# Twitter Sentiment Analysis with LSTM- Importance of the Neutral Classification

Yuncheng Gao
yxg151230@utdallas.edu

Melissa Fulfer
mdf150230@utdallas.edu

Usuma Thet
uxt130030@utdallas.edu

Rishi Dandu
rkd160030@utdallas.edu

## ABSTRACT

In this project, we have implemented a long short-term memory (LSTM) neural network to perform sentiment analysis on a dataset of tweets. The LSTM network is a relatively new implementation of a neural network. It offers many interesting improvements over other types of networks. Many major technology companies such as Google, Apple, and Microsoft utilize LSTM networks in fundamental components of their products. To expand on work that has already been done in this particular field of research, we explore the importance of the neutral category in the typical limited positive and negative classification scheme.

## KEYWORDS

LSTM, Neural Network, Sentiment Analysis, Twitter

## 1 INTRODUCTION AND BACKGROUND

Twitter is a gold mine of data. As one of the largest social networks in the world for news and information, advertising 330 million active users as of Q1 2019 and 500 million tweets daily [2], various machine learning studies may be applied on this data. This naturally includes sentiment analysis, or opinion mining, which is used primarily to discern public opinion on topics such as business products and launches, political views, social phenomena, and many more. Corporate businesses and political campaigns are just a few major customers that would greatly benefit from sentiment analysis to plan their strategic actions for maximized profits.

Sentiment analysis is not a new topic by any means. Rather, this specific field of study has dated roots since the beginning of the 20th century, only finding its mass rise in popularity through computer-based sentiment analysis with the availability of subjective texts through the Web. Statistically, 99% of related research papers have been published after 2004 [7]. In particular, a popular research topic is discovering an efficient automated process to classify positive and negative opinions from Twitter's tweets. Though, this topic raises an interesting point concerning the absence of the neutral sentiment, which more often than not gets ignored. In sentiment analysis, especially when statistical techniques are involved, researchers tend to filter out neutral portions of the inputted text. Neutral texts tend to be less informative and it is more productive to focus on the binary classification for the outputted sentiment. Another reason for the filter is that due to neutral texts lying near the boundary of the binary classification, assumptions are made that there is not much training profit from them compared to clear positive or negative texts [10].

There have been past studies conducted to highlight the importance of the neutral category. One example would be the work done by Koppel and Schler [5], focused on SVM and geometric properties to improve overall accuracy of the sentiment analysis. In addition, even simple intuitive reasoning may be used as an explanation. Take any factual statement or observation: "The weather is hot." How could this possibly by classified as either positive or negative? As we shall see, this paper will showcase the importance of the neutral category with another algorithm that is widely-used.

## 2 RELATED WORK

### 2.1 The Importance of Neutral Examples for Learning Sentiment

An interesting study on the neutral category in classification systems was conducted by Koppel and Schler [5], in which they focused on geometric properties and support vector machines to improve overall accuracy. There are two reasons that researchers tend to ignore the neutral category. The first is that they generally lie near the decision boundary of a binary model and are therefore classified by the model as well as they possibly can be. The second is that they do not offer much for the model to learn anyways, so they do not need to be analyzed. These researchers argue that there are actually many interesting ways that neutral texts can be used to improve learning models.

This paper finds that because negative sentiment in their model was primarily determined by a lack of positive words, many of the neutral texts fell into the negative classification category. Therefore it is not safe to assume that they lie equally distributed along the decision boundary. The researchers tried incorporating the neutral texts in a few different ways. First they used multi-class SVMs, which treated the three classes as unordered. This offered a slight improvement over the results when neutral texts were left out. Offering a similarly small improvement was the use of linear regression. In the linear regression implementation, the neutral sentiment was treated as an intermediate between the negative and positive classes [5].

Neither of these approaches properly leveraged the aspects of the neutrals that were not intermediate, but yet were distinct from the positive and negative categories. The SVM model learned three models (positive-negative, positive-neutral, negative-neutral) and then combined these models. To add on to this approach, the researchers suggest taking the relationship between these pairs into account as well. They employed an optimal stack that maps each possible outcome the majority class of the examples that have that outcome. If both positive and negative are higher than the neutral prediction, then the text is considered neutral. In context, this makes sense because the text probably expresses some type

of mixed sentiment about the topic. This three class stack implementation consistently outperformed the positive-negative models, confirming the importance of the neutral category in sentiment analysis [5].

