

# Predicting Georgia General Elections Results with Tweet Sentiment Analysis

**Rakesh Bhavsar**

Computer Science Department  
The University of Georgia  
Athens, USA  
rakesh.bhavsar25@uga.edu

**Shrutika Gamare**

Computer Science Department  
The University of Georgia  
Athens, USA  
svg17584@uga.edu

**Ayush Jaiswal**

Computer Science Department  
The University of Georgia  
Athens, USA  
ayush.jaiswal25@uga.edu

**Radhika Bhavsar**

Computer Science Department  
The University of Georgia  
Athens, USA  
radhika.bhavsar@uga.edu

**Abstract**—Election empowers citizens to choose their leaders. The public sentiments towards the leader highly influences the decision of the general public. People use Twitter as the social media platform for speaking their views out. Twitter will transform democracy, allowing citizens and politicians to communicate, connect and interact in ways never thought before. Social media tools such as Twitter is now considered as politically transformative communication technologies as radio and television. Twitter influences elections and public opinion poll results. Therefore, it is required to understand how citizens and politicians worldwide share political information and opinion via social media.

This paper presents Predicting Georgia Election Results with Twitter Sentiment Analysis to analyze citizen tweets which can be used as a good predictor of public opinion regarding the Georgia elections and the two candidates (Brian Kemp vs Stacey Abrams). To perform sentiment analysis of tweets we have implemented various machine learning algorithms such as Dictionary based, LSTM using Glove & Bidirectional LSTM using Glove. Our results show that LSTM achieves an accuracy of 94% in predicting the sentiment of a tweet as positive or negative for the respective candidate.

**Keywords**—Cloud Computing, Machine Learning, Natural Language Processing, LSTM, Glove.

## I. INTRODUCTION

Twitter is an online microblogging tool that disseminates millions of messages on topics spread across all industries from entertainment to sports, health to business, even Political orientation and sentiments are being widely expressed on Twitter. Twitter has become a foremost widespread communication tool on the internet. Previously, estimating the sentiments towards any political candidate were done using polls/surveys. These approaches weren't as accurate as using Twitter as a data source platform for extracting the sentiments.

With every event or move of a political candidate, the sentiments of people are being expressed on Twitter and help to accurately estimate public opinions. The General Election was held in Georgia on November 6, 2018 to elect the next Governor of Georgia.

The Two candidates were Miss. Stacey Abrams from the Democratic party and Mr. Brian Kemp from the Republican party. We made use of Twitter tweets to predict the popularity of these two candidates and therefore extrapolate their chances of winning the election. We collected tweets based on the keywords, mainly consisting of most popular hashtags. Collected tweets depicted public opinions, negative campaigning, political polarizations and daily whereabouts of the two candidates. These tweets were hand labelled to have four classes: negative sentiment for Mr. Brian was labelled 0, positive sentiment for Mr. Brian was labelled 1, negative sentiment for Miss Stacey was labelled 2, positive sentiment for Miss Stacey was labelled 3. These labelled tweets were preprocessed. We choose two baseline models,

1. Dictionary based approach
2. LSTM using glove

Dictionary based approach worked with opinion lexicon dataset. This dataset had both list of positive and negative words. Parsing the words in a tweet against this dataset calculated the polarity of the tweet. For positive count greater than or equal to negative count, the tweet was assigned a positive sentiment. For equal values of the positive and negative counts the tweet was assigned a positive sentiment.

LSTM using Glove:

We have used Glove: Global Vectors for Word Representation with LSTM which is an unsupervised learning algorithm for obtaining vector representations for words. Glove is gaining popularity as a word embedding matrix in natural language processing for its useful representation of word vectors. We chose Glove because similar words have similar vectors and it provides for better performance even with memory intensive tasks.

LSTM achieves almost human-level of sequence generation quality. LSTM networks have some internal contextual state cells that act as long-term or short-term memory cells. The output of the LSTM network is modulated by the state of these cells. This is a very important property when we need

the prediction of the neural network to depend on the historical context of inputs, rather than only on the very last input.

To learn of improvements on both our chosen baseline models, we thought of using *Bidirectional LSTM* using GloVe. Bidirectional LSTMs are an extension of traditional LSTMs that can improve model performance on sequence classification problems.

In problems where all time steps of the input sequence are available, Bidirectional LSTMs train two instead of one LSTMs on the input sequence. The first on the input sequence as-is and the second on a reversed copy of the input sequence. This can provide additional context to the network and result in faster and even fuller learning on the problem.

Our selected model performed well to predict more positive sentiments towards Miss. Stacey as the next Governor of Georgia. Accuracy as good as 94% was achieved with a better loss function value indicating minimal errors. Although Mr. Brian Kemp was the elected governor, the difference in votes between both the candidates was marginal with Brian Kemp sharing 50% and Stacey Abrams sharing 49% of votes.

We explain related work in the next section. It is followed by our approach in section III. We present our results in section IV, which is followed by discussion and lessons learned in section V. Finally, we explain conclusion in section VI which is then followed by limitations and references in section VII & VIII respectively.

