Henry Zhou
Comp150NLP Final Project

Cross Domain Sentiment Classification on Reddit Comments Using Distant Supervision

Abstract

The introduction of the internet gave rise to various platforms of open communication such as forums, social media, Internet Relay Chat (IRC), and review sites. Popular websites and social networks such as Twitter and Facebook serve over 1 billion unique visitors monthly, with Twitter receiving 500 million daily tweets per day [5]. In the project, I used Machine Learning techniques in order to create a classifier that can apply sentiment analysis to determine whether a tweet is positive or negative in sentiment. I also examine the cross domain usefulness of my classifier with comments posted on Reddit.com. The result is a surprisingly effective sentiment analyzer that manages to achieve similar accuracy levels of around 80% on a hand labelled test set.

Introduction

Reddit.com is a website where people can create personal accounts and form interest groups called “subReddits”. In these subReddit groups, users post content that pertains to the theme of the subReddit such as links to other webpages, images, music, and text. Users are also able to respond to posts via comments and to each other by sending private messages. One other integral mechanism of Reddit is the concept of “karma”. Users can choose to support other user posts or comments by giving up to 1 “upvote” per post. Likewise, users can also “downvote” other posts when they believe a post is inappropriate, poor in taste, or does does not contribute any value to a comment thread.

Oftentimes, Reddit comments reflect user sentiments about a certain topic. The diversity of the Reddit community offers a myriad of user posts related to many different areas. For my project, I attempted to build a sentiment analyzer that is able to accurately gauge the sentiment of a Reddit post as either positive or negative. This can be extremely useful and could have many applications in the areas of business, politics, international relations, finance, among others. The ability to extract people’s sentiment about different topics could possibly be used to predict future events, estimate interest in a product, or gauge the general public’s feedback to events and changes.

To extract sentiment from Reddit comments, I trained a Naïve Bayes Classifier on a sentiment labeled corpus of 1.6 million tweets. This Twitter corpus was produced by Go, Bhayani, and Huang [1], who used distant supervision to automatically create a weakly labeled training set.

I chose to use the corpus from Go et al. because I was not able to find a polarity labelled Reddit comment corpus to train with and the twitter corpus was significantly richer than many other corpora that I found online. Consequently, this project also tests the viability of cross domain sentiment classification between twitter and Reddit comments. The specifics of how the twitter data was selected can be found in the Methodology section.

Past work has been done with Reddit in the sentiment analysis space. Ting [4] uses Recursive and Recurrent Neural Networks in his work to gauge the sentiment or lack there of in the comments of different subReddits as well as seeking correlation between comment karma and sentiment. Outside of Reddit, sentiment analysis has also been applied areas such as movie reviews, as explored by Pang et al. [2].

Definition of Sentiment

To clarify, I define sentiment as a subjective and person positive or negative feeling. Examples of positive sentiments include feelings of happiness, joy, and excitement. Negative sentiments include sadness, sorrow, feelings of depression, hatred, and sympathy.

One challenge of sentiment extraction is the lack of context. As in part of speech tagging, ambiguity can also apply to sentiment. Take for example the comment “It got shut down”. There is no way to determine the context of the situation when only given the text. It is possible for the commenter to be feeling positive or negative. Tweets and comments can also be neutral, or lack sentiment, as well. Comments such as “the official stream on nfl.com http www.nfl.com draft 2015 live” or “then pikachu uses thundershock” are not particularly polar in sentiment. Consequently, test data is hand labelled as neutral when the sentiment of a comment or tweet is ambiguous or lacks polarity as the classifier is not trained to identify neutral tweets and comments.

Approach

I implemented Naïve Bayes with Laplace Smoothing to train my binary classifier. I also make use of CMU Ark-Tweet-Nlp [5], a part of speech tagger that is trained on a twitter corpus and has proven to provide good performance (90+% accuracy) on twitter tweets. By using a POS tagger, I also explore the viability of training classifiers based on certain POS tags such as adjectives, adverbs, and verbs and compare its performance to using the entire corpus.

Training Data Background

The Twitter corpus was completely gathered by Go et al. over between April 6th, 2009 and June 25, 2009. To acquire the data, they queried for tweets that contained positive and negative emoticons. Any tweet that contained positive emoticons such as “:)”, “:D”, and “:-)” were labelled as positive and tweets that contained negative emoticons such as “:(“ and “:-(“ were labelled as negative tweets. The emoticons were then stripped from the tweet itself because they hurt the performance of MaxEnt and SVM classifiers that Go et al. were also implementing. More information about their distant supervision method can be found here at [http://cs.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf] .

