral near 0 and positive near 1. The numerical lexicon approach is better than categorical because it does not simply categorize the overall sentiment but evaluates it stating how ”Positive” or ”Negative” the text is.

Among the advantages of this approach is that it is very fast compared to the ML-based approach. Also it is flexible, because a lexicon can be easily extended. This is particularly handy when ”translating” lexicon by adding words from other languages, and possibly even from slang and emojis. This, however, comes at a cost, by limiting this approach to the words present in the lexicon only, requiring a manual assignment of each word’s polarity score without access to a specific context, which make it a particularly challenging and time-consuming task.

2.2.2 Machine Learning Approach

Machine learning approach aims at training a model with texts that are already classified and have known sentiments. The idea here is that the trained model will eventually be able to evaluate or categorize unknown texts’ sentiments. The advantage of this approach is that it analyzes whole sentences/documents and gives the trained model an access to the general context, rather than each word separately. This allows to avoid spending time on building and fine-tuning the lexicon, which might yet fail in particular scenarios. However, the machine-learning approach has its drawbacks. It requires a massive amount of diverse textual data such as movie reviews, social media text, product reviews, etc., to be able to work across different domains. It also requires intensive computational power, like CPU and memory, and, therefore, a significant amount of time to train the model, which might get outdated in a short time, requiring re-training and spending additional time. Finally, this approach is difficult to debug, modify, and predict, as techniques used in machine learning usually are not easily comprehended by humans.

3 Features and Improvements on VADER

In this section we will be discussing the choice of the data platform, used for gathering a textual information, and the approach and the type of the sentiment analysis we chose, along with the reasoning behind choosing those. We will discuss why we considered VADER as the perfect choice for our particular scenario, as well as write about our observations on its features, its strengths, and some of the drawbacks that we have found and improved.

3.1 The Data Platform

Twitter is one of the most used social media platforms. It has 321 million monthly active users, 126 million daily active users[4], and 6000 tweets are posted on average every second[5]. What makes Twitter different from other social media platforms is that people mostly use it to share their opinions and ideas freely about different subjects. Also, Twitter has a strong emphasis on trending events and real-time information. These features make it ideal for our use case. Twitter is also one of the few social media platforms that provides an open limited access to its application programming interfaces (API). We are using Twitter's Standard Real Time API [6] that streams one percent of the tweets in near real-time each second.

3.2 Our Approach and Choice

As already mentioned above, machine learning approach for sentiment analysis has many drawbacks. Lexical approach isn't perfect either, but it possesses certain qualities which make it more preferable for social media sentiment analysis. More specifically, it can be debugged, modified and extended to other domains more easily. Qualities like these are important because the Internet's "dictionary" grows day by day. Besides the fact that it is universally accessible from almost any country in the world, making it vastly multilingual, it also has some "inventions" of its own, like slang words, emojis and hashtags. It is therefore important to be able to keep up with the changes by modifying and extending the lexicon if need be. When choosing the approach we also kept in mind that we are going to analyze realtime data, meaning dozens of sentence-long texts per second. So speed was a priority too.

We chose VADER which is a numerical lexicon-based, sentence-level sentiment analysis library written in Python programming language. The main reason is that it covers the problems stated above. It is fast, has an extensive lexicon, is easily modifiable, and more suited to social media text analysis than many other libraries available today. It is based on python3 and is incorporated into Python's NLTK (Natural Language

[4]https://expandedramblings.com/index.php/twitter-stats-facts/
[5]https://www.internetlivestats.com/twitter-statistics/
[6]https://developer.twitter.com/en/docs/tweets/filter-realtime/api-reference/post-statuses-filter.html

ToolKit) NLP package [7].

The main target when writing VADER was creating a library that is easily modifiable works on social media and can be attuned to work in other domains, is fast enough to be able to analyze streaming-data without suffering from speed-performance trade-off, and is based on a high-standard, human validated, valence-based lexicon that can be easily extended and enhanced.

Based on the thoughts discussed above, we decided to stop our approach choice on sentence-level numerical lexicon-based sentiment analysis. The reason is simple. Most of the tweets usually don't exceed a sentence or two, because of the character limitation of 280 characters (140 Unicode characters) enforced by Twitter. This will allow VADER to keep up with the text length, and won't result in a performance penalty. We also use a numerical lexicon approach, because it provides more exact results on a continuous scale rather than choosing between predefined polarity labels. Using this approach will also allow us to use VADER's highly developed valence-based lexicon.

3.3 VADER's Lexicon

Obtaining a rich, high-standard, human validated lexicon is a time-consuming process which requires intensive labor work. People generally do not agree on emotion values of certain words. For example, one might say that the world "hatred" should have a polarity score of -0.9, while others may object, assigning it a score of -0.7. So entrusting the creation of a high-standard lexicon to one person can be prone to many errors.

