Negation-based Sentiment Analysis in Review

Lam Dang
ltdang16@earlham.edu
Department of Computer Science
Earlham College
Richmond, IN

.

1 ABSTRACTION

Many of communications, in an information era, are through texts (Message, Email, Review, etc...) and most of the time, they often convey a certain emotion. While communication through word of mouth makes it easy for people to understand others' emotions on a subject (as these emotions are usually shown in their tones, facial expressions, etc...), communication through texts is harder in emotion identification. Therefore, we can simplify them by having the machine doing that for us. In order to do so, we have to make sure that the accuracy of identifying emotions by the machine is high. And handling negation is one of the problems that are preventing the machine to do so.

Sentiment Analysis is the identification and determination of the emotion within the context of the texts. While researchers have been successful at making the machine identify the emotion based on positive/negative keywords (such as like, hate, love, detest, etcâĂę), they are researching on an ongoing problem of how negation word (i.e not, don't, won't, didn't, hardly, etc...) will affect the emotion of the text. A negation word, which consisted of many phrases, can either amplify or negate the emotion that was in the text. Therefore, knowing how to handle them correctly will increase the accuracy of sentiment analysis.

My work is on handling the negation to improve Sentiment Analysis in the context of Reviews. Most, if not all, reviews usually convey an emotion toward a subject. My plan is to identify the emotion and the negation word in the reviews, establish a relationship between the two words, and based on the relationship, output an indication whether the review is positive or negative.

2 INTRODUCTION

Sentiment Analysis has been one of many popular topics of Natural Language Processing and Text Analysis for the past decade. It involves extracting, quantifying, and studying subjective information to give insight into a person's perspectives through word forms. Therefore, Sentiment Analysis is usually applied in "Voice of Customers such as reviews, responses from social media. Alexander Pak had illustrated how Twitter is a source for mining data for Sentimental Analysis, as most posts on Twitter about a subject usually reflects users' opinions and feelings toward that subject [10]. Twitter's subjects are usually about politics, entertainment, news, product reviews so the application of Sentiment Analysis consists of a wide range, from customer services and marketing to clinical medicines.

How Sentiment Analysis works: When you feed a Machine Learning model sentences, the model would identify words that convey positive and negative tones. After much calculation and evaluation, the model should be able to categorize the sentences as conveying positive or negative tones.

However, the basic methods described above have not taken into account the existence of *Negation* when it comes to Sentimental Analysis. *Negation words* are very common in communication (i.e not, don't, won't, etc..). They can serve to contradict the meaning/purpose/emotions of the sentence or they can serve to amplify these emotions. For instance, the word "Good" conveys a positive tone. The word "Not good" conveys a bad tone. On the other hand, "Not only is it good" will amplify the positive emotions. Failure to identify the purpose of negation word in the context will, in turn, reduce the accuracy of the Machine Learning models tremendously, especially when negation happens in multiple places.

In order to improve the accuracy of the Sentimental Analysis model, we handle negation properly so that the phrase that goes with negation would convey its proper positive/negative tone. In this proposal, I will introduce Negation-Sentiment Scoring System, which will establish the relationship between Negation Word and Sentiment Word in a sentence and convey them into a vector which includes these features [Number of Negation word, Number of Sentiment word, Number of Positive Word, Number of Negative words, and the Accumulated Score]. For reviews with multiple sentences, I apply the Negation-Sentiment Scoring algorithm to each sentence and merge the vectors created into one vector. IâĂŹm going to train a Logistic Regression and Neural Network model to learn the relationship vector created and the expected output to make predictions in the future. The efficiency of my model will be determined by how accurate the model is, which is conveyed by how many correct predictions it makes.

3 RELATED WORK

The problem of negation is one of the major issues that researchers in Natural Language Processing have been tackling for years. According to a survey in 2010[14], many creative solutions have been created, each with their own advantage and disadvantages. One common theme of these solutions was to identify the polarity words

Senior Capstone, May 2020, Earlham College

(words that indicate positive/negative) and incorporate them with negation. In this section, we will review some solutions.

