Sarcasm Detector - text mining approach to help brands conduct business sarcasm analysis

Kartik Umalkar, Kalyani Deshmukh, Mukesh Mogal and Pranav Patil
Department of Software Engineering, San José State University
San José, CA
kartik.umalkar@sjsu.edu, kalyani.deshmukh@sjsu.edu, mukeshbhausaheb.mogal@sjsu.edu, ompranav97@gmail.com

***Abstract*— A business needs to drive product marketing, sales, and operations according to the user feedback, product market fit, and public sentiment. Companies heavily rely on existing marketing, ads, and social sentiment campaign data to tell their clients whether a sentiment behind a certain product was positive or negative. There are organizations providing sentiment analysis but there is a missing link of detecting sarcasm. In today's millennial world, where sarcasm is so prominent a need for sarcasm analysis along with the existing sentiment analysis also arises.**

We intend to provide a way for businesses to use our tool to help them understand if the media sentiment they see from other tools is actually the sentiment the users/media is expressing or if its sarcasm.

I. Introduction

A business needs to drive product marketing, sales, and operations according to the user feedback, product market fit, and public sentiment. There are organizations providing sentiment analysis but there is a missing link of sarcasm NER analysis that is based on activity in the social media landscape. In today's millennial world, where sarcasm is so prominent a need for sarcasm analysis along with the existing sentiment analysis also arises. This is an interesting connection where businesses can find hidden value due to name entity recognition analysis derived out of sarcastic sentiment around a business, trend, or product in social media.

II. Architecture

There are 5 major components of the project. Data, Data Pre-Processing, Training Data, Testing Data, and Tableau Dashboard.

Our hunt for the data set started in our exploration of social media sentiment. Since sarcasm is human driven, witty, and changing, we needed our data source to be of similar type. We found a Kaggle dataset that had headlines from The Onion and The Huffington Post and it was perfect because if we were to use Twitter to analyze business sarcasm behind products, trends, and industry then having a data set with short/medium text length (like headlines) is perfect for our need.

We found an annotated data set on Kaggle that fit our need. We used a neural network to process the annotated data set and after training the model, we would ran it against live tweets pulled from Twitter in real time around a product, trend, business, or any entity. Once the data is pulled, we take a good mix of tweets such as high favorite count, high retweet count, and some with a mix of low or medium number of retweet and favorite count. Once we take that input from Twitter and prepare our json file, we send it to the model to run through each tweet and give us a 1 or 0 count that represents 'sarcasm' or 'no-sarcasm'. Then NER (name entity recognition) is performed on top of tweets that are marked as sarcastic. This data is then visualized using a Tableau dashboard.

This setup is captured in a simple 3 page setup. The 1st page is the landing page explaining the product and its value and also offers users a way to sign up and log in via Google, Twitter, Facebook, and traditional email methods. Once inside, the user is presented with the option of going to the console and entering the trend, business, or product (any entity) they want to analyze for sarcasm and sarcastic NER. Once the user presses 'analyze', the application pulls tweets from Twitter, runs it against the model, and presents the user with a dashboard containing information such as sarcastic/non-sarcastic count, a location cloud with mapping to tweet count, and a table showing sarcastic tweets. The table also shows tweet specific NER with name, organization, and location with retweet and favorite count.

The application runs on a Python Flask server and a React application deployed on an AWS EC2 instance. The Flask server is handling the ML computation, pulling new tweets, processing them, and updating the dashboard.

III. Data Extraction and Processing

In recent years, social media sites such as Twitter have gained immense popularity and importance. These sites have evolved so much that users express their ideas and opinions uninhibitedly. Companies leverage this unique ecosystem to tap into public opinion on their products or services. Most large companies have a social media presence and a dedicated team for marketing, after-sales service, and consumer assistance through social media.These social media websites thus prove to be a good sources of data. To mine data regarding the products and services, we have made use of Twitter API. This data is in form of Twitter objects. Data processing is done on these objects to extract text and other relevant information. Standard Python libraries such as regex are used for this data pre-processing. This data in JSON format is provided as input to the machine learning model for prediction. [1]