### 2.2 LSTM based Sentiment Analysis for Patient Experience Narratives in E-survey Tools

Another study conducted by researchers in China compared the effectiveness of five machine learning methods on sentiment analysis of patient experience narratives. They compared support vector machine, random forest, gradient boost decision tree, XGBoost, and LSTM. They found that LSTM was the most effective method to use for this purpose. It greatly outperformed SVM, and got slightly better results than the other three models. The researchers concluded that this was due to two reasons. First, because the input of LSTM is a sequence of vectors that are generated by word-embedding, the vectors of synonyms are similar to each other instead of being treated as totally different features. The other reason is that the LSTM model is designed to recognize patterns in sequences of data over time, while the other models are not [1].

## 3 THEORETICAL/CONCEPTUAL STUDY OF LSTM

The algorithm that we are implementing is a long short-term memory (LSTM) neural network. The LSTM neural network is an artificial recurrent neural network (RNN). To fully understand this algorithm, let us first define a few terms.

A neural network is a machine learning algorithm that is modelled after the human brain, through a network of perceptrons joined together in layers. The network takes data as input and produces predictions as output. The algorithm is able to then learn by running the algorithm multiple times on training data, and its accuracy can be measured by running on test data. A recurrent neural network has an internal memory, utilized so that the output of each input depends on the computation of the previous input. After an output is produced, it is copied and fed back into the recurrent network. This allows each decision to be made based not only on the current input but also on the output that the network has already learned in previous iterations. In a plain neural network, all of the inputs are independent of each other, whereas a recurrent neural network provides the advantage that it allows inputs to be related to each other. However, there are a few disadvantages. For example, recurrent neural networks have the vanishing gradient problem, which means that as you get further through the network, the gradient decreases and it becomes very hard to update the weights. This makes training a recurrent neural network quite difficult and time-consuming [9].

LSTM neural networks are an improved version of recurrent neural networks that makes it easier to remember past data in memory and solves the vanishing gradient problem. LSTM networks are able to selectively remember patterns for long periods of time, handled through memory blocks called cells. Each cell passes two things to the next cell: the cell state and the hidden state, while each memory block is responsible for remembering things. Manipulations to memory is done through three different gates: the forget gate, input gate, and output gate [9].

The forget gate removes information from the cell state, as it may no longer be required for the LSTM to understand things or it may be of lesser importance. The forget gate also takes the output from the previous cell and the current input, and these values are multiplied by the weight matrices then added with bias. Then, following typical neural network steps, the sigmoid function is applied and a value in the range of 0 to 1 is output. If the value is 0, the forget gate wants to completely forget that piece of information. If the value is 1, the forget gate wants to remember the entire piece of information. The vector output from the sigmoid function is multiplied to the cell state, towards the next step [9].

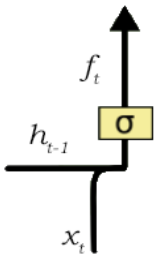


**Figure 1: Structure of the forget gate.**

The input gate controls the addition of information to the cell. This is a three step process. First, there is a sigmoid function similar to that of the forget gate to regulate what values need to be added. Next, a vector is created that contains all possible values to be added using the tanh function, outputting values in the range of -1 to 1. Finally, the values obtained from the sigmoid function and the tanh function are multiplied, and this value is added to the cell state. This process makes sure that the information added to the network is not redundant and is important [9].

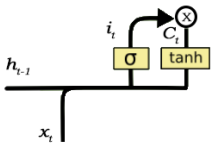


**Figure 2: Structure of the input gate**

The third gate is the output gate, which selects useful information from the current cell state to show as output. The process for the output gate is similar to that of the input gate and follows three steps. First, a vector is created after applying the tanh function to the cell, scaling the values. Next, a regulatory filter using the sigmoid function is used on the output from the previous cell and the current input. Finally, the value from the filter and vector from the first step are multiplied and outputted from the current cell and to the hidden state of the next [9].