## II. RELATED WORK

Sentiment analysis is a growing area of Natural Language Processing with research ranging from document level classification to learning the polarity of words and phrases. A lot of research has been done on Twitter data in order to classify the tweets and analyze the results. As the sentiment analysis of tweets has gained popularity in recent years, the sentiments of queries generated by users has been calculated by a lot of machine learning techniques. Most people have performed sentiment analysis using twitter data for the prediction of the 2016 US elections. In the paper [2], the authors manually labelled the dataset of 36,465 tweets and then performed 5-fold cross validation. Machine learning models such as Naive Base, SVM and Dictionary based were used. Support Vector Machine obtained .75 as precision and .78 as recall predicting BJP(Bharatiya Janta Party) to win and BPJ did win 60 out of 126 of the constituencies in India in the year 2016. In the [3] paper, the authors experimented with deep learning models in an effort to build the best possible sentiment classifier for tweets. The authors had made use of Logistic regression on 1-3 grams baseline, CNN with different word embeddings, LSTM with different word embeddings and ensemble of 10 CNNs and 10 LSTMs with different hyper-parameters and different pre-training strategies. The Paper [26], is about how the public sentiment towards candidates will influence the future leader of USA of 2016 November presidential election. It focused on what are the public views the top election candidates, namely Donald Trump, Hillary Clinton, Ben Carson, and Bernie Sanders. Twitter provided them with live access to opinions about the election across the globe. The project mainly focused on solving the real-world

problem of understanding the current sentiment towards election candidates based on the public's live opinions and emotions rather than off of smaller, localized polls typically done by mainstream media corporations. Additionally, they worked on relevant assessment of sentiments towards candidates than the media and polls have and classification of positive or negative sentiment towards a candidate by classifying tweets into five emotions - happy, sad, fear, laughter, and angry. They used Naive Bayes as well as SVM on classifying tweets with 49.02% accuracy in which they found SVM was the most accurate classifier.

There have been twitter sentiment analysis performed for U.S Presidential elections in the past. We attempted to analyze the sentiments of twitter towards the Georgia Midterm elections held on November 06, 2018.

## III. APPROACH

In this section, we explain our approach to predict Georgia Election Results with Tweet Sentiment Analysis. Our approach mainly consists of two important steps:

### A. Data Preparation

### B. Model Training and Assessment

Data Preparation mainly consists of collecting, preprocessing and labelling of data, whereas model training and assessment involves implementation of various machine learning algorithms and evaluating their results in the form of graphical visuals. Figure show architecture of our entire system.

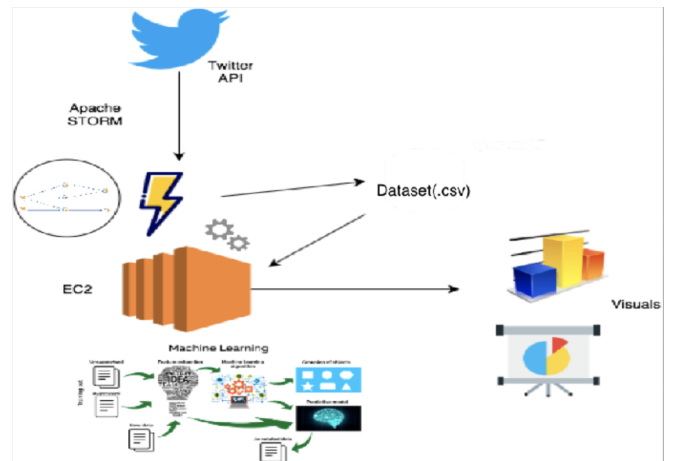


Figure 1: System Architecture

### A. Data Preparation

#### a) Data Collection

We gathered thousands of tweets using Twitter Streaming API on Apache Storm. A Storm Application which takes in data and processes it is known as Storm Topology. Every Storm topology has two important components. First, is Spout which simply collects tweets from the source that is Twitter Streaming API. The Spout is the connection between the storm application and the external data source that is pushing data to the storm topology. The Spout is responsible for pushing data into the rest of the storm topology. The other components in a storm topology are bolts. Bolts are the component which actually perform the tweet processing which includes extracting tweet information such as tweet ID, screen name, handler, tweet, published date and writing this information into a .CSV file. Once Storm Topology consisting of spout and

bolts is created, it runs on Storm Cluster. A Storm Cluster is a group of processes which are centrally coordinated. There would one single process which knows what all processes are running & where they are running. This single coordinating process is known as the Nimbus. When you want to start running the storm application, you would create a topology and put it in a .jar and then submit that topology to the Nimbus. The Nimbus then figures out what all spouts and bolts are required for running the topology. Data collection is performed from October 20 to November 6, 2018 using above described storm topology. The collected tweets are mainly written in English. Figure shows storm topology for data collection process.