One of the solutions proposed was to turn the words following a negation into features that reflects the opposite sentiments (positive to negative and vice-versa) of themselves[11]. An example of this is 'I do not like cake'. The proposed solution would transform the sentence to 'I do not NOT_like NOT_cake'.How Pang et al envision when he created the solution was that it would be applied to every sentence and create multiple features that reflect positive/negative semantics of the text data [11]. This solution could catch all the negated semantics in a sentence. However, it will also turn the non-polarity expressions (a word with no sentiments) into negated expression, like the example above ('NOT_cake' does not have any meanings and reflect any semantics). This will not only be literacy inaccurate but also create a sparse amount of features that the model has to handle.

Polanyi and Zaenen raised the solution of 'Contextual Valence Shifting', which assigns scores to words that contain polar expressions(positive negative)[12]. For every word that contains a positive tone, it will be assigned to a positive score. On the other hand, every negative word will be assigned with a negative score. For example, the word 'like' will be assigned a score of '+2', according to Polanyi and Zaenen's solution. However, since the negation 'not' precedes the word, the negation will reverse the score to '-2'. This method will transform the text data to a numerical-based score system, which will in turn, helps the machine learning models find the patterns for classification more accurately and efficiently. The bigger visions of this method were not only identifying the existence of polar expressions[1] but also their implicated 'intensity'. For example, if 'good' is the equivalence of a '+2' score, then 'excellent' should be the equivalence of a '+4' score. A slight disadvantage of this method in terms of handling negation is that the score is only reversed if the polar expression **immediately** follows a negation.

A third solution, proposed by Choi and Cardie, was to apply a set of previously-defined inference rules to phrase[4]. For example, if we apply the rule Polarity(Negative Polar (-) + Preposition + Negative Polar (-)) = Positive Polar (+) to a phrase 'Lack (-) of Crime (-)' then the model will evaluate it as a positive expression. This method requires a large predefined set of semantic rules and the phrases should be syntactically accurate. With designing, implementing, and testing this solution, Choi and Cardie's test results indicated their models outperformed the models that use scopes of negations that consider no-rule-based expressions.

These papers are just some of the solutions that are proposed in.

4 DESIGN

The framework of the project includes four major elements: Data Collection [RED], Data Preprocessing [BLUE], Model Training [PURPLE], Model Prediction and Evaluations [GREEN].

[1]Polar Expression: words that indicate emotions

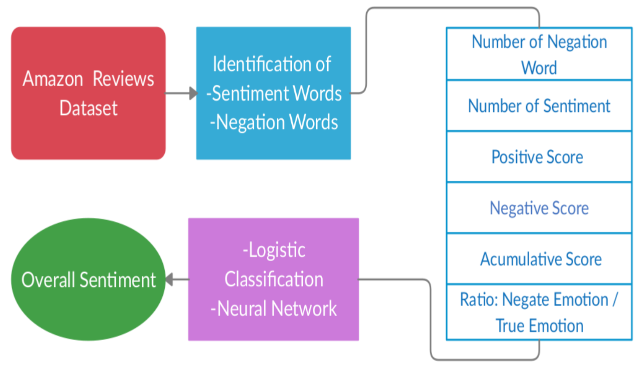

Figure 1: Framework

First Stage - Data Collection: Data Collection includes gathering data that includes two important features: Text Review and Grade Review (Star ladder or simple Positive/Negative). There are many sources that include Sentimental Analysis datasets, which include reviews from reliable sources such as Amazon, Airlines, etc. The dataset used in the project is the Amazon Reviews dataset, which will contain reviews from 2000 to 2015, with millions of reviews, and is one of the most helpful and reliable online review sources [9]. It also has the star rating, which serves labels for the dataset. The dataset satisfied all of the requirements needed for a Sentiment Analysis model: It contains many data points (more than 100 million) in order to fix common problems in training ML models such as (over-fitting, lacking data, etc...). It also contains the features needed (text review, star reviews) for Sentiment Analysis training. Lastly, it comes from a reliable source (Amazon official dataset), carefully preprocessed by Amazon engineers for researchers to study and test on.