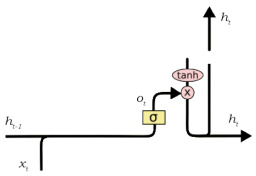


**Figure 3: Structure of the output gate**

## 4 METHODOLOGY

We chose to use the LSTM algorithm because it is well-suited for application on time-based data, even with given lags of unknown duration [4]. As the LSTM is able to remember and is quite insensitive to gap length, sentiment analysis on given topics of Twitter data over time may be possible.

For the preprocessing of the raw data, we removed any strings beginning with the symbol "@", as this denotes a username and depending on its construction, may potentially sway the sentiment result. Words beginning with a hashtag are processed as separate words, as they are indicative of a topic or a category and not a part of the sentence.

After the data is preprocessed, noise removal is the next step. URLs, stop words, punctuation, are some examples of noise that is not necessary for interpretation of the text. Cleansing the data of these things makes it easier for the system to process. Then, we implemented feature selection, which is the process used to extract the required features from the ingested dataset such as tweet ID, tweet contents, date, and sentiment annotation. The basic LSTM model is generated with the threshold changed to account for the sentiment of the tweets to improve accuracy. In addition, code was written to preprocess the tweets from sentences to a string array. Individual words are assigned values between 0 to 1, depending on the sentiment and fed into the manual LSTM network. The manual model returns a single number, and it determines if the sentiment is positive or negative.

We use word embedding available through python to map individual words to vectors of real numbers. Word2vec is one of the most popular techniques, utilizing a two layer neural network with one input layer, one hidden layer, and one output later to automatically learn relationships between words. It does this by manipulating natural language into computer ingestible data. We are using the continuous bag of words (CBOW) algorithm for word2vec, ingesting multiple inputs and context to predict a target word. In this case, we are changing tweets into vectors, so that CBOW and LSTM may be applied to detect a specific sentiment [6]. These outputted sentiments include positive and negative, and of course, neutral.

Word2vec has various key hyperparameters that may be selected to improve the accuracy of the output. We have chosen four parameters for testing: sequence length, epoch, batch size, and sentiment thresholds. Sequence length is the size of the input text. Epoch is the number of training steps. Batch size is the target size, determining the length of each epoch of the neural network. Sentiment thresholds is the cutoff score towards either a positive or negative sentiment output. Anything in between will be automatically categorized as a negative. All hyperparameters may influence the size of the neutral network, controlling the space and resources required to run the code.

## 5 RESULTS AND ANALYSIS

The Kaggle dataset, sentiment140, contains 1,600,000 tweets extracted using the Twitter API. Though the dataset description has noted that there are three sentiment annotations, the available data only contains positive (4) and negative (0) as seen in the figure below. As such, this dataset will be considered as one structured for binary classification [3]. Though this dataset does not contain the neutral classification in the dataset, it's arguably the most widely used for Twitter sentiment analysis and contains a larger sample set for training and testing.

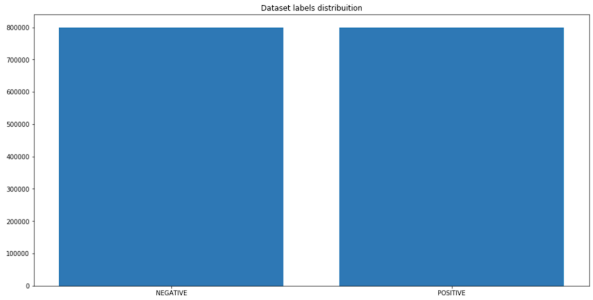


**Figure 4: Dataset Output Split**

For our training process, we utilized the supervised learning scheme and ingested a training dataset of 1.28 million tweets into our model, which was done using Google Colab environment and our local machines. Based on the recommendations stated by Xia and Zhao in their research paper, "LSTM based Sentiment Analysis for Patient Experience Narratives in E-survey Tools," four hyperparamters were tuned: SEQUENCE_LENGTH, EPOCHS, BATCH_SIZE, and sentiment THRESHOLD(s) [1]. The first few sets of parameters were adjusted based on what our hardware and system resources could handle, testing the limits with different values for TRAIN_BATCH, and EPOCHS. Eventually, we discovered the amount of workload our machines could handle, and moved on to start tuning the SEQUENCE_LENGTH and sentiment thresholds to discover the best accuracy values. Though the average training time each iteration took was approximately 4 hours, we eventually produced the following results in Figure 5.