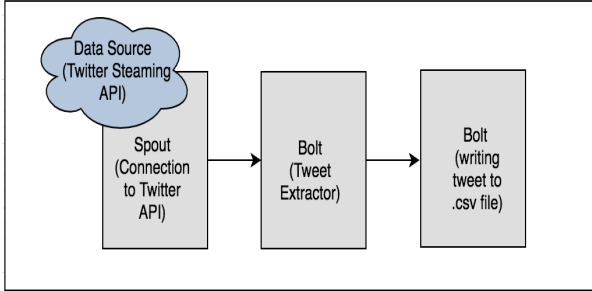


Figure 2: Storm Topology

#### b) Data Preprocessing

Tweets collected from Twitter generally contains noisy information such as emoticons, user mentions, URL's, pictures, etc. We have applied various preprocessing steps in-order to standardize the dataset so that it can be easily learned by any classifier. We have preprocessed the data by removing stop words replacing URLs, Hashtags, handles, check for valid alphabet. Data Preprocessing involves:

- Replacing Emoji: In-order to express feeling or views people often use different types of emoticons such as Love <3, Smile :), Sad :(, Cry:” (, etc. We have replaced such emoji with EMO\_POS or EMO\_NEG depending upon whether it is conveying a positive or negative sentiment respectively.
- Replacing URL: People often share URLs of other web pages while posting a tweet. Such URLs are of no use to us in tweet classification. Therefore, we have replaced hyperlinks with word URL during preprocessing of data.
- Removing Hash: Users often use hashtags to favor a particular topic on twitter. Hashtags consist of a symbol # followed by a word. We have removed # symbol and just kept the word. For instance, #VoteBlue is replaced by VoteBlue.
- Replacing User Mentions: All twitter users have a unique identifier associated with them called handle. Users often mention other users in their tweets by @handle. We have replaced all user mentions with the word USER\_MENTION.

Table 1: Tweet Example after Preprocessing

Before Preprocessing	After Preprocessing
"@BrianKempGA :loudspeaker:ATTENTION #GA VOTERS:loudspeaker:  The COMMUNIST PARTY USA supports @staceyabrams!!YES, you read that correctly. Here it is in black and white. Stacey Abrams is TOO EXTREME for Georgia!!VOTE @BrianKempGA for #Governor!!  #GaPol #MAGA #GaFirst #GaGOP https://t.co/GLs2wOOiKa	briankempga Hashtag the communist party usa supports you read that correctly here it is in black and white stacey abrams is too extreme for briankempga for Hashtag Hashtag Hashtag Hashtag Hashtag URL

#### c) Data Labelling

- Collected tweets are labelled manually by all the team members. We have labelled a total of five thousand tweets. If we think tweet does not relate to a particular candidate or if the tweet is neutral in nature, then we have discarded such tweets.
- Each candidate had positive and negative sentiment respectively, that makes for 4 classes as follows:
  - 0 -> Negative sentiment for Brian Kemp
  - 1 -> Positive sentiment for Brian Kemp
  - 2 -> Negative sentiment for Stacey Abrams
  - 3 -> Positive sentiment for Stacey Abrams

Table 2: Tweets labelled for their sentiment polarity.

28	staceyabrams voted against online voter regist...	1
29	remember when folks were bent about staceyabra...	0
...	...	...
4508	USER_MENTION USER_MENTION staceyabrams because...	3
4509	USER_MENTION USER_MENTION staceyabrams fuuueckkk...	2

## B. Model Training & Assessment

After the tweets were labelled and preprocessed, we choose to feed the tweets to our baseline models.

As already mentioned, we had two baseline models: Dictionary based approach and LSTM using glove.

Dictionary based model was chosen to try out the simplest form of model training and testing.

We used opinion lexicon dataset that comprised of positive and negative words. For each row, words were tokenized and checked if those belonged to either positive word list or negative word list. For every positive word in a row pos\_count variable was incremented by 1. Likewise, for every negative word, neg\_count variable was increased by 1. The sentiment towards a tweet was predicted to be positive if pos\_count was greater than or equal to neg\_count and the tweet was labelled 1. The tweet was labelled 0 otherwise.

Using this dataset, the model was first trained using the classify method. For the purpose of testing, we had hand labelled the test file tweets. This test file without the label column were passed to the classify method that produced the labels for each row in the test file. To validate the prediction, the predicted label for each tweet was checked against the already hand labelled tweets.

The dictionary-based approach gave us an accuracy of 63.38. This minimal accuracy was expected, however we wanted to try out the most basic approach for experience.

We came across a few reads that performed sentiment analysis using LSTM models and how LSTM would be applied to solve natural language processing related tasks.

### a) Why LSTM (Long Short-Term Memory)?

We collected around five thousand preprocessed tweets in order to train the model. These tweets are long range of connecting sentences. LSTM allows to learn very long-range connection in sequence. It has memory cell  $c^{<t>}$  along with 3 gates i.e. update, forget and output gate.

