Sentiment Analysis on Twitter Data Tracking Perceptions Over Time

Jasper Drumm, Timothy Machnacki, William Morland, Evan Parres, Alexander Pohlman

{jasperd, tmachnac, wmorland, evparres, aohlman}@umich.edu

Abstract

Companies are greatly interested in public perception of the business or its actions, particularly whether that perception is positive perception. We create a tool that quickly allows a company to judge just that, providing sentiment analysis on Tweets in a recent timespan. First, we train multiple models, using combinations of features and machine learning classifiers to find which give the best performance. We also create a web application that retrieves tweets matching a query in a time frame and use our classifier to provide the sentiments of the tweets to inform company perception.

1 Project Description

1.1 Problem

Our project is doing sentiment analysis on Twitter data to see how perceptions of a company change over time. This will allow us to see how positive events such as a product launch or marketing campaign or negative events such as data breaches affect the public conversation surrounding a company.

Companies inherently have a strong vested interest in public perception, which would make an accurate sentiment analysis of public opinion to be of great practical use. How people view a company is a strong influence on how the company makes money and succeeds. Positive opinion: stocks prices go up, profit margin increases, and the company grows. Negative opinion: quite the opposite, money is lost, and the company stagnates. By being able to have a good judge of what kind of view the public has toward their business, companies can swiftly change course. If there is negative talk about the company, they can quickly change course to negate the damage. Or if there are positive sentiments about the company, they know that their actions are having positive impacts and to keep up the same trend.

Other than general, longer-term scales in company perception, businesses also would be interested in shorter term public responses to recent events. For example, companies could use our model to see how a particular marketing campaign is doing, whether the public likes a new logo, and more. Let's look at a specific case. Suppose Apple has just released the latest iPhone and they want to see what people think about it. One option is that the company can waste money conducting expensive public surveys which can take a lengthy time before the results and a wide view of public sentiment will even be determined. Or then can quickly use our model and see the latest tweets about the company and product and whether they are positive or negative. Our project will also be of potential use to others who are doing market research, such as those in academia and government organizations.

1.2 Proposed Solution

To meet this much needed demand, we build a program that utilizes the Twitter API to gather tweets about a given company over a given period of time, then uses a machine learning model to classify these tweets as positive or negative. We will then be able to see how the number of positive and negative tweets changes over time and thus gauge public sentiment about the company.

Because social media platforms like Twitter are so popular, there is robust prior research in related fields such as NLP and sentiment analysis. This research includes a 2017 study that sought to gauge specific Twitter users' opinions on companies and brands using a collection of their Tweets, among other relevant works. We plan on

using aspects of these studies in our own work when relevant and feasible.

After researching which dataset best fit our needs to train the model, we decided that the Sentiment140 dataset from Kaggle was the best option. This is because Sentiment140 contains Tweets about a broad range of products, brands, and companies, so our model will work for a wide range of subjects. In addition, Sentiment140 is massive (containing 1.6 million Tweets) so our model should be fairly accurate after training on it.

There will be two main aspects to our project: training the classifier using Sentiment140 and retrieving and classifying Tweets from both the Twitter API and from company-specific Twitter datasets (such as the Kaggle datasets for US Airline sentiment).

To train the classifier, we preprocess the Sentiment140 dataset by tokenizing it in a Twitter-specific manner, removing punctuation and stop words, and apply stemming We will test all combinations of features and a classifier in our model to see which one yields the best results. For features, we will explore building matrices from tf-idf values, raw term frequencies, and BERT embeddings to take advantage of deep learning. Next, we will use this matrix to train a variety of classifiers such as Naïve-Bayes, Linear SVM, and Logistic classification. Finally, we will use whichever combination performs the best when classifying Tweets.

After our model is complete, we will build a web application that, given a query and a timeframe, retrieves Tweets from the Twitter API and measures the sentiment of those Tweets using our model. Given our current API license, our system will only be able to gauge sentiment from the last 7 days, which should still be of great use to companies and other interested parties. Our project will present its results in a neat graphical format that will make human interpretation easy and paint a clear picture of company perception.