VADER's lexicon was created as follows. At first, many of the best existing lexicons were examined, along with lexical features that are used in social media text were taken into consideration such as (emoticons, acronyms, slang, idioms). Then a wisdom-of-the-crowd approach was used by many qualified candidates using Amazon's Mechanical Turk[8] to estimate polarity scores for more than 9000 words on a scale of -4 to +4. Finally, for each of the words the mean and standard deviation (SD) of valence scores assigned by each candidate were calculated. The 1500 words with standard deviation bigger than 2.5 or mean valence equal to 0 were filtered out, leaving only 7500 words in the lexicon. For those the mean valence score was taken as a final one. Apart from the main lexicon, an emoji lexicon for Unicode emojis was created, mapping emojis to their descriptions.

[7]http://www.nltk.org/index.html
[8]https://www.mturk.com/

3.4 Features

3.4.1 Emoji and Emoticon Handling

Expressing emotions through text is not an easy task. Emoticons and emojis are hieroglyphic languages that were introduced as a way to help people express their emotions more easily. Emoticons were invented before emojis as a way of differentiating between a text that is a joke and a text that is non-joke. In 1982, a professor from Carnegie Mellon University called Scott E. Fahlman invented the emoticon[9]. Online forums were used for announcements and information sharing between students and faculty. Some people posted sarcastic jokes or made some funny remarks which some people would not understand or interpret. Mr. Fahlman proposed the usage of :-) emoticon for assigning a text as joke and :-(for non-jokes. The usage of emoticon increased over time and evolved into emojis which are pictorial representations of the emoticons.

Emoticons and emojis are widely used in twitter and play an important role in analyzing the sentiment of a tweet. Emoticons in VADER have predefined values in the lexicon. For example, the text ”The weather is great” results in a score of [’neg’: 0.0, ’neu’: 0.423, ’pos’: 0.577, ’compound’: 0.6249] where neg is Negative, neu is Neutral, pos is Positive, and compound is the overall normalized score which will be explained in section 3.4.8. Whereas, the text ”The weather is great :)” outputs [’neg’: 0.0, ’neu’: 0.297, ’pos’: 0.703, ’compound’: 0.7964] which has higher compound score. Emojis are handled differently, since they are Unicode representations of emoticons, all of them are placed in a file with their descriptions. For example the emoji 😁 has the description ”grinning face with smiling eyes”. The sentence ”The weather is great 😁” is transformed into ”The weather is great grinning face with smiling eyes” in VADER and when we analyze it the output is [’neg’: 0.0, ’neu’: 0.385, ’pos’: 0.615, ’compound’: 0.8625].

3.4.2 Punctuation

Punctuation is crucial for understanding the emotion and meaning of a text. Thus, it should be taken into account while analyzing the sentiment of a text. For example, the exclamation mark (!) is used to indicate strong feelings or to show emphasis without changing the semantic meaning of the text. The text ”The weather is bad” is different from ”The weather is bad!” or ”The weather is bad!!!”. The compound scores for three sentences are -0.5423, -0.5848 and -0.6571, respectively.

3.4.3 Capitalization

All capital words increase the sentiment of a sentence specially in the presence of non-capital words. ”The weather is GREAT” is definitely more intense than the ”The weather is great” so VADER takes all capital

[9]https://www.businessinsider.com/how-the-emoticon-was-invented-2015-9

lexicon words into account by increasing their valence scores by 0.733 which is the empirically derived mean sentiment intensity rating for all-capitalization of a word. The valence scores for the sentences above are ['neg': 0.0, 'neu': 0.383, 'pos': 0.617, 'compound': 0.7034] , ['neg': 0.0, 'neu': 0.423, 'pos': 0.577, 'compound': 0.6249], respectively. We can see that the capitalization of the only lexical word "great" has increased the compound score from 0.62 to 0.7.

3.4.4 Boosters

Booster words(also known as amplifiers and de-amplifiers) are words that impact the sentiment of a word by increasing or decreasing its intensity. For example, "The movie was extremely good" is more intense than "The movie was good", on the contrary, "The movie was marginally good" is less intense. The compound scores for three sentences are 0.4927, 0.4404, 0.3832, respectively.

3.4.5 Usage of Conjunctions

Conjunctions are parts of speech (POS) in English that are used to connect words, phrases, clauses, or sentences. Usage of a conjunction such as "but" can have a shifting impact on the sentiment polarity following it. For example, analyzing the text "The movie was great. The staff were so rude" results in ['neg': 0.229, 'neu': 0.486, 'pos': 0.285, 'compound': 0.204] whereas, the valence scores of the sentence "The movie was great, but the staff were so rude" is ['neg': 0.296, 'neu': 0.534, 'pos': 0.17, 'compound': -0.4385]. We can see that the first sentence has a positive value of 0.285 whereas, the second has positive value of 0.17 which means the valence was shifted due to the conjunction.