In addition to the reviews data from Amazon, the corpus of negation word, positive word, and negative are also in need. Since the algorithm depends heavily on correctly identify Negation words and Sentiment words, these corpus needs to be complete and accurately represent their intentions. The Sentiment Corpus used for the project is provided by Minqing Hu and Bing Liu[7], which includes 4800 negative sentiment words and 2000 sentiment positive words. While it is arguable that the sentiment words in English are more than 7000 words, the corpus provided are the total 7000 words that are widely and most commonly used sentiment words in online media, therefore covers most, if not all, of the reviews in my dataset. The Negation Corpus is mainly from the Vader Corpus by C.J. Hutto[8]. The Corpus has been standardized and added as one of the major Negation corpora in Natural Language Processing Toolkit

Second Stage - Preprocessing: Preprocessing Data is the main algorithm that was developed in this project. For every review that is to be preprocessed, the algorithm should generate a compact vector that displays all the information of the review, rather than the review itself. The vectors should have the information on negations, sentiments in the review, and their relationship to each other. From that information, we would be able to generate an accumulative score that will ultimately predict the sentiment of the review as either "Bad", "Good" or "Neutral".

The overview of the algorithm is as following: For every review, the algorithm checks and store every negation and sentiment words that appeared in the review. If the review does not have any sentiment words, the algorithm will disregard the reviews. If it is vice-versa, it will identify the relationships between the negation words and the sentiment words that follow it. If the sentiment words do not have a relationship or have any negation words that precede it, we will increment either positive or negative scores based on the positive/negative bias of the sentiment word. If a relationship exists between negation words and the sentiment words, we will identify and calculate how strong the relationship is and convert them into scores, using the formula as follow:

$$Negation\_Score = \frac{1}{Distance(Negation\ word, Sentiment\ word}$$

The assumption of the formula is that the further the two words are from each other, the less strong the relationship and the more unlikely that the negation word will reverse the sentiment of the sentiment words.

Algorithm 1: Preprocess Review

```
Result: Five dimension array with features: [negation count,
        sentiment count, positive score, negative score,
        accumulative score]
initialization;
while For each sentence in a review do
    Get a list of negation and sentiment word;
    if There's no sentiment word then
        pass;
    else
        Add number of negation word, sentiment word to the
          appropriate feature;
        while For every negation word that precede a sentiment
          word do
            Calculate Negation Score based on the distance of
              the pair;
            Add it to accumulative score;
        end
        while For the sentiment words that were not paired with
          a negation word do
            if the sentiment word is positive then
                +1 for positive score;
            else
                +1 for negative score;
            end
        end
        Add Positive Score and Negative Score to the
          Accumulative Score;
    end
end
```

In the end, the algorithm will generate the following 5 features:

- The number of negation word
- The number of sentiment word
- Positive/Negative Scores, generated by sentiment words that holds their true meaning
- The Accumulative Scores = Negations Scores + Positive Scores - Negative Scores

These features will become the variables that would predict the sentiment of the whole reviews.

Third Stage - Model Training: At this stage, the data set that has been processed will be applied to two types of model: the Logistic Regression model and Neural Network model.

Logistic Regression is one of the most popular probabilistic classification models in Machine Learning.[3]

$$Pr(Y_i = 1|X_i) = \frac{exp(\beta_0 + \beta_1 X_i + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \beta_5 X_5)}{1 + exp(\beta_0 + \beta_1 X_i + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \beta_5 X_5)}$$

The Logistic Regression operates as followed: X_1 *to* X_5 depicts the 5 features that is provided from the preprocess. $Pr(Y_i = 1|X_i)$ depicts the probability (from 0 to 1) of what labels is most fitting given the five features. By constantly training the model with the 5 features and their correct label, the model will automatically adjust the β variables so that when the model is given a set of features to predict the label, the model will calculate the probability based on the features and the trained (and now fixed) labels for the three presented labels "Good", "Neutral" and "Bad". The label with the highest probability be the predicted label of the model.[13]

Neural Network is also regarded as one of the most popular and effective models in classifications. The Neural Networks operates similar to Logistic Regression. Each of the five features will be contained in five nodes. Those nodes will them connects with the nodes from the hidden layers, each of those nodes contain a weight that will determine the importance of each features will be. By training the models, we actively changes the weight of the hidden nodes to convey which features factors more into the predictions and which are less.[6] In the end, when the model is used to predict an label, it will receive 5 features, multiply each with the weight of the nodes, and output a prediction label. [15] [5]