2 Related Work

Alongside the rise in popularity of micro-blogging platforms and social media, sentiment analysis too has become a topic of increased interest in machine learning and the NLP research space as a whole. There has been substantial prior research in sentiment analysis, especially in domains consisting of larger content such as movie reviews, product reviews and blog posts. More recent advancements in sentiment analysis have shifted focus to analyzing platforms like Twitter in which posts contain less content. Furthermore, micro-blogging platforms often contain less grammatically accurate language, slang, and the use of emotes and hashtags, presenting new challenges to language processing.

A paper from the Department of Computer Science at Cornell University in 2002 analyzed the effectiveness of three machine learning techniques on the IMDb archive of movie reviews: Naïve Bayes, maximum entropy, and support vector machines. The findings showed that the three models were able to achieve 81%, 80.4%, and 82.9% accuracy respectively on unigrams (Pang, Lee, 2002).

Building off the Cornell study, in 2009 a team from Stanford University analyzed the sentiment of tweets in relation to a query term. They compared different machine learning algorithms (Naïve Bayes, maximum entropy, and SVM) using different features (unigrams, bigrams, both, and parts of speech). For training data, they used tweets that contained emoticons like ‘:)’ or ‘:(‘, labeling those tweets as positive or negative respectively. They found this training on emoticons to be an effective method for distant supervised learning. This data was compiled in the Sentiment140 dataset (Go et al., 2009).

From the University of Indonesia in 2015, researchers used Twitter data to analyze sentiment toward mobile phone providers. They compared Naïve Bayes against decision tree and SVM classifiers, finding, as previous studies had, that SVM performed the best. Using sentiments of tweets, they measured Net Brand Reputation to measure customer loyalty. They found that XL Axiata had the best reputation out of the companies they analyzed. They also created a live dashboard of this NBR score along with common keywords in tweets to help the Mobile Phone Providers keep track of how the public perceived their companies (Vidya et al., 2015).

In 2015 Kharde et. Al with the University of Pune conducted an analysis of various methods for sentiment analysis. In addition to the typical machine learning approaches, they also evaluated lexicon-based approaches including those based on WordNet and corpus-based methods based on Latent Semantic Analysis. These were found to give fairly high accuracy in some cases and don’t

heavily rely on human labeled data. They also found that bigram features tend to work best compared with other features (Kharde et al., 2015).

More recently, in 2017, researchers attempted to analyze the opinion of Twitter users towards different industries and consumer brands by using sentiment analysis of their tweets. They used a Long Short Term Memory implementation of a Recurrent Neural Network. Moreover, they analyzed the opinion of 19 million Twitter users towards 62 popular industries, consisting of 12,898 enterprise and consumer brands, totaling a dataset of 330 million tweets over one month. They found that users tend to feel most positive towards manufacturing and most negative towards service industries. They also concluded that users tend to be more positive or negative with regards to a brand than in other niche spaces on Twitter (Hu et al., 2017)

Lastly, a group of researchers from the Association for Computational Linguistics analyzed target-dependent Twitter sentiment classification in 2011. Different from other sentiment classification approaches, they used a target-dependent system in which they considered subjects of tweets that were implicitly related to a company. According to their experimental results, target-dependent classifiers significantly outperformed previous target-independent models. Furthermore, they utilized a graph-based optimization approach to take tweets related to a given tweet into consideration. They found that graph-based optimization significantly improves sentiment analysis as well (Jiang et al., 2011).

3 Data Selection

In our preliminary search for datasets of Tweets, we looked for two main criteria: Tweet sentiment must be target-dependent as well as effectively annotated and preprocessed. We included only datasets with target-dependent sentiment annotation to align with the aim of our overall project, which is of course to perform Tweet sentiment analysis based on a query. Additionally, it was important to us that candidate datasets were annotated well with clear and accessible methods to indicate polarity.