3.4.6 Slangs and Idioms

Acronyms and abbreviations, like "LOL" or "gr8" are considered part of slang, and when it comes to social media analysis, or generally any text on the internet, decision on processing or discarding slang can have a huge impact on the final polarity score. In VADER, slang words just like any other words are assigned corresponding score, using the techniques described in section 3.3.

Idiom processing is divided into two types, for the case when idiom words are in the lexicon and for the case when idiom words are not there. At the moment of writing of this paper, by default VADER has code implementation only for the first case but, it is incomplete, as besides considering the idiom itself, it also considers valence score of each of the word separately. Thus, for example, an idiom like "kiss of death" which has strictly negative meaning is getting scores like ['neg': 0.506, 'neu': 0.13, 'pos': 0.364, 'compound': -0.2732], where negative part comes from the word "death" and positive part comes from word "kiss", indicating that they are treated separately. Implementation for idioms containing non-lexicon words doesn't exist at all, so a positive expression like "cut the mustard" which means "to succeed; to come up to expectations." gets scores of ['neg': 0.512, 'neu': 0.488, 'pos': 0.0, 'compound': -0.2732], resulting in a negative compound score, which doesn't correspond to reality. Here the problem is that "cut" has valence of -1.1 while "mustard" is not in the lexicon, so it's discarded, leaving the only contributor to the total score the word "cut", thus resulting in negative score.

3.4.7 Preceding Tri-gram Negation Check

Every word from the lexicon in the processed text is being checked for a negation three words before it. The author of the library states that by checking preceding tri-gram sentiment for each lexical feature, 90% of the negation cases are caught. For example, "The food wasn't as good as they kept telling us" here the word good has a 1.9 valence score in the lexicon but after negation check the score flips by a value of -0.74 (which is the empirically derived mean sentiment intensity rating for negative scalar) becoming -1.406 and after normalization the valence scores are as follows ['neg': 0.211, 'neu': 0.789, 'pos': 0.0, 'compound': -0.3412].

3.4.8 Valence Score Normalization

A sentence can have positive, negative, and neutral meaning at the same time but overall it has only one of the classifications. In VADER after all the checks such as check for boosters, negations, conjunctions, etc., are done and polarity scores are determined, the final step is the normalization of the polarity scores. The normalization is needed because the lexical features in VADER have a range of [-4:4] for valence scores whereas, polarity scores have a range of [-1:1]. After each analysis an output of four different scores are given, the first three are negative, neutral, and positive and none of them are a negative number because they represent a sentiment ratio of the whole analyzed text. For example, a negative score of 0.5 and positive score of 0.5 indicates that the text was 50% negative and 50% positive. The final and most important score is the compound score which is calculated by the sum of valence scores of each word from the text that is being processed that are in the lexicon, attuned to the rules, and then normalized by a normalization formula to be between -1 (most negative) and +1 (most positive). The score ranges for normalized compound score is as follows

Sentiment	Score Range
Positive	0.05 to +1
Neutral	-0.05 to +0.05
Negative	-1 to -0.05

Table 1: Score Ranges with Neutrality Threshold

The normalization formula is $f(x) = \frac{x}{\sqrt{x^2+\alpha}}$ where x is the sum of sentiment scores and alpha is a parameter that is used to approximate max expected value. Alpha is by default set to be 15. The normalization graph is represented below

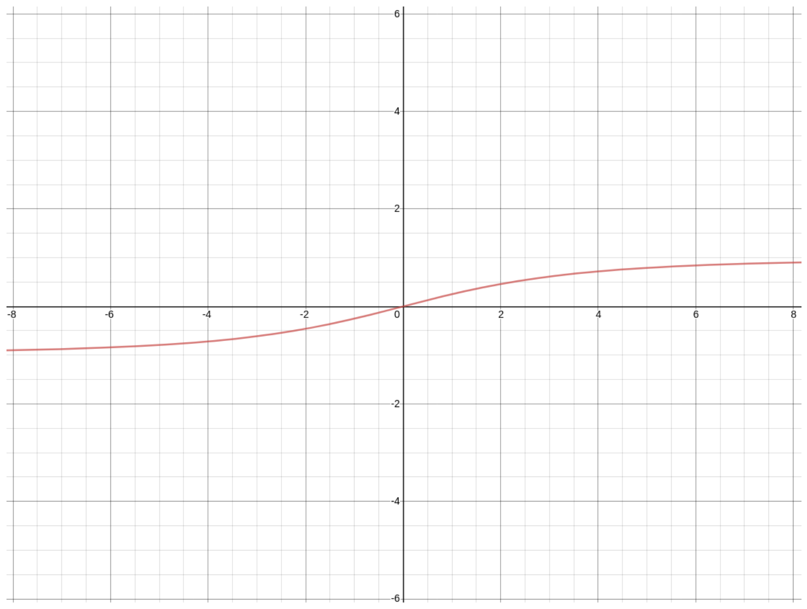


Figure 2: Normalization formula graph

We see that as x or the sum grows larger it gets closer to -1 or 1 which means that the longer the analyzed texts the closer the final polarity score gets to -1 or 1. Thus, VADER works best with short texts such as tweets.