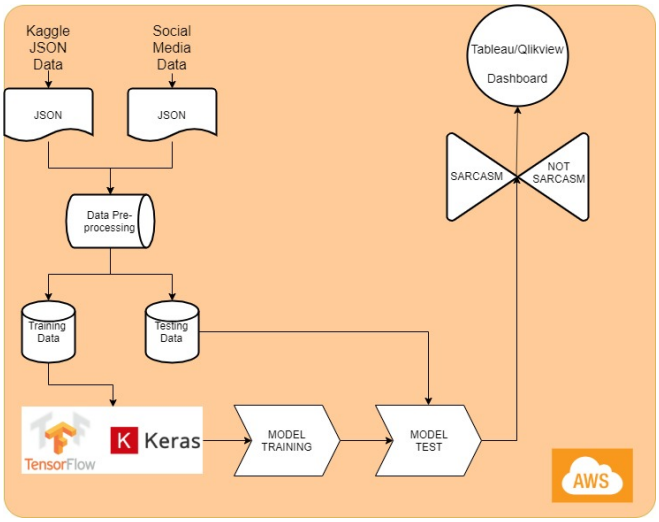

Fig. 1. Project Architecture

IV. Data Analysis

The dataset used for this project is collected from two news website. TheOnion `https://www.theonion.com` aims at producing sarcastic versions of current events and we collected all the headlines from News in Brief and News in Photos categories (which are sarcastic). We collect real (and non-sarcastic) news headlines from HuffPost `https://www.huffingtonpost.com`.Each record consists of three attributes: 1) is-sarcastic: 1 if the record is sarcastic otherwise 0 2)headline: the headline of the news article 3) article-link: link to the original news article. Useful in collecting supplementary data. The dataset is publicly accessible from the author's website [2]

A. Data Pre-processing

Below steps were done for data-pre processing 1) Getting rid of unwanted data : Removing all the unwanted characters from the data. 2) Checking up if the data is normalized or not 3) Filtering Headlines 4) Defining max features 5) Vectorize and Convert text for input 6) Splitting data to train and test : Data was randomly spilit in testing and training dataset for calculating accuracy.

B. Defining the LSTM RNN Model

LSTM-Long short-term momory is an artificial RNN-recurrent neural network used in the field of deep learning. Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. Unlike standard feed-forward neural networks, LSTM has feedback connections. It does not only process single data points, but also entire sequences of data. For example, LSTM applies to tasks such as unsegmented, connected handwriting recognition, speech recognition, and anomaly detection in network traffic or Intrusion detection systems (IDS's).

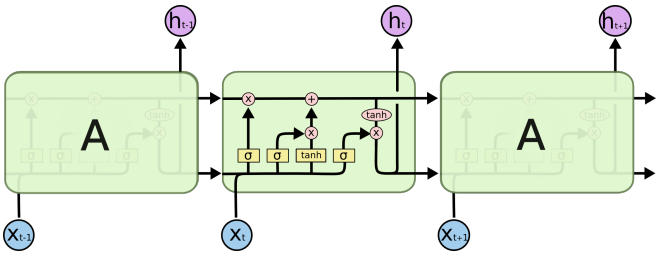

Fig. 2. Sequential processing in LSTM

C. Building model

We use Keras for developing put deep learning model. Keras is a high-level API for neural networks. It is written in Python and its biggest advantage is its ability to run on top of state-of-art deep learning libraries/frameworks such as TensorFlow, CNTK or Theano. Converting the model to dot (graph description language) format.

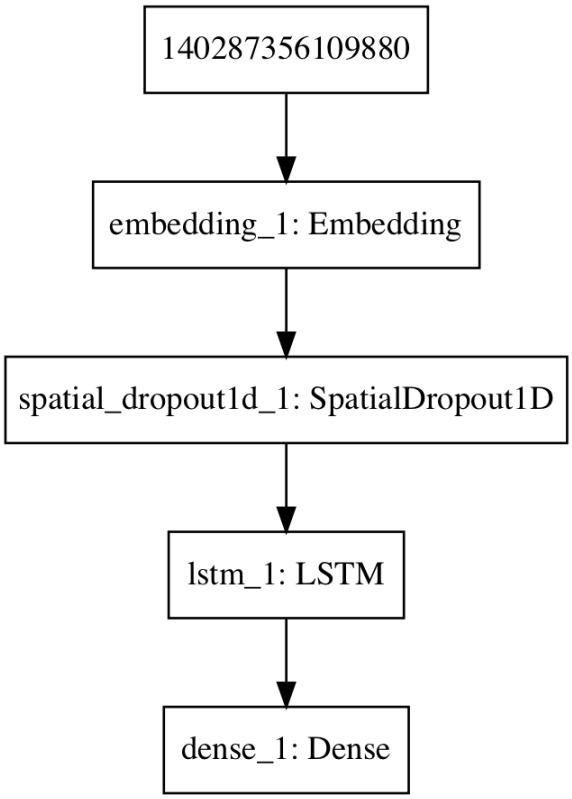

Fig. 3. model dot

D. Tuning the model

Analyzing model with graph of accuracy and Loss vs Epoch.

As you can see from above charts, ideal epoch is achieved at the value 25. After that model does not show any further improvement so we used 25 as epoch value.