The values for accuracy are measured against the binary classification of the sentiment140 dataset. This paper is not intended to develop a model the existing LSTM model to predict sentiment analysis. Rather, it's to showcase the importance of the neutral classification. Therefore, once the model provided a decimal number, ranging from 0 to 1, the declared sentiment thresholds would identify the tweet as either positive (i.e., given the number ranged from 0-0.4) or negative (i.e., given the number ranged from 0.7-1). Anything between THRESHOLD 1 and THRESHOLD 2 would be declared neutral. Then, we purposely tested the three sentiment outputs against the binary classification.

According to the results, we got accuracy values at approximately 80%. Not bad but not the best for a machine learning algorithm.

| SEQUENCE_LEN | EPOCHS | BATCH_SIZE | THRESHOLD 1 | THRESHOLD 2 | Results |
|---|---|---|---|---|---|
| 300 | 6 | 1024 | 0.4 | 0.7 | Precision: 0.79<br>Recall: 0.79<br>F1-store: 0.79<br>Accuracy: 0.7894 |
| 400 | 4 | 1024 | 0.3 | 0.7 | Precision: 0.78<br>Recall: 0.78<br>F1-store: 0.78<br>Accuracy: 0.7845 |
| 300 | 8 | 512 | 0.3 | 0.7 | Precision: 0.78<br>Recall: 0.79<br>F1-store: 0.79<br>Accuracy: 0.7887 |
| 250 | 5 | 512 | 0.3 | 0.7 | Precision: 0.79<br>Recall: 0.79<br>F1-store: 0.79<br>Accuracy: 0.7889 |
| 250 | 8 | 256 | 0.4 | 0.7 | Precision: 0.78<br>Recall: 0.78<br>F1-store: 0.77<br>Accuracy: 0.7712 |

**Figure 5: Training results with varying parameter values**

| Sentiment | Tweet |
|---|---|
| 0 | @marieclr I was serious  LOL |
| 0 | wanttss to go out |
| 0 | @cheechbud i think ur right!! hahaha!! 4.5 hrs now!! |
| 0 | @ITS_NEMESIS ------- |
| 0 | @LevenRambin: Take it easy, and be good to you. |
| 0 | Soaring eagle is two hours away. |
| 0 | heading out for a run |
| 4 | http://twitpic.com/4huau - we measured ourselves |
| 4 | @TrinaWright Do I look it in this photo? http://bit.ly/19cHua |
| 4 | Eating Fish |
| 4 | Testing. testing. is this on? someone at reply me. |
| 4 | Useing qtweeter on my iPod for the first time |
| 4 | Ms. Marple time |
| 4 | is new on twitter. I'm looking around. |

**Figure 6: Dataset Sample with positive (4) and negative (0) sentiments.**

However, this comes into question why sentiment140, a dataset widely used in research for prominent publishers, only has two sentiment classifications. A couple of research paper examples include "Sentiment Analysis of Tweets using Deep Neural Architectures" by Cai, published through Stanford, and "Combining a rule-based classifier with weakly supervised learning for twitter sentiment analysis" by Siddiqua and Ahsan, published through IEEE. Exploring the data, we notice a few tweets, sampled in the figure below, that may be better classified as neutral considering the lack of context available for them.

With the given dataset like this that the model must base its predictive accuracy upon, this highlights the importance of the neutral category. Even if the training set must be decreased, it's more beneficial for the model to test on tweets that have clear sentiments, rather than those simply classified without justification or background context. This, combined with the binary classification, may easily skew the results and cause the model to overfit since even arguably non-sentiment and/or neutral tweets will need to be forced into either positive or negative sentiments. This would cause even the 80% accuracy derived from our model become questionable and arbitrary to other datasets.