For example, the text "The cafeteria staff are great. The food is meh. The environment is noisy" has three sentences, the first one is positive, the second one is almost neutral, the third one is negative. Analyzing that text gives us the following output

Metric	Score
Positive	0.227
Negative	0.166
Neutral	0.608
Compound	0.4767

Table 2: Classifications Metrics and Their Corresponding Polarity Scores

Polarity scores for sentences are determined and the final compound score of the whole text is 0.4767 which means that overall it is more positive. We see that 60% of the text is neutral, 22% is positive and 16% is negative.

3.5 Improvements

We realized that VADER has a lot of space for improvements. Even though, the code was fast, it was really difficult to understand it as it was badly written. We tested and concluded that the code was not working correctly with idioms and some sentences with complicated negation cases.

3.5.1 Text cleaning

The huge number of tweets received through the live feed, makes it essential to filter the tweets which are potentially useless and can create unnecessary noise in the results. Among those are the tweets which include only URLs and/or only multimedia content, the ones which consist only of one letter or words consisting only of one letters, and the ones which include only hashtags. Some of the tweets are discarded and some of them are pre-processed before analysis using VADER. The reason for this is to avoid huge number of neutral tweets, which in the long run would have a big impact on the overall picture of the analysis. The primary way of processing the strings is by using regular expressions, using built in string methods of Python or the combination of both. URLs are filtered first. Anything matching the regular expression corresponding to a URL is cut out of the text. The result is then checked for the availability of a text, in case if tweet is only URL or multimedia, like images and videos it is discarded. Also it is checked whether or not text is meaningful, i.e. its words contain more than one letter, because VADER classifies text like this as neutral.

If the conditions mentioned are satisfied then the text is processed for hashtags and user mentions. By default, VADER doesn't process mentions at all, which contributes to the increase in the neutrality of the tweet, and discards words with hashtags but they play an important role in identifying the sentiment of a tweet. All mentions are filtered out in the very first stage. After mentions, hashtags are processed. They usually occur in three places in a tweet, in the beginning, inside the main text, and in the end of the text. If there are words with hashtags in the beginning of the text, the hashtag symbols preceding words are removed and a dot is added to the final word with a hashtag in the beginning of the text. If there are words with hashtags in the end of the text, the same method for hashtag removal is applied in reverse. If there are words with hashtag inside the main text, they are simply removed as they consist part of the sentence. For example, the text ”#Great #Vacation The city of Paris is a #beautiful destination #Amazing #View” is transformed to ”Great Vacation. The city of Paris is a beautiful destination. Amazing View.”. Otherwise, it would have been treated as ”The city of Paris is a destination” which has no emotional sentiment.

3.5.2 Advanced Idiom Checking

As discussed above VADER has an incomplete implementation for checking idioms. It partially is a result of the code structure of VADER, which splits the text into separate words and start processing each separately, making idiom detection hardly possible. The improved implementation splits the text in a way that idioms are preserved, making valence score detection of idioms easier. The resulting valence score is then added to the total sentiment score like any other word. This also allows to avoid processing each of the words separately, which has a significant impact on the final score. Improved idiom checking has desired output. For comparison, phrases like ”kiss of death” and ”cut the mustard” are now classified correctly. Compound score for ”kiss of death” dropped from -0.2732 to -0.3612, while having only negative sentiment. For ”cut the mustard” compound score rose from -0.2732 to 0.4588, while having only positive sentiment. The results are generalized in Table 1.

Idiom	Before	After
kiss of death	-0.2732	-0.3612
cut the mustard	-0.2732	0.4588

Table 3: Comparison of default and improved idiom checking

As we see advanced idiom checking is an improvement from the default one, as it manages to span all the idioms available in the dictionary, assigning them their correct valence.

3.5.3 Preceding N-gram Negation Check

After checking the code for negation check we realized that it can be improved both in terms of result and performance. Instead of doing a preceding tri-gram negation check we decided to identify a negation word and have a specific punctuation list that would help us identify where in a sentence the negation ends. This would improve both the performance as we would not have to loop unnecessarily over each lexical feature three times, and the result as we would capture negation cases that precede a lexical feature by 3 or more words. For example, the text ”The weather here isn’t really all that great” is clearly a negative one. Checking it with the default VADER gives us the following output [’neg’: 0.0, ’neu’: 0.616, ’pos’: 0.384, ’compound’: 0.6557]. Whereas, the improved version outputs [’neg’: 0.333, ’neu’: 0.667, ’pos’: 0.0, ’compound’: -0.5407]

4 Case Studies

One of the motivating factors of this project was to create a world happiness map where we can see the happiness index of each country live. The first case study showcases our main goal where we processed streaming tweets, analyzed them and compared the results to world happiness index of countries. For the second case study we created a survey where people rated the sentiments of sample tweets and compared the survey results with different libraries.