Based on these principles, our research led to a handful of strong potential datasets divided into two categories:

1. Sector or company-specific Twitter sentiments such as select Kaggle datasets on US Airlines and Apple Computers
2. Broader Kaggle datasets containing Twitter sentiments across a multitude of products, brands, and companies

3.1 Sentiment140

For training purposes, we decided to utilize the second option of broader datasets including Twitter sentiments regarding multiple companies, specifically the Sentiment140 dataset on Kaggle. We reasoned that company or sector-specific data likely contains vocabulary and sentiment "buzzwords" more specific to that company or industry than others. Seeing as our project goal is to allow for users to query across sectors, products, and companies, we believe the larger, more general Sentiment140 dataset provides the most appropriate data.

In addition, several aspects of the Sentiment140 dataset make it advantageous for our purposes. First, it is significantly larger than the narrower datasets, containing 1.6 million Tweets compared to the ~3800 and ~14,600 Tweets contained in the Apple Computers and US Airlines datasets. We believe this gives us a greater ability to evaluate sentiment of Tweets from a wide range of queries.

Moreover, the Tweets contained in Sentiment140 are each annotated with polarity ranging from negative (0) to neutral (2) to positive (4) with respect to a query term, and the dataset is in *.csv format with fields including target (polarity), Tweet ID, Tweet timestamp, query, Tweet owner, and Tweet ext. These attributes all fit our needs well regarding training our model.

One typical example from our dataset is the following entry:

> *[0, 2055843801 Sat Jun 06 10:03:24 PDT 2009, NO_QUERY, EmptyIsAwesome, @google Why does it take so long to get a check from you? You're like the alcoholic Uncle that mails my bday check 2 weeks late].*

This data point is a prime candidate for training our system. We see a customer's apparent reaction to the company Google with a negative sentiment label. Due to the meticulous data parsing, datapoints like these are easily integrated into our system.

3.2 US Airline Sentiment

Though we have selected the broader Sentiment140 dataset for training purposes, we believe the sector and company specific datasets are still valuable for our project. To simulate domain-specific testing (such as tweets about a specific company, product, or industry), we utilize the US airline twitter sentiment dataset. This dataset contains around 14,000 tweets already annotated with various fields like id, sentiment label, target airline, username, tweet text, etc. For our purposes, we only use the text and sentiment label fields. A typical example is as follows:

> *[…, negative, …, Virgin America, …, jnardino, …., @VirginAmerica seriously would pay $30 a flight for seats that didn't have this playing.]*

For additional processing, as sentiment is labeled with a string, we convert this to numerical values: 0 for negative, 2 for neutral, and 4 for positive.

4 Methodology

At a high level, the end goal of our program is to take a user query consisting of a company name or product, retrieve relevant tweets over a certain time frame, and then classify the tweets as positive, neutral, or negative so that we can see how the perception of the target changes over time. The formulation and execution of our system can be divided into three core components: the training of our machine learning model, testing, and the retrieval and classification of new tweet data via user-interface.

4.1 Training and Model Selection

As mentioned previously, we used the Sentiment140 dataset to train our system. However, we only used a subset of 4,000 tweets with an even class distribution so as not to skew evaluation metrics. Since BERT requires significant computing overhead, it was necessary to use a relatively small amount of training data for model selection.

For preprocessing the data, we sequentially applied tokenization, stopword removal, and stemming. To do so we utilized Python's NLTK library which provided useful methods. NLTK was able to specially tokenize "@" and "#" characters used for twitter handles and hashtags. NLTK also provided a straightforward stopword removal method as well as a Port Stemmer method. It is important to note that preprocessing was not used with BERT because BERT requires raw text for its own tokenization.

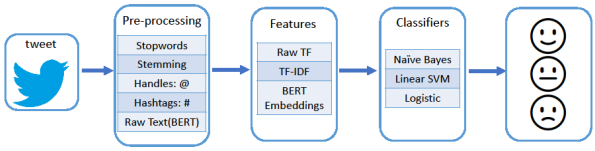

Figure 1: Classification system pipeline

Furthermore, we decided to implement three feature extraction methods: raw term frequency, tf-idf, and BERT embeddings. The raw tf and tf-idf matrices were generated with sklearn's CountVectorizer and TfidfVectorizer methods, respectively. We were able to generate a vanilla BERT model using PyTorch and HuggingFace's transformers, as laid out in "A Visual Notebook to Using BERT for the First Time" and "Text Classification with BERT in PyTorch".