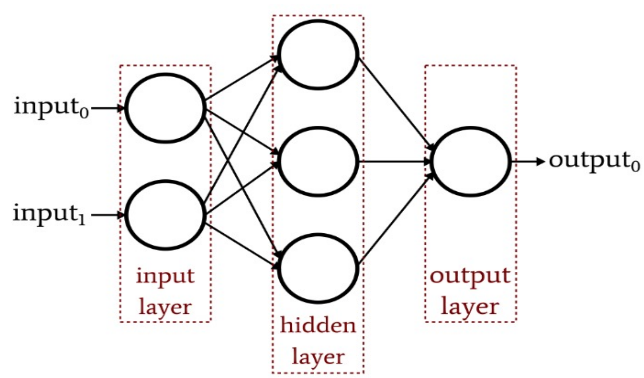

Figure 2: Example of a Neural Network[1]

Fourth Stage - Model Prediction and Evaluation: In stage 4, we will divide the test set of the data into 2 separate parts: The features that will be used to generate predictions, and the labels that will be used to compare with the predictions to observe the accuracy. After being trained, we would feed the ML models with unlabeled preprocessed features. and let the models from stage 3 predict the data point as conveying positive or negative. We would then compare with the true label of the dataset and extract the accuracy of the model.

5 RESULT

With 105,000 data points being trained, and 45,000 data points being used to test, the algorithm achieved the following results:

Logistic Regression: The model trained and rendered achieved 52% accuracy, with most of it accuracy falls in the class of "Good" and "Bad". The preprocessing algorithm assigns the score of positive and negative to the review that the algorithm was assessing. Therefore, it is highly possible that those result in positive score will be categorized into the "Good" class and those result in a negative score will be categorized into the "Bad" Class. With most of the reviews is one-sided, using emotion keyword that points to either positive or negative emotions, the algorithm has achieved true positive/ true negative more than 80% of the time. The "Neutral" Class, however, is hard to predict accurately, with mix emotion sentences and keywords that leads to False Positive/False Negative rather than True Neutral.

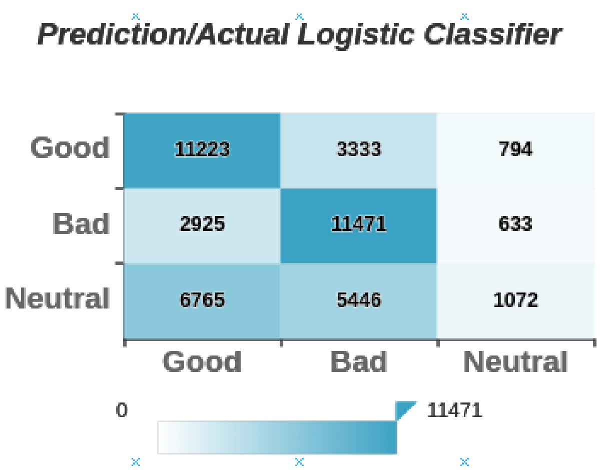

Figure 3: Logistic Regression Confusion Matrix

Neural Network: The Neural Network achieved a close proximity accuracy, 54.43% accurate predictions. The predictions of each labels also has the same result as the Logistic Regression models, as "Good" and "Bad" classes have good predictions and "Neutral" class has a low-accuracy predictions. Another contribution to the 54.43% accuracy score, is that the loss value of the Neural Network is higher than expected. The loss value of Neural Network determines how well a Neural Network is optimized with the dataset to it efficiency.[2] The lower the score, the more optimized the network. With an optimized Neural Network's loss value usually smaller than 0.5, the Neural Network that is developed and trained above can be an under-optimized Neural Network.