4.1 Case Study 1

We have built a world happiness map[10] using a diverse technological stack such as Tweepy[11] which is a wrapper that makes interacting with Twitter API easier, Redis[12] as NoSQL database for storing the results from analysis of sentiments of texts, and Plotly Dash[13] for the representation of the countries with their data on the world map.

We used the world happiness map to analyze world happiness using data gathered from sentiment analysis performed on tweets. For this purpose we analyzed 1000000 tweets gathered from realtime feed during period of 14-15 May, 2019. As we mentioned above, on average twitter provides about 60 live tweets per second. The measurements were taken twice with different calculation techniques, however with identical conditions for both cases, such as computational power and network connectivity.

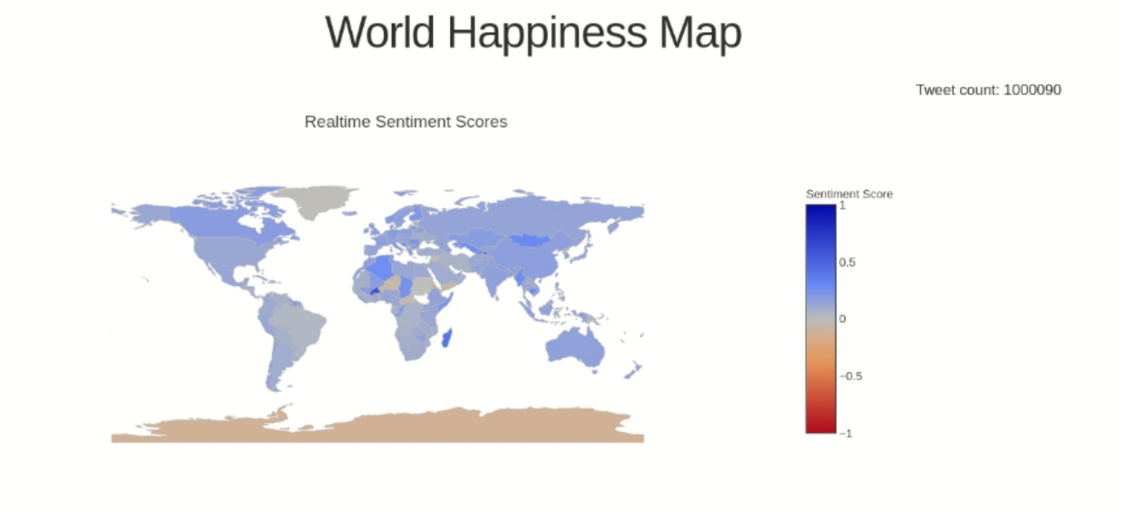
First time we accounted for the neutral tweets, meaning that tweets with neutral score equal to 1.0 were counted too. Second time we processed, but didn't count those, meaning that they were not shown on the map, so we got different results. The reason behind taking different approaches is that even though we did our best to avoid meaningless tweets, it's impossible to avoid them at all. For example, we used Twitter's streaming API with a filter that included a bounding box geocode of [-180, -90, 180, 90] which resulted in gathering of geolocated tweets from all around the world, and a filter for English language. When we experimented our world happiness map, we realized that there are a lot of countries that have neutral tweets, the reason was that a lot of people write in their own language with English letters. Processing these tweets resulted in increase in overall neutrality of the tweets for each country.

Thus, the time required to reach 1000000 tweets for two cases was different. In the first case it took about 13 hours to get and process this many tweets, while in the second case it was about 23 hours. Simple calculation shows that neutral tweets represented about 10 hours worth of data, which is a significant amount.