We then used the resulting matrices to train three classifiers: Naïve Bayes Complement, Linear SVM, and Logistic classification, each of which is provided by sklearn. We then proceeded to use all combinations of feature extraction methods and classifiers with the exception of Naïve Bayes-BERT embeddings; negative embeddings prevent the model from working properly, and the Naïve Bayesian assumption of independence invariant would be violated, rendering the model inappropriate.

It is here that we should note our shortcomings in working with BERT. We have written code to fine-tune and optimize BERT-Base on the Sentiment140 dataset using PyTorch, but we did not have the necessary computing power to run it readily available to us. Furthermore, we looked into alternatives such as using CUDA toolkit or PyTorch on a cloud computing resource such as AWS but lacked the resources and ample time to implement such features. Given the immense computing overhead required to use our own BERT implementation end-to-end, we resorted to using PySentimiento for testing and for our front-end web application that necessitates rapid classification based on user input. PySentimiento is a library with a pre-trained BERT model for sentiment analysis of tweets, generated using similar methods as we would have done given more computational resources and time.

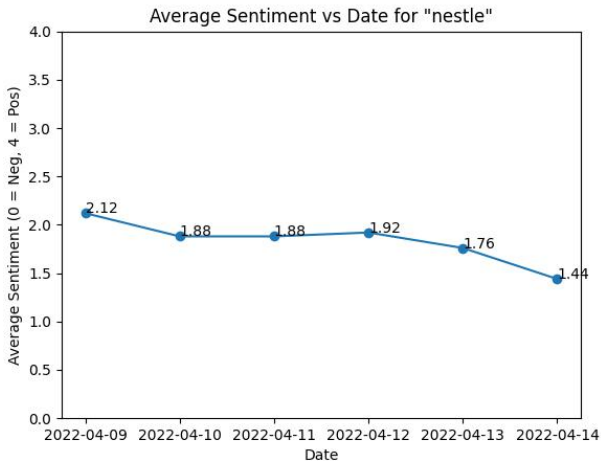

Figure 2: This graph shows average sentiment by date given query "nestle" and period of one week

4.2 Testing and Evaluation Methods

We used 5-fold cross-validation to evaluate the performance of combinations of feature extraction methods and classifiers on a 4,000-tweet subset of the training data. Furthermore, we used accuracy as our preferred metric; accuracy is an appropriate metric in this case since there are even class distributions in the training data. Following the training of our system and evaluation of our systems on the training data, we used a 2,000-tweet subset of the US airline dataset to evaluate performance on new domain-specific tweets. We measured the performance of a term frequency-Naïve Bayes model and PySentimiento and generated results in a formatted table. PySentimiento performed multiclass classification (0-negative, 2-neutral, 4-positive), and Naïve Bayes performed binary classification (0-negtive, 4-positive). We used F1-score as the preferred performance metric in this case due to the uneven class distribution in the subset of test data. Following our evaluation, we have the optimal system to use for our front-end web application.

4.3 Application

After our model evaluation, we used Flask to build a web app with a user-friendly interface. After the user inputs a target query and a requested time frame, the application retrieves relevant Tweets via the Twitter API and classifies them. Subsequently, the app uses the classifications to generate a line graph displaying the change in sentiment over the requested time period. An example of such a generated line graph is displayed in Figure 2. The application can also display graphs depicting tweet counts by category (positive, neutral, negative) and tweet counts by date. Given our standard API license, our system can only use tweets from the last 7 days, which, while less than ideal, nevertheless is still valuable to companies.