## 6 CONCLUSION AND FUTURE WORK

In this paper, we discussed the importance of the neutral category in studies related to Twitter sentiment analysis and quality of the dataset that the model is being trained and tested on. As mentioned in the results and analysis section, the neutral category should exist as its own state to denote lack of sentiment in text. Otherwise, the data may be overfit by the algorithm due to the input being forced into binary classification. There may also be situations where the neutral text may randomly occur more often than clear positive or negative texts in the training dataset [5].

A further related study that may be done on this topic is sarcasm detection. Sarcasm is simply when the individual states the opposite of what is being implied and in this scenario, through Twitter's tweets. It's no myth that sarcasm is naturally difficult to analyze through machine learning. Even humans have trouble because to completely understand the underlying message, one must be knowledgeable of the context of the situation, the culture, and potentially the specific topic and involved parties of the sarcastic tweet. Current research on sarcasm detection typically focuses on the ability to classify whether a tweet is sarcastic or not. This is also considering that sarcastic comments are susceptible to netspeak, or Internet slang, and that makes it even more difficult to train machine learning algorithms, which references standard language, to efficiently identify them. Nevertheless, sarcasm detection is definitely helpful in order to finetune Twitter sentiment analysis, and there are numerous studies that may back this. One is the paper done by Maynard and Greenwood that observed in improvement of sentiment analysis by developing rules for sarcasm detection through identified sarcastic hashtags [8].

Nevertheless, sentiment detection tools are overall still in the early stages of development, particularly due to its difficulty in analyzing text from social media yet is very much a current headline. Improvements may span from testing with varying algorithms aside from the common LSTM and SVM to obtaining more user-input Twitter sentiment data, perhaps through surveys with the submitted sentiment averaged for a more particularized data point. After all, if we would benefit from a neutral sentiment to avoid the black and white pitfall given only by positive and negative classifications, perhaps we may expand this practice to each tweet in the given dataset. A positive tweet may not always be considered positive to another. The same sentiment may be said for negative and neutral tweets.

# REFERENCES


[1] Jing Wang Jing Liu Chenxi Xia, Dong Zhao and Jingdong Ma. 2018. LSTM based Sentiment Analysis for Patient Experience Narratives in E-survey Tools". https://link-springer-com.libproxy.utdallas.edu/content/pdf/10.1007%2F978-3-030-03649-2_23.pdf

[2] Dan Goodin. [n. d.]. "Twitter Usage Statistics - Internet Live Stats". https://www.internetlivestats.com/twitter-statistics/

[3] Kaggle. 2017. "Sentiment140 dataset with 1.6 million tweets". https://www.kaggle.com/kazanova/sentiment140

[4] E. Kang. 2017. "Long Short-Term Memory (LSTM): Concept". https://medium.com/@kangeugine/long-short-term-memory-lstm-concept-cb3283934359

[5] M. Koppel and J. Schler. 2005. "The Importance of Neutral Examples for Learning Sentiment". http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.9735&rep=rep1&type=pdf

[6] Z. Li. 2019. A Beginner's Guide to Word Embedding with Gensim Word2Vec Model". https://towardsdatascience.com/a-beginners-guide-to-word-embedding-with-gensim-word2vec-model-5970fa56cc92

[7] D. Graziotin M. Mäntylä and M. Kuutila. 2017. "The Evolution of Sentiment Analysis - A Review of Research Topics, Venues, and Top Cited Papers". https://arxiv.org/ftp/arxiv/papers/1612/1612.01556.pdf

[8] D. Maynard and M. Greenwood. 2014. "Who cares about sarcastic tweets? Investigating the impact of sarcasm on sentiment analysis". http://eprints.whiterose.ac.uk/130763/1/sarcasm.pdf

[9] P. Srivastava. 2017. "Essentials of Deep Learning : Introduction to Long Short Term Memory". https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/

[10] V. Vryniotis. 2013. "The importance of Neutral Class in Sentiment Analysis". https://blog.datumbox.com/the-importance-of-neutral-class-in-sentiment-analysis/