Feature Reduction

Unlike a corpora containing grammatically correct and coherent text such as the Brown Corpus, the twitter corpus is not guaranteed to be grammatically correct, be spelled correctly, or even be composed of words. Language wise, tweets are very different from traditional English and can contain intern slang words like "lol". Tweets are often casual and are not in pristine grammatically conditions. Users often misspell words both on purpose when they all map to the same word. Heavy preprocessing was done in order to convert and capture as many words as possible without changing the meaning of the sentence.

Links and URLs

Many tweets incorporate links and URLs to other websites. However, there is a great variety of websites and a tweet that suggest for a user to follow it is unlikely to be conveying significant sentiment. Consequently, links and URLs are converted to tokens. Substrings that started with "www." or "http" were considered as URLs. These strings were further subdivided into corresponding tokens. URL's that ended in ".gov" were given the token URL_GOV_TOKEN. ".edu" and ".com" tokens were treated similarly. These URL's did not necessarily have to start with "www." or "http", but they required the postfix to be assigned these tokens.

User Shoutouts

Often times, users tweet at other user accounts. These shoutouts are indicated by the "@" symbol followed by an account name. Shoutouts were converted to USER_TOKEN because a shoutout usually does not convey any sentiment intrinsically.

Punctuation

Many punctutation marks are removed after URLs have been properly tokenized. This allows for the tokenization of words as best as possible.

Extraneous Consecutive letters in words

Often times users misspell words in different ways that all convey the same meaning. For example, a user may spell the word "love" as "looooooooovvveeeeeee". Misspellings are commonly made by duplicating characters in a word. As utilized in Go et al., I reduce any consecutive series of 3 or more of the same character to 2 characters. This should help reduce different misspellings of the same word and strengthen the feature of the word the misspellings derived from originally.

Machine Learning Methods

Naïve Bayes

I built a Naïve Bayes Classifier to classify tweets and Reddit comments as either positive or negative in sentiment. The equation to describe Naïve Bayes is as follows:

$$C_{Best} = argmax_{cj} P(C_j) \prod_{i=1}^{length} P(x_i|C_j)$$

Where C_{Best} is the most probable class given a document containing features x, C_j is a class that is a member of the set C, and x_i is some feature in a document. Naïve Bayes makes use of several assumptions:

- Bag of Word Assumption
  Naïve Bayes assumes that the multiplication of probabilities is associative. It is assumed that the order in which the features are placed does not matter.
- Independence Assumption
  Naïve Bayes assumes that the generation of different features given a certain class is independent. Consequently, it is assumed that in a document D the P(A | C) is independent of the P(B |C) for some features A and B and some class C.

In order to handle words that do not occur for a certain class, I apply plus one or Laplace smoothing to distribute probability mass.

Bigram Hidden Markov Model

Another method that I used for sentiment extraction was calculating the Bigram Part-of-speech tag and emission probabilities for a tagged test set. To do this, I utilized the BigramHMM class I wrote in the third problem set and trained it on a corpus of 100,000 POS labeled tweets. The idea was to see if certain part of speech tag transitions and tag emissions are more likely in positive or negative comments or tweets.

$$C_{Best} = argmax_{cj} \prod_{i=1}^{length} P(t_{i-1}|t_{i-1})_{Cj} P(x_i|t_i)_{Cj}$$

The above equation explains the Bigram HMM model I use. The best class C_{Best} is the most likely product of Bigram tag transitions $P(t_{i-1}|t_{i-1})_{Cj}$ and feature emissions $P(x_i|t_i)_{Cj}$ for all Hidden Markov Models of a class C_j. I trained two Bigram Hidden Markov Models, one with 50,000 POS tagged positive tweets and the other with negative tweets. Whichever model produces a higher probability for the sentence decides the polarity of a tweet or Reddit comment. As Naïve Bayes is an established method of sentiment analysis, I wanted to explore the viability of applying the Bigram Hidden Markov Model as a different avenue for sentiment analysis.

Evaluation/Setup

Baseline

For my baseline, I test my classifiers against a hand labelled test set of 177 negative and 182 positive tweets. I use the exact same test set that is used in Go et al. [1], in which they were able to reach accuracies of around 80 to 83% with their Naïve Bayes Classifier.