5 Results

The results from our 5-fold cross-validation evaluation on training data are displayed in Figure 3 below:

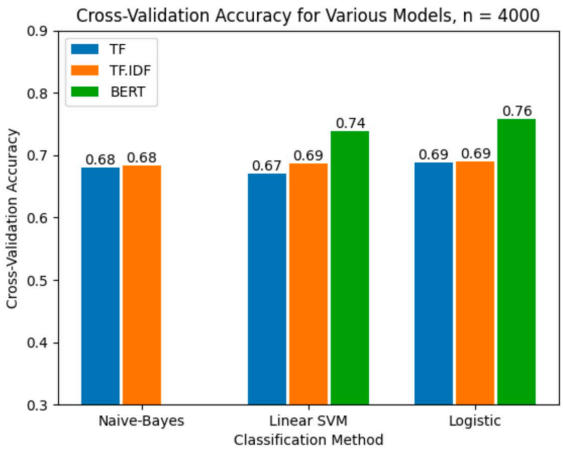

Figure 3: This graph shows the accuracy of feature extraction and classifier combinations over a subset of the training data

From the graph displayed above, it is evident that models using BERT embeddings clearly outperformed those using raw term frequency or tf-idf for feature extraction. The best performing model, yielding an accuracy of 0.76, utilized BERT embeddings in conjunction with the logistic model. However, BERT embeddings and a linear SVM classifier yielded an accuracy of 0.74 – only a difference of 0.02. The lowest performing model consisted of raw term frequency in combination with a linear SVM classifier, yielding 0.67 accuracy. Similar to BERT models, tf and tf-idf models' accuracy varied by at most 0.02 when paired with different classifiers. This suggests feature extraction methods contribute to a system's performance more than the classifier.

Figures 4 and 5 display the results from testing a tf-Naïve Bayes system and PySentimiento on 2,000 tweets from the US airline dataset, respectively. While not a perfectly designed test, we can still expect relatively similar results in other domain-specific applications as this

	Precision	Recall	F1-Score	Support
0 ☹	0.91	0.82	0.86	1260
2 😐	0.56	0.65	0.60	416
4 🙂	0.73	0.86	0.79	324
Accuracy			0.79	2000
Macro Avg.	0.73	0.78	0.75	2000
Weighted Avg.	0.81	0.79	0.79	2000

Figure 5: This table shows PySentimiento's performance on 2,000 tweets from the US ariline dataset

	Precision	Recall	F1-Score	Support
0 ☹	0.95	0.46	0.62	1260
4 🙂	0.30	0.90	0.45	324
Accuracy			0.55	1584
Macro Avg.	0.62	0.68	0.54	1584
Weighted Avg.	0.82	0.55	0.59	1584

Figure 4: This table shows Naive Bayes' performance on 2,000 tweets from the US airline dataset

simulates the retrieval process for getting new tweets about a specific topic which the models were not trained on. Looking at the macro-averaged F1-scores for each system, we can see that PySentimiento outperforms Naïve Bayes 0.75 to 0.54. Though a less relevant metric due to the class imbalances, PySentimiento also outperformed Naïve Bayes with respect to weighted average as well.

It is interesting to note that both systems performed best on negative tweets and performed significantly worse on non-negative tweets(i.e., positive or neutral). Naïve Bayes produced an F1-score of 0.62 for negative tweets and only 0.45 for positive tweets. Moreover, PySentimiento produced an F1-score of 0.86 for negative tweets, 0.60 for neutral tweets, and 0.79 for positive tweets.

6 Conclusions

6.1 Findings

The first and foremost conclusion that can be drawn from our project is that BERT feature extraction significantly outperformed other methods, while classifiers yielded similar results. Thus, we should expect to see companies and university courses adopt this cutting-edge technology if they have not done so already. In addition, our results suggest that our system had more difficulty correctly classifying positive and neutral tweets in comparison to negative tweets. One hypothesized reason for this could be that users may be more inclined to post about a negative experience with a brand or company. Consumers may treat a positive experience with a company or product as the status-quo and only take to Twitter to express their dissatisfaction when their expectations are not met. Furthermore, it's possible that users may use stronger language when expressing negativity.