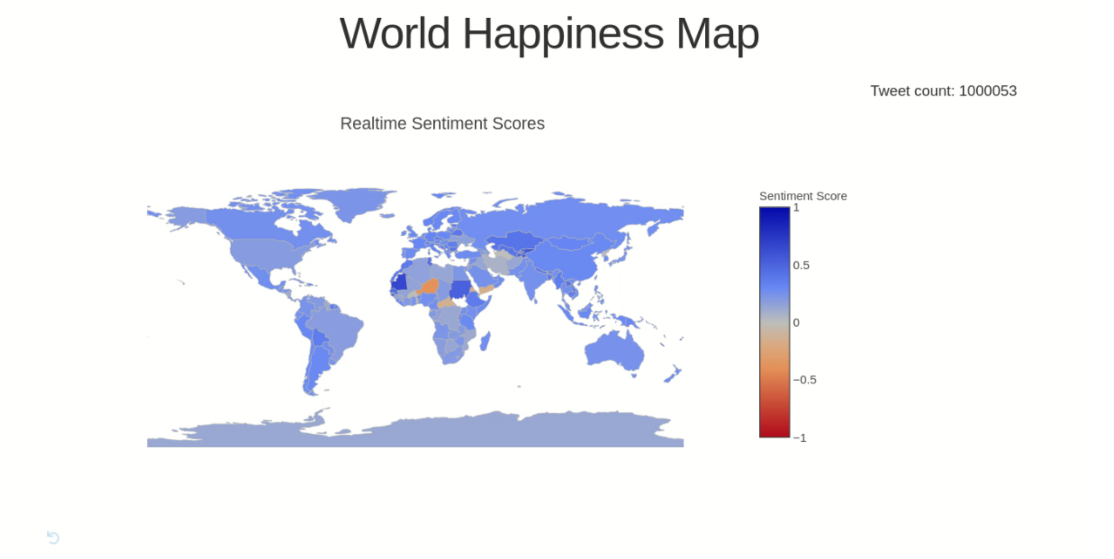
We can also count how much tweets we processed per second. In the first case it was $1000000/13*3600 \approx$

[10]https://github.com/NaregB/world-happiness-map
[11]https://www.tweepy.org/
[12]https://redis.io/
[13]https://plot.ly/products/dash/

21 tweets/sec, which is about one third of the 60 tweets. In the second case we reached million in 23 hours, so we had $1000000/23 * 3600 \approx 12$ tweets/sec. As we see, out of every 21 tweets 9 were neutral, accounting for about 40% of all tweets. So they tended to lean the happiness map towards more neutrality. The results are summarized in Figure 3. As we see colors in the first case are more faded, indicating overall neutrality of the world, while in the second case we find more vibrant colors, with overall happiness leaning towards the positive end in the developed countries, with some negative and neutral spikes over the developing countries in the Global South.



(a) Results with neutral tweets



(b) Results without neutral tweets

Figure 3: World happiness map

We compared the results from World Happiness Index(WHI)[14], analyzed tweets including neutral ones,

[14]https://countryeconomy.com/demography/world-happiness-index

and processed tweets excluding the neutrals. The table below represents the top 20 happiest countries according to results from WHI, VADER results on all tweets and VADER results on non-neutral tweets. With very few exceptions (like Denmark), there is almost no direct correspondence between the WHI data and sentiment-based tweet data, as the WHI data is a measurement taken for a whole year based on several features, while our data was gathered in less than a day and is based purely on text sentiments. Despite this, we see a general pattern in this list, with mainly Northern hemisphere countries and other developed countries taking top positions, while least developed Southern hemisphere countries either are absent at all or have lower positions.

Ranks	WHI	VADER w/ Neutral	VADER w/o Neutral
1	Finland	Switzerland	Switzerland
2	Denmark	Denmark	Denmark
3	Norway	Germany	France
4	Iceland	Poland	Italy
5	Netherlands	Sweden	Sweden
6	Switzerland	Ireland	Belgium
7	Sweden	UK	Greece
8	New Zealand	Norway	Netherlands
9	Canada	Belgium	Ireland
10	Austria	France	UAE
11	Australia	Greece	Indonesia
12	Costa Rica	Finland	Turkey
13	Israel	UAE	Russia
14	Luxembourg	New Zealand	UK
15	UK	Italy	China
16	Ireland	Netherlands	Singapore
17	Germany	Russia	South Korea
18	Belgium	Honk Kong	Spain
19	USA	Canada	Portugal
20	Czech Republic	China	Philippines

Table 4: Comparison of Rankings of WHI, VADER results with and without neutral Tweets

4.2 Case Study 2

As a second case study we decided to compare human classification results with VADER and other NLP and/or sentiment analysis libraries written in python.

For this purpose we took a wisdom-of-crowd approach similar to the one used for building the lexicon of VADER. A tweet set consisting of 100 tweets was considered. These tweets, however, were not chosen randomly. They were rather carefully examined and chosen out of hundreds of others, to avoid confusion for the future work. URL-, media- and hashtag-only tweets were filtered out. Only tweets which could be assigned proper polarity scores were left. Those tweets were then distributed among 17 volunteer survey takers, who agreed to assign those tweets corresponding polarity scores on scale of -4 (most negative) to +4 (most positive). It's important to note that scores were assigned out-of-context, meaning the participants didn't know the intention behind writing the tweet, so the scores given were purely sentimental. The results were then gathered, mean and standard deviation were computed for polarity scores of each of tweets. Polarity scores were rounded to nearest number and those equal to 0 were discarded. Also, tweets with SD < 2.5 were discarded, leaving us with a set of 58 tweets. The mean of all the scores was chosen as a final sentiment score. A similar tested approach was used by VADER, that's why our methods can be considered valid. Scores gathered in this way are one of the variables of comparison used in this case study.