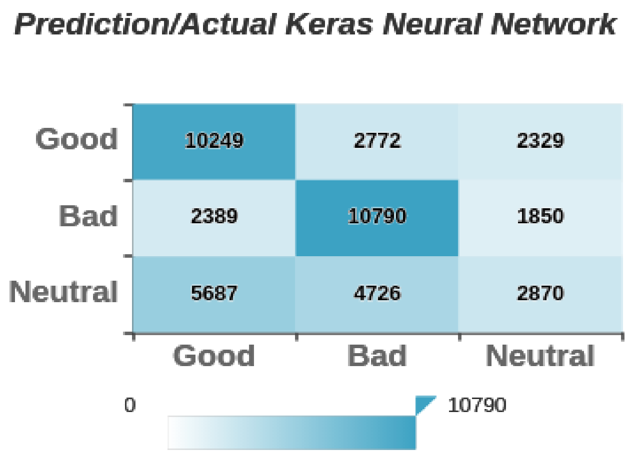

Figure 4: Neural Network Confusion Matrix

The major flaw of the models is that it fails to categorized most data point with "Neutral" class. The "Neutral" class, depicts the indifference feeling of the customers toward the product. The customer would use a mixture of positive and negative to describe what they like and don't like about a product that lead to a "Neutral" feeling. It is hard to categorize this class because the class is on the border of "Good" and "Bad". The ideal scenario, and the scenario that my algorithm is aiming for, is that the amount of negative and positive indications are balance, which will lead the category "Neutral". However, as the result shows, only 20% of the case are so, and categorized as a "Neutral" class. In many cases, even though the positive and negative indications were not balanced, the customers would still give it a "Neutral" emotion class.

6 FUTURE WORK

The design of the algorithm is based on many assumptions about sentiments in text. Therefore, the assumption that Neutral class resulted from a balance of Positive and Negative indications, have led the models in the project produce poor predictions on the Neutral class. For the model to be more accurate and predict the Neutral Class better, further study the pattern of "Neutral" class and add it as a feature such that when the pattern appeared, the data point will be categorized correctly, thus improving the accuracy of the model.

Efficiency of the preprocessing algorithm also raise a big problem. Having to check whether each word in the review belongs to one of the categories (Negation, Positive, Negative) is extremely time-consuming. However, the algorithm is possible for parallelism, which decrease the process of preprocessing significantly, allowing the model to handle bigger datasets.

7 CONCLUSION

The growing use of technology and the internet brings with it a growing needs of automation, especially towards texts. In the growing e-commerce business, automating reviews from text to score will not only aid the purchasing decision of a customer, but also aid the evaluation on quality and quantity of the products by companies. This paper introduced a new process of handling and preprocess data, especially negation within the context of sentiment analysis, that would generate a good model for Sentiment Analysis. Were the requirements only identify "Good" versus "Bad" labels, the preprocessing algorithm above will generate a predictive models that can achieve up to 80% accuracy. However, were it to predict more labels, there are still rooms for improvements. By combining, expanding the algorithm shown in this paper, researchers will be able to create a multi-class prediction model that could apply not only in e-commerce, but also other areas as well.

REFERENCES

[1] [n. d.]. How to Train a Multilayer Perceptron Neural Network - Technical Articles. https://www.allaboutcircuits.com/technical-articles/how-to-train-a-multilayer-perceptron-neural-network/

[2] Algorithmia. 2020. Introduction to Loss Functions. https://algorithmia.com/blog/introduction-to-loss-functions

[3] Weiwei Cheng and Eyke Hüllermeier. 2009. Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning* 76, 2-3 (2009), 211–225.

[4] Yejin Choi and Claire Cardie. 2008. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 793–801.

[5] Martin T. Hagan, Howard B. Demuth, and Mark Beale. 1995. Neural Network Design.

[6] Larry Hardesty and MIT News Office. 2017. Explained: Neural networks. http://news.mit.edu/2017/explained-neural-networks-deep-learning-0414

[7] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 168–177.

[8] Clayton J Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth international AAAI conference on weblogs and social media*.

[9] Susan M Mudambi and David Schuff. 2010. Research note: What makes a helpful online review? A study of customer reviews on Amazon. com. *MIS quarterly* (2010), 185–200.

[10] Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining.. In *LREc*, Vol. 10. 1320–1326.

[11] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*. Association for Computational Linguistics, 79–86.

[12] Livia Polanyi and Annie Zaenen. 2006. Contextual valence shifters. In *Computing attitude and affect in text: Theory and applications*. Springer, 1–10.

[13] Zew Steven and Rohit C. 2019. Logistic Regression for Multi-class Classification with Example in Python. https://acadgild.com/blog/logistic-regression-multiclass-classification

[14] Michael Wiegand, Alexandra Balahur, Benjamin Roth, Dietrich Klakow, and Andrés Montoyo. 2010. A survey on the role of negation in sentiment analysis. In *Proceedings of the workshop on negation and speculation in natural language processing*. 60–68.

[15] Tony Yiu. 2019. Understanding Neural Networks. https://towardsdatascience.com/understanding-neural-networks-19020b758230