Hand Labelled Reddit Corpus

I built a test corpus by querying a Reddit database for comments that resemble tweets in length. To query for positive comments, I use a distanced supervision strategy of querying for comments in between 30 and 140 characters in length that contain positive emoticons. To query for negative comments, I do the opposite and search for negative comments. A table of what counts as a positive and negative comment can be found in table 1.

These emoticons were later stripped off similar to how the training corpus was built. They were also preprocessed to remove extraneous punctuation as well as subject to the same preprocessing standards applied to tweets, mostly for URL tokenization. I then hand labelled the comments as either positive, negative, or neutral. What remained was a test set of 121 positive and 139 negative comments.

I decided to query in this manner because I believed that it makes sense to first compare tweets to similar Reddit comments before moving onto more complex analysis, such as comments with longer length.

POS impact

In order to implement a method using POS tags, I used Carnegie Mellon's ARK Twokenizer [7], a Part of Speech tagger trained on twitter data, specifically for twitter data. Unlike the NLTK POS tagger, Twokenizer uses a different set of part of speech tags, some of which are specific to tweets. Aside from labelling normal words with more traditional tags such as verb, adjective, and adverb, Twokenizer also recognizes internet slang such as tagging "lolololol" as an interjection and "fb" (shortened for Facebook) as a proper noun. Owuputi et al. demonstrated that Twokenizer performed very well and exhibited around 90 percent accuracy for certain test sets [6]. Consequently, I believed that it was a favorable POS tagger to use with the Twitter corpus I was using to train.

I used Twokenizer to create a separate, POS tagged corpus of 100,000 tweets, half positive and half negative. Using this tagged set, I created another training set by filtering out certain part of speech tags. The tags I chose to filter for and the reduction impacts can be found in table [SOURCE]. I then trained a Laplace Smoothed Naïve Bayes Classifier with this reduced training set. To fairly gauge the relative performance of this classifier, I trained another Laplace Smoothed Naïve Bayes Classifier with the 100,000 tweets, but I did not reduce the set.

I also used this 100k tagged corpus to train my Bigram Hidden Markov Model Classifier. Twokenizer was also used to tag my original Twitter and Reddit test sets. This allowed me to test my Bigram Hidden Markov Model method for sentiment analysis.

Positive	Negative
☺	☹
:D	>☹
=)	=(
☺	☹
=D	

Table 1. Emoticons used to query for Polar reddit comments

Results and Discussion

#	Method	Test Accuracy	
		Twitter Test Corpus	Reddit Corpus
1	Naïve Bayes, Laplace Smoothing	79.39%	82.69%
2	Naïve Bayes, Laplace Smoothing, UNK 3	79.39%	83.08%
3	Naïve Bayes, POS filtering, 100K Tweets	72.42%	77.31%
4	Naïve Bayes, Laplace Smoothing, 100k Tweets Only	73.82%	77.69%
5	POS Bigram HMM	72.07%	81.47%

Table 2. Test Accuracy Results for all classifiers on both test corpuses

Naïve Bayes

Overall, Naïve Bayes performed better than the Bigram model. Relative to Go et al., my results were slightly lower than the accuracies they were able to achieve. (around 80% – 83%). Wilson, Wiebe, and Hoffmann from University of Pittsburgh [3] observed that the two annotators that labelled documents with positive or negative sentiment disagreed around 18% of the time. Consequently, a test accuracy near 80% indicates very good performance as that is the level of certainty that people can reach between one another when it comes to classifying sentiment in the real world.

Filtering by Frequency

Method #2 filtered out words with frequency less than 3 in the training set as “<UNK>” whereas a method #1 did not. Methods 1 and 2 are otherwise the same classifier. However, this did not have significant impact on the Twitter or Reddit Test corpus. There was no significant difference between not changing words to unknown tokens and doing so. This was evident in the accuracies for both test results for Twitter and Reddit.

POS Filtering

Filtering by part of speech proved marginally significant. In method #3, the only features that were extracted were those with part of speech tags: `A,R,G,V,!,T` where:

- A – Adjective
- R - Adverb
- G – Other abbreviations
- V - verb
- ! - interjection
- T – verb participle

I decided to filter by these POS tags because they seemed like they could be more informative than features of other tags. The magnitude of reduction that the filtration caused can be seen in table 3.