The next variable is the compound sentiment scores assigned to each of the tweets by VADER's default implementation, without any tweaks and improvements introduced by us. Another variable is the modified version of VADER[15].

Besides the human based scores and VADER's default and improved implementations, we used another popular natural language processing library written in python - TextBlob[16]. Being a general NLP library, it also supports sentiment analysis. It has two methods for calculating the sentiment. One of the methods, called Pattern Analyzer (PA) is similar to VADER and uses lexicon-based approach. However, TextBlob's lexicon[17] is considerably smaller, consisting of about 2900 words and doesn't handle some words/symbols, like Unicode emojis. Another approach that TextBlob uses is machine learning based classification, called Naive Bayes Classifier (NBC). Rather than assigning direct score to a text, it classifies text as positive or negative, by calculating the probability of text belonging to one of those categories. Like any classifier, it needs a training set, so it uses NLTK's movie review corpus (a collection of texts with movie reviews). Training time required for this approach is enormous, though, so it suits better for analysis of text longer than sentence or two-sentence long, and absolutely doesn't work for real time analysis for dozens of short tweets per second. For comparison reasons, however, we decided to try this approach and see if the time spent on training the model is indeed justified.

Out of 58 tweets, 33 were classified as positive, and 25 were negative by survey results[18]. Both VADER and modified VADER classified 35 tweets as positive and 23 as negative. TextBlob's Pattern Analyzer classified 41 tweets as positive, 8 as neutral and 9 as negative, whereas TextBlob's Naive Bayes Classifier classified 39 tweets as positive, 19 as negative. Out of 33 positive tweets from survey results, 31 of them were classified positive by VADER, and modified VADER, whereas, TextBlob's Pattern Analyzer classified 30 of them as positive, 2 of them as negative, and 1 as neutral, and its Naive Bayes Classifier classified 25 of

[15]https://github.com/NaregB/vaderSentiment
[16]https://textblob.readthedocs.io/en/dev/
[17]https://github.com/sloria/TextBlob/blob/dev/TextBlob/en/en-sentiment.xml
[18]https://github.com/NaregB/world-happiness-map/blob/master/Survey$_Results.csv$

them as positive, 8 as negative. For negative tweets, VADER and modified VADER both classified 21 out of 25 negative tweets as negative, and the other four as positive whereas, TextBlob's PA classified 8 out of 25 negative tweets as negative, 7 as neutral, and 10 of them as positive and its NBC classified 11 out of 25 negative tweets as negative, 14 as positive.

Tweets	Survey Results	VADER	Modified VADER	TextBlob PA	TextBlob NBC
Positive	33	93.9%	93.9%	90.9 %	75.7%
Negative	25	84%	84%	32%	44%

Table 5: Comparison of Correct Classifications by Percentage

The average polarity score of positive tweets by survey results was 0.7272, whereas for VADER it was 0.6005, for modified VADER it was 0.6121, and for TextBlob's PA it was 0.5001. The average polarity score of negative tweets by survey results was -0.6528, while for VADER it was -0.3805, for modified VADER it was -0.4108, and for TextBlob's PA it was -0.0263.

Tweets	**Count**	**Survey Results**	**VADER**	**Modified VADER**	**TextBlob Pattern Analyzer**
Positive	33	0.7272	0.6005	0.6121	0.5001
Negative	25	-0.6528	-0.3805	-0.4108	-0.0263

Table 6: Comparison of Average Polarity Scores

Table 4 shows that all libraries perform better for positive tweets than negative ones compared with survey results. The difference between average polarity score of positive tweets for modified VADER and survey results is marginal, whereas, for the case of negative tweets the difference is bigger. TextBlob performs worse than VADER both for positive and negative tweets. Modified VADER performs better both for positive and negative cases as the average polarity scores for these cases are closer to survey results.

Mean score comparison, however, isn't enough to understand how good each of the libraries performs against the humans, and how closely they are correlated between each other. The heatmap constructed on the correlation matrix of the results, presented in Figure 4 reveals that despite the modified VADER and VADER performed similarly, there is still a difference. Their correlation is close to 98.72%. As such, the modified VADER got the closest to the human results, showing a correlation of about 81.36%, while VADER's default implementation has a correlation of about 80%. TextBlob's Pattern Analyzer by far shows the worst performance, by getting a correlation of only 67.10%, which is quite logical, considering its small lexicon.