Formula to calculate memory cell value:

$$\tilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[a^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_f = \sigma(W_f[a^{<t-1>}, x^{<t>}] + b_f)$$

$$\Gamma_o = \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_o * \tanh c^{<t>}$$

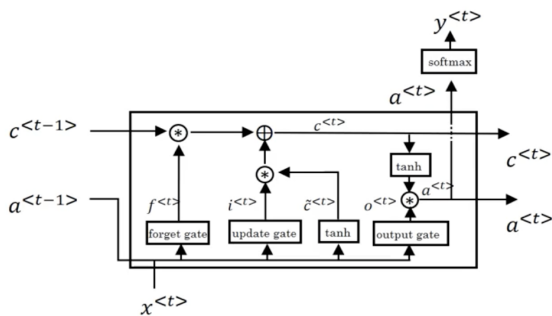


Figure 3: LSTM flow

As shown in figure,  $a^{<t-1>}, x^{<t>}$  is used to compute the value of update, forget and output gate. They also go to tanh to compute  $\tilde{c}^{<t>}$ . All these values are combined to get  $c^{<t>}$  from previous  $c^{<t-1>}$ . The core thing about LSTM is, line at top shows how old value can be memorized even for long time. It is relatively easy for LSTM to pass old value i.e.  $c^{<t-1>}$ . This is how LSTM stores certain value for long time.

Advantage of LSTM:

Powerful, Useful for big scale application, deals with the problem of vanishing gradient, human-level of sequence generation quality, long-term or short-term memory cells, maintains historical context of inputs, improved performance with Glove.

### b) Bidirectional LSTM:

Previously, most people used unidirectional or forward directional RNN in order to predicate the component. Bidirectional has forward recurring component like  $a^{<1>}, a^{<2>} \dots$  and  $y^{<1>}, y^{<2>}, \dots$  as predicted output, with input information in  $x^{<1>}, x^{<2>}, \dots$  which processes from left to right. In Bidirectional LSTM, there is an addition of backward connection which works from right to left. If we supposed to calculate the prediction value of middle component of sentence, Bidirectional LSTM takes into account the input information from past as well as future along with present input information. So, using all this input information we can calculate the predicted value of middle component which ultimately increases the accuracy of predication.

### c) Glove: Global Vectors for Word Representation

Natural language processing makes use of word embeddings. One such word embedding matrix we found useful was Glove. Previously SVM approach was used to tackle the text classification problems. N-grams were typically used to find words that have meaning together. However, it leads to the problem of dimensionality. Word vectors tackle this problem as similar words have similar vectors.

Glove is an unsupervised learning algorithm for obtaining vector representations for words. A log-bilinear model with a weighted least-squares objective.

Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

Highlights:

1. Nearest neighbors:

Euclidean distance provides an effective method for measuring the linguistic or semantic similarity

2. Linear substructures:

It is necessary for a model to associate more than a single number to the word pair. Glove is designed in order that such vector differences.

The training objective of Glove is to learn word vectors such that their dot product equals the logarithm of the words' probability of co-occurrence. This helps to identify all kinds of similar words. We used glove with 100 dimensions.



#### IV. EXPERIMENTS

##### A. Deployment and Evaluation System

We used Amazon's EC2 p2.xlarge instance for Model Training and Testing. Also, all the experiment evaluation is performed on the same Amazon's EC2 P2 instance machine. Amazon EC2 P2 Instances [8] are powerful, scalable instances that provide GPU-based parallel compute capabilities. P2 instances are ideally suited for machine learning. Pre-installed with popular deep learning frameworks such as Keras.

Table 3: p2.xlarge instance details

GPUs	vCPUs	RAM (GiB)	Network Bandwidth	Price Per Hour*	RI Price Per Hour**
1	4	61	High	\$0.90	\$0.425

\* Pricing for US East (N. Virginia) and US West (Oregon)

\*\*3-Year Partial Upfront RI

##### B. Baseline I

For baseline I, we are using the dictionary-based algorithm for assigning sentiment to a given tweet. In the early days of quantitative text analysis, word-frequency counting in texts was a common mode of analysis. At first, we tokenize the processed tweet and count the number of positive and negative words in a given tweet. For list of positive and negative word, we have use Opinion Lexicon [9] (A list of English positive and negative opinion words or sentiment words around 6800 words). Based on the polarity of the count, we assign positive or negative sentiment to a tweet. In cases when the numbers of positive and negative words are equal, we assign positive sentiment. Using this baseline model, we achieve model's classification accuracy of 63.38 % on the 4538 hand labelled dataset.

The reason of using dictionary-based algorithm for baseline I was to observe the drastic change in the model accuracy between simple approach to highly efficient *LSTM* using *GLOVE* for twitter sentiment analysis. We have improved accuracy from 63.38% for Dictionary-Based to 93.5 % for Bidirectional *LSTM* using *GLOVE*.

##### C. Baseline II: LSTM using GLoVe

Long Short-Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. LSTMs are explicitly designed to avoid the long-term dependency problem. These have widely been used for speech recognition, language modeling, sentiment analysis and text prediction. [10] GloVe is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

[13] We are using GLoVe.6B.100d, which is a pre-trained word vectors. This data is made available under the Public Domain Dedication and License v1.0 whose full text can be found at:

<http://www.opendatacommons.org/licenses/pddl/1.0/>.