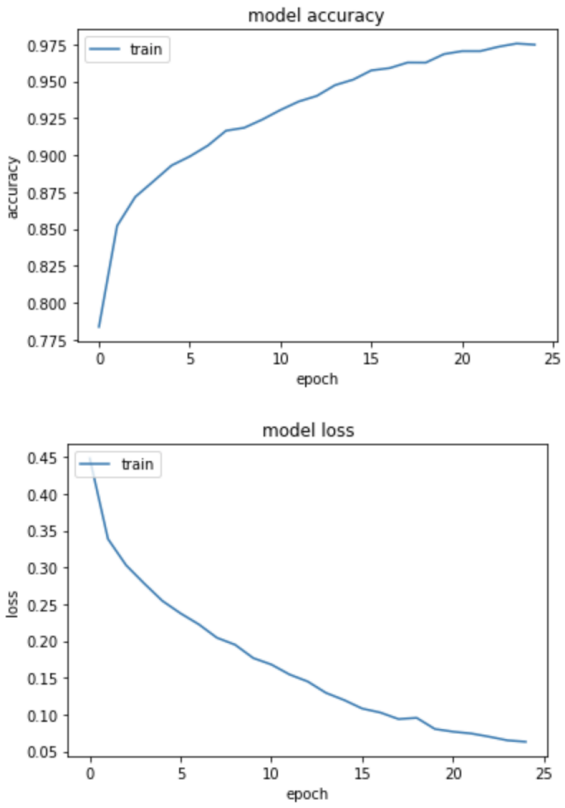

Fig. 4. Plot of Model Loss and accuracy against epoch

E. Accuracy Calculation

We calculated the accuracy of model for both Sarcastic and non-Sarcastic data set separately. Below were the results we were able to achieve.

```
Sarcasm_acc 78.28054298642535 %
Non-Sarcasm_acc 85.78255675029868 %
```

Fig. 5. Accuracy score of the model

F. Saving the model

As the trained mode is getting used for web application to show the result dashboard the model is saved using model.save() method.

V. Service

A python Web-service using Flask is developed to scan twitter for relevant tweets. The data returned from twitter API need to be processed before using them as input to the machine learning model.The processing on twitter data is done using standard python libraries such as Regex. The processed data is then provided further pre-processing before feeding it to actual machine learning model.

VI. Tableau

Application requires a live streaming data visualization in REST API. For this requirement we studied several Business Intelligence tools such as Qlikview, powerBI and all products of Tableau such as Tableau Desktop,Tableau Public, Tableau Server and Tableau Online. From this variety of products offering variety of functionalities Tableau Online is found to be the best fit. This application uses AWS RDS instance to connect Tableau Online with the data coming from the Machine Learning model implemented in Python.

Fig. 6. Sarcasm Detect Dashboard

A. Review Classification

This gives overall classification of the tweets based on result predicted by the machine learning model. In the above chart, out of total 43 tweets, 16 tweets are predicted as Sarcastic and 27 tweets are predicted as not sarcastic tweets.

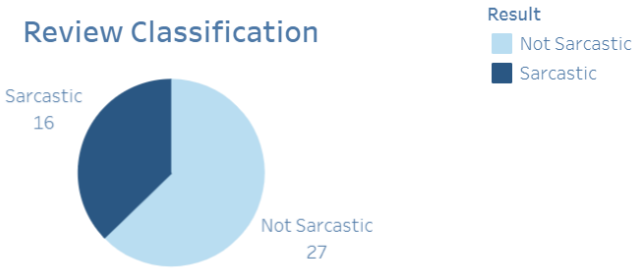

Fig. 7. Review Classification

B. Location Cloud

As the tweets data does not give the latitude and longitude information but it gives the location from which the tweet is tweeted by the user. This chart gives the locations from which the tweets are posted and the font size of the application is proportionate to number of records for the location.

Fig. 8. Location Cloud

C. Most Retweeted Sarcastic Tweets

This chart gives the details about the sarcastic tweets text, no. of retweets, no. of favorite count, and named entity recognition for organization, name and place.

Fig. 9. Most Retweeted Sarcastic Tweets

VII. Conclusion

Sarcasm Detect machine learning model gives 84 percent of accuracy in the result along with Named Entity Recognition for the entities relevant to the tweets. The Accuracy and Named Entity Recognition can be improved with the use of better Twitter API. We learned an exceptional amount of material in this project where we had a chance to build an end to end full-stack ML application with React and deploy it on AWS. It also pushed us to research into various data visualization tools and know about their strengths and weaknesses.

A. Future Improvements

1) Multiple number of social media platforms can be used as data source to add more value to the input. Once way to do this would be to apply for full access to Twitter Firehose API enterprise product and be able to pull more tweets. We would also be able to set various options and pull tweets with more variance.:

2) Other sentiment's such as anticipation, fear, trust, anger, disgust, surprise etc. analysis can also be included in the future work the application.:

3) To extend this further, annotated data set can also be maintained for further extending the scope of analysis. A possible path forward is running a script that scans 10s of various online English newspapers in various parts of the world (EMEA, Americas, Asia, and Oceania) and then annotating them for relevant sarcasm for each region.:

VIII. Deliveries

A. GitHub Repository

`https://github.com/SJSUFall2019-CMPE272/SarcasmDetect`

B. Presentation Slide

`https://drive.google.com/file/d/1c-TV8ZFSPsxeUZLcKiD81JZzqLgE31Dm/view?usp=sharing`

References

[1] Sarcasm Detection on Twitter: A Behavioral Modeling Approach. [Online]. Available: https://ashwinrajadesingan.com/files/SarcasmDetection.pdf

[2] R. Misra, "News headlines dataset for sarcasm detection," 06 2018.