6.2 Limitations

Given we are all undergraduate students working from personal computers or laptops, there were limiting factors to a project of this degree of ambition. Firstly, we would have liked to use our own BERT-Base implementation from end to end. We believe running a more fine-tuned version of BERT over the entirety of our dataset would lead to better results. However, this would have taken far too long on our hardware and given time constraints we had to opt for alternative solutions. Furthermore, we were only granted a standard access level to the Twitter API. This only allowed us to retrieve tweets from the last seven days and also capped the amount of tweets we could retrieve. Ideally, our front-end application could display more tweets over a longer period of time such as a year or quarter to better represent the changes in sentiment.

6.3 Future Work

Firstly, our Flask application currently runs locally on our machines. The apparent next step is to host on it on AWS or another cloud service to potentially provide companies with a valuable resource. Once available in production environments, we could receive user feedback and iteratively update our application.

Subsequently, we would like to evaluate the efficacy of our system after training on more domain-specific datasets. We used a broad-topic dataset, but individual companies would benefit from the ability to narrow down the domain space to more relevant tweets.

Furthermore, we think it would be valuable to companies if we could expand our system to include posts from other media platforms. While Twitter is a user-dense platform that provides valuable information, it is a small subset of the

totality of microblogging content available on the internet. Our system could greatly be improved if we could perform similar sentiment analysis on YouTube video transcripts, YouTube comments, Instagram posts, Instagram comments, etc.

Lastly, we think our system could be used to benefit more than just companies; we would like to expand our system to accommodate for public institutions such as universities or even individuals.

7 Contributions

First and foremost, all team members contributed greatly to the development and success of our system. As for the implementation of the system, Jasper and William spearheaded the training of our models. Alexander implemented the testing as well as the generation of graphs to display evaluation metrics and the performance of our models. Furthermore, Timothy and Evan worked in tandem to develop the Flask web application and front end. As for the poster and reports, we each took a section and helped with other parts of the assignments if needed. This group worked well together, and we are all satisfied with the efforts put forth from one another; not all group projects tend to go as smoothly.

References

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. B. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (EMNLP), pages 79–86. https://www.cs.cornell.edu/home/llee/papers/senti ment.pdf

Guoning Hu, et al. 2017. Analyzing users' sentiment towards popular consumer industries and brands on Twitter. *2017 IEEE International Conference on Data Mining Workshops* (ICDMW 2017). arXiv:1709.07434v1 **[cs.CL]**

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent Twitter Sentiment Classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 151–160, Portland, Oregon, USA. Association for Computational Linguistics.

Nur Azizah Vidya, Mohamad Ivan Fanany, and Indra Budi. 2015. Twitter Sentiment to Analyze Net Brand Reputation of Mobile Phone Providers. Procedia Computer Science, 72: 519-526 https://www.sciencedirect.com/science/article/pii/S1877050915036200?via%3Dihub

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter Sentiment Classification using Distant Supervision. Stanford University Department of Computer Science. https://cs.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf

Vishal Kharde and Sheetal Sonawane. 2016. Sentiment Analysis of Twitter Data: A Survey of Techniques. International Journal of Computer Applications. 139:5-15. https://arxiv.org/ftp/arxiv/papers/1601/1601.06971.pdf

Jay Alammar. 2019. A Visual Guide to Using BERT for the First Time. The Illustrated Transformer. http://jalammar.github.io/a-visual-guide-to-using-bert-for-the-first-time/

Ruben Winastwan. 2021. Text Classification with BERT in PyTorch. Towards Data Science. https://towardsdatascience.com/text-classification-with-bert-in-pytorch-887965e5820f

Juan Manuel Pérez, Juan Carlos Giudici, and Franco Luque. 2021. PySentimiento: A Python Toolkit for Sentiment Analysis and Social NLP Tasks. https://github.com/pysentimiento/pysentimiento

Twitter US Airline Sentiment. https://www.kaggle.com/datasets/crowdflower/twitter-airline-sentiment

Sentiment140. http://help.sentiment140.com/for-students