For Baseline II using *LSTM GLoVe*, we divided the hand labelled dataset into 67-33 ratio of 4538 tweet count, where 67% comprised of Training Dataset and 33% of Validation Dataset. We created Embedding Matrix using glove.6B.100d that has shape of (2000,100), where 2000 is the top word features and 100 is the vector size. Below are the *LSTM GLoVe* model parameters,

- SpatialDropout1D = 0.4
- dropout = 0.2, recurrent\_dropout = 0.2
- activation = softmax
- loss = categorical\_crossentropy

Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 100, 100)	200000
spatial_dropout1d_1 (Spatial)	(None, 100, 100)	0
lstm_1 (LSTM)	(None, 196)	232848
dense_1 (Dense)	(None, 4)	788
Total params: 433,636		
Trainable params: 433,636		
Non-trainable params: 0		
None		

Figure 4: LSTM using GLoVe Model summary

We trained the model with batch size = 32 and epochs = 50. After 50 epochs, the loss was 0.0211 and accuracy 0.9924.

For validation, we used 1498 labelled tweets and achieved an accuracy of 94% with a loss score of 31%.

##### D. Model - Bidirectional LSTM using GLoVe

*LSTM* in its core, preserves information from inputs that has already passed through it using the hidden state. Unidirectional *LSTM* only preserves information of the past because the only inputs it has seen are from the past. Bidirectional Recurrent Neural Networks (BRNN) connects two hidden layers of opposite directions to the same output. With this form of generative deep learning, the output layer can get information from past (backwards) and future (forward) states simultaneously [14]. Bidirectional LSTM will run inputs in two ways, one from past to future and one from future to past. The two hidden states combined enables at any point in time to preserve information from both past and future. [15]

For Bidirectional LSTM using GLoVe Model, we divided the hand labelled dataset into 67-33 ratio of 4538 tweets, where 67% comprised of Training Dataset and 33% of Validation Dataset.

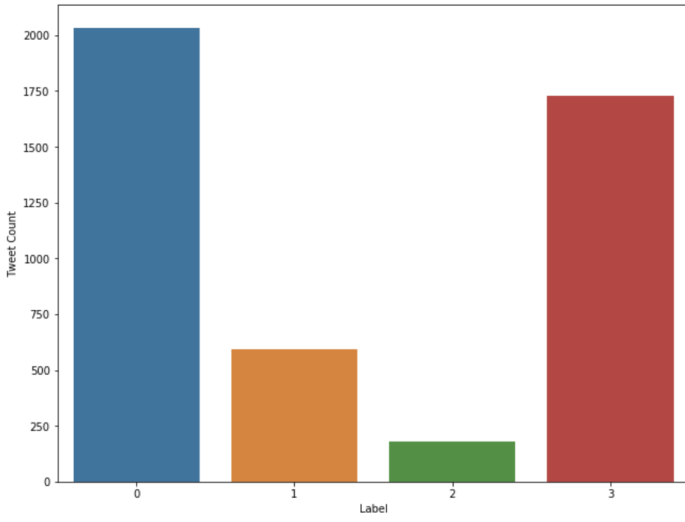


Figure 5: Labeled Tweets Distribution

- 0 - Negative sentiment for Brian Kemp
- 1 - Positive sentiment for Brian Kemp
- 2 - Negative sentiment for Stacey Abrams
- 3 - Positive sentiment for Stacey Abrams

We created Embedding Matrix using glove.6B.100d that has shape of (2000,100), where 2000 is the top word features and 100 is the vector size. Below are the Bidirectional LSTM using GLoVe Model parameter,

- Bidirectional(LSTM(392,return\_sequences=True, dropout=0.1, recurrent\_dropout=0.1))
- Dense(256, activation="relu")
- Dropout(0.1)
- Dense(128, activation="relu")
- Dropout(0.1)
- Dense(4, activation="softmax")
- loss = categorical\_crossentropy

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	(None, 100)	0
embedding_3 (Embedding)	(None, 100, 100)	200000
bidirectional_2 (Bidirection	(None, 100, 784)	1546048
global_max_pooling1d_3 (Glob	(None, 784)	0
dense_3 (Dense)	(None, 256)	200960
dropout_2 (Dropout)	(None, 256)	0
dense_4 (Dense)	(None, 128)	32896
dropout_3 (Dropout)	(None, 128)	0
dense_5 (Dense)	(None, 4)	516
Total params: 1,980,420		
Trainable params: 1,980,420		
Non-trainable params: 0		

Figure 6: Bidirectional LSTM using GLoVe Model summary

We trained the model with batch size = 32 and epochs = 50. After 50 epochs, the loss was 0.0121 and accuracy 0.9941. For validation, we used 1498 labelled tweets and achieved an accuracy of 93.5% with a loss score of 42.5%.