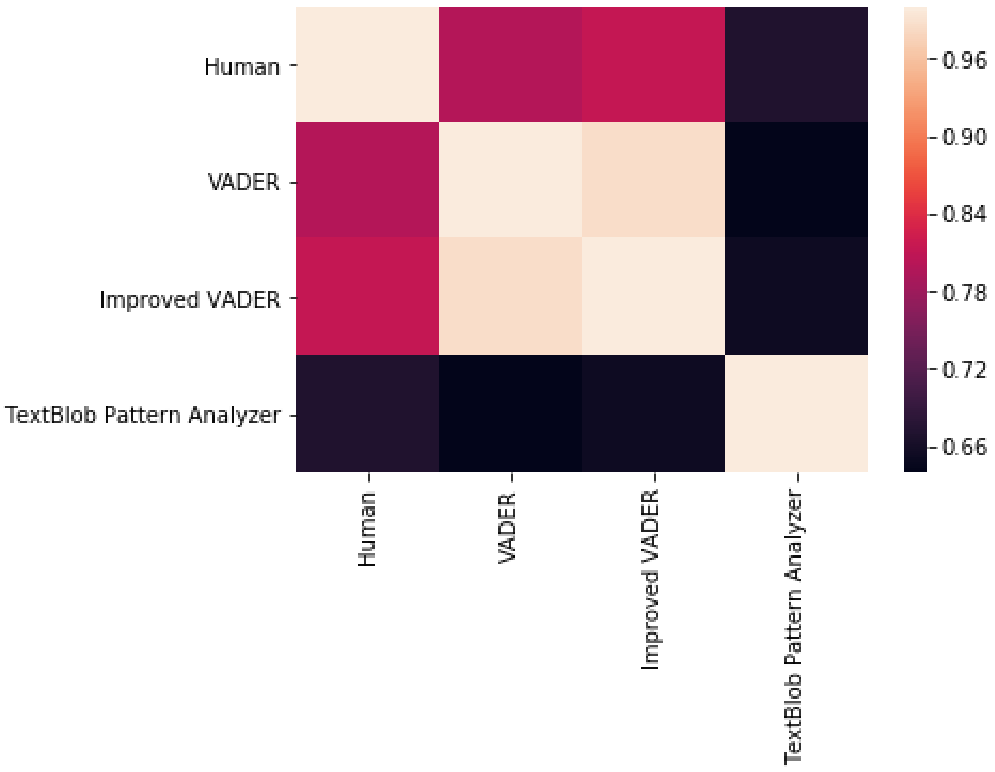


Figure 4: Correlation between library results and human assigned sentiment scores.

5 Conclusion and Future Work

Sentiment analysis is a field of many open-ended problems that are yet to be solved. We have researched many libraries and found that VADER is the most suitable library that provides fast and highly accurate analysis of sentiments for texts. We examined some lexical features that are important and should be considered especially for social media sentiment analysis. We have processed tweets before analyzing them, removed URLs and included words with hashtag in our text since they add sentimental value to tweets to make the results more accurate. Moreover, we have improved some aspects of the algorithm of VADER such as idiom and negation checking.

For future work, we are planning to implement an algorithm for separating concatenated words that would improve the overall analysis since a lot of tweets contained hashtags with multiple concatenated words which added no sentimental value such as ”#mentalhealthawarenessweek” instead of ”#mental #health #awareness #week”. We are planning to implement a stemming[19] algorithm that is fast enough so that we can include it in our text processing pipeline because stemmed words are much easier to analyze especially for idiom checking. Finally, it is crucial to implement an algorithm that can detect transliterated texts and discard them because even though we filtered tweets by English language, we kept receiving a lot of tweets transliterated to English.

[19]https://searchenterpriseai.techtarget.com/definition/stemming

6 References

[1] Statistical analysis of tweets about Trump and Hillary Clinton during 2016 elections.

https://monkeylearn.com/blog/trump-vs-hillary-sentiment-analysis-twitter-mentions

[2] Mining Opinion Features in Customer Reviews. Minqing Hu and Bing Liu. Department of Computer Science. University of Illinois at Chicago.

https://pdfs.semanticscholar.org/ee6c/726b55c66d4c222556cfae62a4eb69aa86b7.pdf

[3] Twitter users statistics.

https://expandedramblings.com/index.php/twitter-stats-facts/

[4] Twitter live usage statistics.

https://www.internetlivestats.com/twitter-statistics/

[5] Overview of Sentiment Analysis

https://monkeylearn.com/sentiment-analysis/

[6] Sentiment Analysis Types, Tools and Use-cases

https://www.altexsoft.com/blog/business/sentiment-analysis-types-tools-and-use-cases

[7] Hutto, C.J. Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf

[8] VADER's library.

https://github.com/cjhutto/vaderSentiment

[9] Modified version of VADER.

https://github.com/NaregB/vaderSentiment

[10] VADER's library.

https://github.com/cjhutto/vaderSentiment

[11] Survey results used in case study 1.

https://github.com/NaregB/world-happiness-map/blob/master/Survey$_Results.csv$

[12] The code for World Happiness Map used in case study 1.

https://github.com/NaregB/world-happiness-map

[13] World Hapiness Index.

https://countryeconomy.com/demography/world-happiness-index

[14] TextBlob documentation.

https://textblob.readthedocs.io/en/dev/