POS filter feature reduction	
	Total Features
Before	1,528,346
After filter	452,557

Table 3. Feature reduction due to POS filtration

Overall, filtering performed slightly worse than regular classification on the original 100,000 tweet training corpus. As expected, both classifier performed more poorly than the classifier trained on 1.6 millions tweets as both had much less information to train on. This lack of significance in POS filtration was also observed by Go et al. [1], who also did not observe significant differences from POS filtration.

POS Bigram HMM

The Bigram HMM performed better than expected. I was surprised that the model was able to achieve almost as high accuracy as the other classifiers that were trained on 100,000 tweets. One significant result was the relatively high accuracy observed when testing on the Reddit Corpus.

Twitter vs Reddit

Interestingly, every classifier performed several points higher on their respective Reddit test set than on the baseline Twitter test set. This seems strange because every classifier was originally trained on Twitter Data. Consequently, it may be worth reexamining the Twitter and Reddit Data for possible sources of bias. Regardless, the accuracy values are still relatively close, indicating that cross domain classification between Twitter and Reddit may be a viable way to classify Reddit data.

Future work

Neutrality

The classifier I built works relatively well for twitter and short Reddit comments that are already polarized. However, this limits the functionality of the classifier because it can only handle such data. Being able to detect neutral comments and tweets would significantly boost the usefulness of this sentiment analyzer. This would allow the sentiment analyzer be more robust and flexible in its input and predictions. This could be implemented with a relabeled corpus with neutral labels. Such a corpus could be retrieved by using Amazon Mechanical Turk to produce a corpus. Other paths include finding the mathematical threshold that must be met between two probabilities for one document for the document to be considered polar.

Different degrees of Positivity and Negativity

Aside from being able to classify neutrality, it would be extremely useful to be able to classify positive and negative sentiments into more specific subcategories. For example, being able to classify positive Reddit comments or tweets as somewhat or strongly positive would provide further nuance and depth to the sentiments extracted. Even more ambitious would be the ability to specify the specific positive or negative sentiment a document falls under (aka happy, joyful for positive. Sad, angry, frustrated for negative).

SubReddit Sentiment and Real World Events

Now that we have shown some Reddit data can be classified relatively accurately with Twitter data, perhaps statistical models can be made by identifying patterns between Reddit comment sentiment and real world events. For example, perhaps aggregate sentiment analysis of a certain company can be used to predict the behavior of the corresponding security on the stock market. Another idea would be to use sentiment analysis to gauge Reddit's opinion about a certain presidential candidate. T the possibilities are endless if a reliable sentiment analyzer can be built to accurately extract sentiment from Reddit.

Conclusion

I successfully explored and implemented several different ways to build a sentiment analyzer using Naïve Bayes and Bigram Hidden Markov Models. I was able to attain accuracies near my baseline target and accurately classify Reddit comments using distanced supervision. In doing so, I was able to provide possible evidence that classifiers trained on Twitter data can successfully perform cross domain classification with specific, polar Reddit data.

Given that sentiment itself is subjective between people and sometimes dependent on context, it is extremely challenging to determine document sentiment and polarity. However, the ability to statistically gauge and predict human emotion may provide solutions to many moon shot problems, and advancements in Natural Language Processing may provide the key to unlocking this power.

References

[1] Go, Alec, Richa Bhayani, and Lei Huang. "Twitter sentiment classification using distant supervision." CS224N Project Report, Stanford 1 (2009): 12.

[2] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 79–86, 2002.
CMU Ark tweet NLP
[3] Wilson, Theresa, Janyce Wiebe, and Paul Hoffmann. "Recognizing contextual polarity in phrase-level sentiment analysis." Proceedings of the conference on human language technology and empirical methods in natural language processing. Association for Computational Linguistics, 2005.
[4] Ting, Jason. "A Look Into the World of Reddit with Neural Networks."
[5] Internet Live Stats. Accessed December 2015. http://www.internetlivestats.com/twitter-statistics/.
[6] Owoputi, Olutobi, et al. "Improved part-of-speech tagging for online conversational text with word clusters." Association for Computational Linguistics, 2013.
[7] ARK Twokenizer. Accessed December 2015. http://www.cs.cmu.edu/~ark/TweetNLP/

Special thanks to Madison May from Indico for advice regarding finding corpora, preprocessing advice, and general tips about sentiment analysis. Also special thanks to Nate Winters for tips and help with the many headaches associated with data processing, implementation, and other difficulties associated with sentiment analysis.