A confusion matrix is a table that is often used to describe the performance of a classification model (or “classifier”) on a set of test data for which the true values

are known. It allows the visualization of the performance of an algorithm. [16]

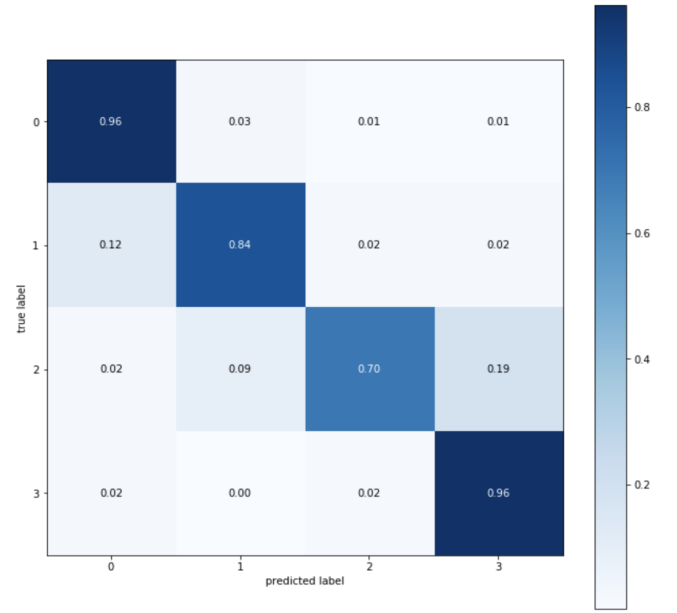


Figure 7: Confusion Matrix

From the Confusion Matrix, we can say that Bidirectional LSTM using GLoVe was able to predict Kemp negative (0) with 96% accuracy, Kemp positive (1) with 84% accuracy, Stacey negative (2) with 70% accuracy and Stacey positive (3) with 96% accuracy.

#### E. Election Prediction

The election date was 6th November 2018. We collected 25K tweets for the period between 1st Nov to 5th Nov 2018. We pre-process the tweets and passed the processed tweets to the Bidirectional LSTM using GLoVe Model. Below bar graph shows the prediction of 25K tweets.

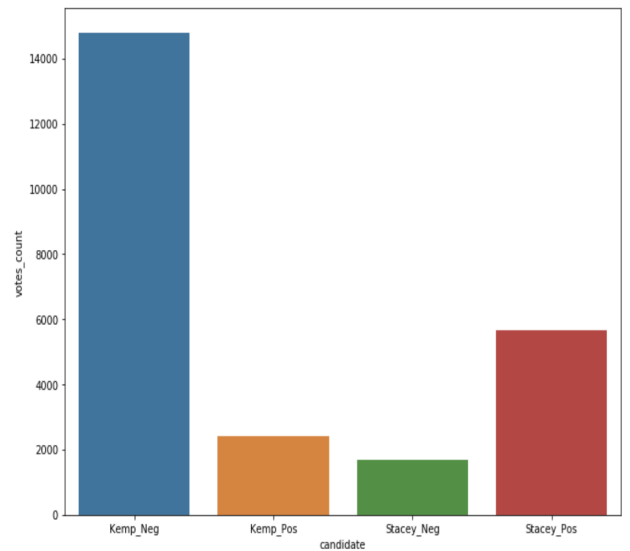


Figure 8: Bar Graph of Bidirectional LSTM using GLoVe Model prediction

Table 4: 25 K Tweet Distribution

Candidate	Tweet Count
Brian Kemp	17100
Stacey Abrams	7800

Table 5: 25K Tweet Percentage Distribution




Candidate	Positive %	Negative %
Brian Kemp	12%	87%
Stacey Abrams	76%	23%

Our Bidirectional LSTM using GLoVe Model predicts Stacey as the winner of Georgia General Election 2018.

## V. DISCUSSION & LESSONS LEARNED

### A. Discussion

## Georgia Election Results 2018 Election results

Map	Percent	Candidate	Party	Votes	Winner
	50.2%	Brian Kemp	GOP	1,978,408	✓
	48.8%	Stacey Abrams	Dem	1,923,685	
	0.9%	Other		37,235	

100% of precincts reporting (2,634/2,634)  
3,939,328 total votes

Figure 9: Georgia Election 2018 Results

Source: <https://www.politico.com/election-results/2018/georgia>

Our model predicted Stacey Abrams as the winner of the Georgia Election 2018 which contradicts to actual Georgia Election 2018 result. Brian Kemp of GOP party wins by 50.2% of 3,939,328 votes. [21]

According to the prediction behaviour of our Bidirectional LSTM using GLoVe Model, some of the reasons of predicting Stacey Abrams as the winner can be as follows,

1. Twitter dataset which we have collected had many negative tweets against Kemp.
2. The time frame for twitter data collection was between 15<sup>th</sup> Oct - 5<sup>th</sup> Nov. During this phase, twitter was flooded with negative comments against Brian Kemp. Though, there were negative

tweets against Stacey Abrams but lower compared to Brian Kemp.

3. Brian Kemp won with a margin of 1.4% against Stacey Abrams. If we see the other way around, model almost predicted correct considering the minimal margin difference.
4. Twitter cannot be source of predicting results as not every voter is on Twitter platform.

### B. Challenges

We first started tweet data collection using Apache Storm with the help of twitter4j library. At first tweets were truncated, if it had more than 140 characters. To get full tweet of 280 characters by enabling the TweetModeExtended in the configuration builder as follows:

```
ConfigurationBuilder configurationBuilder = new
ConfigurationBuilder();
configurationBuilder.setTweetModeExtended(true);
```

One of the most challenging task was to handle label 4538 filter tweets. It was a time-consuming process at it involved reading individual tweet and make a sense out of it for a positive or negative sentiment.

### C. Lessons Learned

- Learned Amazon Web Services (AWS):
  - We have performed training & testing of all machine learning models on amazon EC2 Instance (p2.xlarge).
- Learned using Apache Storm:
  - We have downloaded all tweets from Twitter Streaming API using Apache Storm.
- Learned Alexa Skill Development.
  - We have also developed an Alexa Skill Set to announce Georgia elections results by developing a webservice using Java Spring Boot framework and connecting it with Amazon developer portal using https endpoint. In-order to create a https endpoint we have used ngrok which helps to expose local servers behind NATs and firewalls to the public internet over secure tunnels. We have faced some challenges in making our endpoint https as we need to provide valid certificates.

### D. Source Code

Source code of our project can be found at below location:  
<https://github.com/RakeshrBhavsar/Twitter-Sentimental-Analysis-using-Apache-Storm.git>

### E. System Demo

System Demo of our project can be found at below location:  
<https://www.youtube.com/watch?v=XHiMdXLawt8>  
<https://www.youtube.com/watch?v=bcYLN6xRgDk>

## VI. CONCLUSION

We have made use of Apache STORM to collect real time asynchronous tweets based on keywords. The spout of our STORM topology was integrated with Twitter API which handed the tweets to bolts. One of our bolt was responsible for writing the collected tweets onto an csv document.

The tweets were hand labelled with four classes for our models to learn and identify the sentiment against any tweet. The total number of tweets labelled were 4538. Our dataset comprised of the textual content alongside the labels: 0,1,2,3 each representing four classes of sentiment towards the contenders.

We had two baseline models: Dictionary based using opinion lexicon and LSTM using glove. Glove word embedding helped the model learn and extract sentiments with better accuracy. Dictionary based approach gave us an accuracy of 63.38. LSTM with glove gave an accuracy of 94.00 (Score: 0.31). We used Bidirectional LSTM using glove as our final chosen model to improve the baseline. Bidirectional LSTMs are an extension of traditional LSTMs that can improve model performance on sequence classification problems. Using Bidirectional LSTM using glove gave an accuracy of 93.5 (Score: 0.425). We were able to obtain a minimal score value for both our LSTM models, lower loss implies lower prediction error. Our model when tested against tweets from November 1, 2018 to November 5, 2018 predicted more positive sentiment towards Miss Stacey Abrams acclaiming her as the new Governor of Georgia. Although Mr. Brian Kemp won the election, the votes margin between Mr. Brian Kemp and Miss Stacey Abrams is minimal. We noticed that a lot of tweets were in proclivity of Miss Stacey Abrams. All our training and testing was performed on an ec2 instance p2x.Large. We also made use of lambda function to work with Alexa. Alexa was programmed to announce the elected candidate using data from a publicly available API[21].

## VII. LIMITATION

The limitation of the project is that sarcastic tweets weren't taken care of and neutral tweets weren't classified correctly.

Models could not label neutral tweets accurately because our training dataset didn't comprise of labels for neutral tweets, leaving the models to predict falsely.

Our training dataset comprised of 4938 hand labelled tweets. Had we labelled more number of tweets our model could learn even better and predict more accurately the new tweets exposed to it.

Twitter is one of the popularly used social media platform for sentiment analysis. However, there have been instances from the 2016 presidential elections that a lot of the republican supporters weren't actively contributing to this platform that resulted in flawed predictions. Large chunk of public opinion go neglected because not all the voter base go on to twitter to express their sentiment towards a political candidate. Even with our prediction, the dataset flexed well in favour of Miss Stacey Abrams and so prediction that she is the next elected Governor was viable. However, reality still remains that Mr. Brian Kemp was elected with slightly more number of votes, nullifying the predictions of our model.

## VIII. REFERENCES

- [1] [https://www.youtube.com/watch?v=efWIOCE\\_6HY&list=PL1w8k37X\\_6L\\_s4ncq-swTBvKDWNrSrinl](https://www.youtube.com/watch?v=efWIOCE_6HY&list=PL1w8k37X_6L_s4ncq-swTBvKDWNrSrinl)
- [2] Prediction of Indian Election Using Sentiment Analysis on Hindi Twitter (paper) <https://www.computer.org/csdl/proceedings/big-data/2016/9005/00/07840818.pdf>
- [3] BB twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs <https://arxiv.org/abs/1704.06125>
- [4] Twitter Sentiment <https://nlp.stanford.edu/courses/cs224n/2009/fp/3.pdf>
- [5] Twitter Sentiment Analysis: The Good the Bad and the OMG! <http://www.aaii.org/ocs/index.php/ICWSM/ICWSM11/paper/download/2857/3251?height%3D90%26iframe%3Dtrue%26width%3D90%26>
- [6] A system for real-time Twitter sentiment analysis of 2012 U.S. presidential election cycle. [https://dl.acm.org/ft\\_gateway.cfm?id=2390490&ftid=1304229&dwn=1&CFID=2600189&CFTOKEN=15b416846c5a33d9-A5101CA9-9846-6C80-87BCBBB74EF9BF02](https://dl.acm.org/ft_gateway.cfm?id=2390490&ftid=1304229&dwn=1&CFID=2600189&CFTOKEN=15b416846c5a33d9-A5101CA9-9846-6C80-87BCBBB74EF9BF02)
- [7] Combining lexiconbased and learning-based methods for twitter sentiment analysis. [ftp://altea.dlsi.ua.es/people/armando/OMSA/Combining\\_Lexicon-based\\_and\\_Learning-based\\_Methods.pdf](ftp://altea.dlsi.ua.es/people/armando/OMSA/Combining_Lexicon-based_and_Learning-based_Methods.pdf)
- [8] Sentiment Analysis of Twitter Data. [https://dl.acm.org/ft\\_gateway.cfm?id=2021114&ftid=1013247&dwn=1&CFID=2603244&CFTOKEN=89183d8b15bee40b-A5B4744F-9894-D006-46961C41B240D712](https://dl.acm.org/ft_gateway.cfm?id=2021114&ftid=1013247&dwn=1&CFID=2603244&CFTOKEN=89183d8b15bee40b-A5B4744F-9894-D006-46961C41B240D712)
- [9] Twitter as arena for the authentic outsider: exploring the social media campaigns of Trump and Clinton in the 2016 US presidential election. <http://journals.sagepub.com/doi/abs/10.1177/0267323116682802>
- [10] <https://ngrok.com/product>
- [11] <https://aws.amazon.com/ec2/instance-types/p2/>
- [12] <https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html>
- [13] <https://towardsdatascience.com/understanding-lstm-and-its-quick-implementation-in-keras-for-sentiment-analysis-af410fd85b47>
- [14] <https://chatbotmagazine.com/introduction-to-word-embeddings-55734fd7068a>
- [15] [https://ballotpedia.org/Georgia\\_gubernatorial\\_election,\\_2018](https://ballotpedia.org/Georgia_gubernatorial_election,_2018)
- [16] <https://nlp.stanford.edu/projects/glove/>
- [17] [https://en.wikipedia.org/wiki/Bidirectional\\_recurrent\\_neural\\_networks](https://en.wikipedia.org/wiki/Bidirectional_recurrent_neural_networks)
- [18] <https://stackoverflow.com/questions/43035827/whats-the-difference-between-a-bidirectional-lstm-and-an-lstm>
- [19] <https://www.geeksforgeeks.org/confusion-matrix-machine-learning/>
- [20] <https://techcrunch.com/2016/11/10/social-media-did-a-better-job-at-predicting-trumps-win-than-the-polls/>
- [21] Georgia Election Results 2018 <https://www.politico.com/election-results/2018/georgia/>
- [22] Sentiment Analysis of Tweets to Gain Insights into the 2016 US Election <https://cusj.columbia.edu/wp-content/uploads/sites/15/2017/06/Sentiment-Analysis-of-Tweets-to-Gain-Insights-into-the-2016-US-Election.pdf>
- [23] Sentiment Analysis for Tweets Konstantinos Korovesis [http://www2.aueb.gr/users/ion/docs/korovesis\\_msc\\_thesis.pdf](http://www2.aueb.gr/users/ion/docs/korovesis_msc_thesis.pdf)
- [24] <http://storm.apache.org/releases/1.1.2/Tutorial.html>
- [25] <https://www.kaggle.com/ngypr/lstm-sentiment-analysis-keras>
- [26] <https://github.com/laviavigdor/twitter-sentiment-analysis>
- [27] Apache Storm: <http://storm.apache.org/>
- [28] [https://www.tutorialspoint.com/apache\\_storm/](https://www.tutorialspoint.com/apache_storm/)
- [29] <https://github.com/abdufatir/twitter-sentiment-analysis/>
- [30] <https://www.kaggle.com/lbronchal/sentiment-analysis-with-svm>
- [31] <https://web.stanford.edu/~jesszhao/files/twitterSentiment.pdf>
- [32] Twitter Sentiment Analysis System <https://arxiv.org/pdf/1807.07752.pdf>
- [33] Storm@Twitter: <https://cs.brown.edu/courses/cs227/archives/2015/papers/ss-storm.pdf>
- [34] Twitter sentiment analysis using apache storm <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.678.255&rep=rep1&type=pdf>
- [35] Twitter sentiment analysis <https://sites.google.com/site/anthonykunnelljose/home/MajorReport.pdf>
- [36] <https://github.com/ayushoriginal/Sentiment-Analysis-Twitter>
- [37] <https://nycdatascience.com/blog/student-works/web-scraping/build-near-real-time-twitter-streaming-analytical-pipeline-scratch-using-spark-aws/>
- [38] <https://www.youtube.com/watch?v=aDKxFmV6i4s>