SCHOOL OF COMPUTER SCIENCE

Fact.Err - Fake News Classifier

Inderpartap Cheema Najeeb Qazi
Sachin Kumar Ishan Sahay

December 7, 2019

1 INTRODUCTION

Fake news in recent times has come into importance, because of the possibility of swaying elections, public opinions and even being the main cause behind the loss of human lives. We aim to analyze this and tackle this problem, specifically the fake news in the politics category. An important goal in improving the trustworthiness of information in online social networks is to identify the fake news timely. In this project, we aim at classifying fake news from subtle yet consistent differences in the two classes of articles using Big Data Technologies.

2 PROBLEM DEFINITION

The challenges in fake news detection is a combination of many -

1. Multiple linguistic patterns
2. Numerous news categories.
3. Unconventional news vs Fake news
4. Testing on a category of news that the system has never trained upon

The goal is to combine various data sources, provide an architecture to process large amounts of data in a distributed manner and train a fake news classifier model which classifies news on a real-time basis for a specific category with a high degree of accuracy. Finally, we aim to expose the model through a web application which helps any user to check the accuracy of news, while providing visualization and trends of our data.

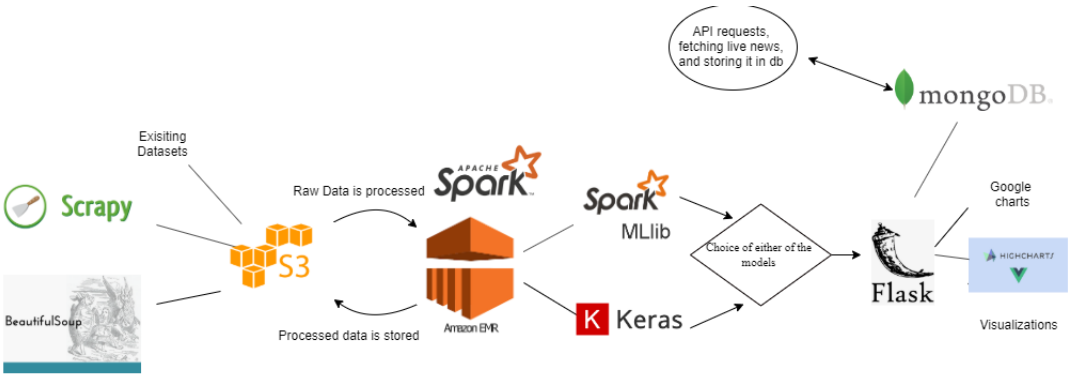

Figure 3.1: Caption

3 Methodology

1. Data Integration.
    - Built a web crawler using Scrapy and Beautifulsoup, which scraped satirical sites such as WorldnewsDaily, The Onion, Infowars, WorldTruthTv and real news web sites such as BBC, CNN only with the 'Politics tag'
    - Found existing datasets - Kaggle Fake News (212 MB), ISOT Fake dataset (110 MB), Fake News Corpus (9.1 GB)
2. Data Cleaning and Processing
    - For processing such large amount of data, $\tilde{3}$2 GB after unzipping all files. The data was uploaded onto s3 buckets making gzip partitions.
    - Read the data using Spark Dataframes on Amazon EMR (configured to an m4.Large instance with 5 nodes), and used standard NLP processing techniques for text data - Tokenization, stop words removal and stemming words to their base forms using PorterStemmer.
    - Stored the processed data as gzip files for easier reading later.
3. Modeling
    - While reading various research papers by academicians who implemented a Fake news classifier, deep learning approaches using Recurrent neural networks gave the best results. Our aim was to implement a similar model using Keras, albeit in a distributed manner. There is a distributed wrapper for Keras, called Elephas which runs distributed deep learning on Spark. We tried implementing the model, but it gave a multitude of errors while serializing: with little to zero references on solving those errors on stackoverflow and their documentation.
    - Decided to go ahead with a simple feed-forward network using Multi layer perceptron Classifier provided by Spark for the distributed model
    - Converted words into vectors using Word2Vec provided by PysparkML.
    - This model gave 80% accuracy on our validation set.
    - Along with the distributed mode, a BI-directional LSTM model with 75 hidden units and 32 hidden units in the Dense layer and , with 2 dropout layers, using Keras, on a 10% sample of the original data set was also made. The original processed data was used, with shuffling

and then retrieved 10% of the processed records. Pre-trained GloVe embeddings on Wikipedia corpus-300 dimensional, for word vectors was used. This gave better results than the Multi layer Percetpron Classifier with 88% accuracy on validation set. We decided to go ahead and deploy this model on our web application.

4. Front-end Deployment and Database
    - Configured a Flask application, with a MongoDB connection, which ran using the default MongoDB server with a single cluster composed of 3 nodes. The MongoDB retrieved news real-time (every 30 mins) from a newsAPI, and classified it using the deployed model.
    - The classified news was stored in the MongoDB, with the prediction score, text and title of the article.
    - Future scope is to re-train the model using the live news.
5. Visualization and Analysis
    - Used Google charts for visualizing trends occurrence of Fake and real news over a period of 2 years.
    - Also used Highcharts.js for wordcloud to see the differences in frequencies of words in Fake news and Real news.
    - For finding similar words that occur together, dimensionality reduction analysis: T-SNE was used. This helped to visualize the context in which similar words occur together in real and Fake news.
    - Using Latent Dirichlet allocation, Topic Modeling was done to visualize the dominant topics in fake news data

4 PROBLEMS

1. Data Processing
    - Processing large amounts of gathered data was a challenge, this was overcome by partitioning the data and uploading it on S3 bucket, and processing using Spark on EMR.
2. Distributed Deep learning
    - Distributed deep learning is still a challenge, and making a distributed deep learning model which runs on Spark is still in our future scope.
3. Model performance on different categories other than Training category
    - Focused our effort on modeling for Politics category, especially US and Canadian Politics. The model performs unsatisfactorily on categories such as Sports, medicine, etc and with a highly noisy data.

5 Results

Each phase of the implementation gave us interesting challenges and significant learning outcomes.

1. Implementation learning
    - Scraping websites and combining various datasets, handling large and messy data on Spark taught us the methodologies required to handle huge amounts of Data. Through this, we learnt to use scraping with IP spoofing, AWS EMR, and navigate various AWS instances.
2. Data Analysis
    - Since our data focused on Western Politics, there were large occurrences of 'Trump' in the data sets, almost close to double the amount in Fake news as compared to real.
    - There were large amounts of fake news related to 'terrorism' and 'attacks', confirming our intuition that some websites generate fake new with malicious intents.
    - An interesting observation was that Fake news had the large occurrence of the word 'fake' co-occurring in context with 'propaganda', 'news', 'media', 'CNN' and 'mainstream'. True news had barely any occurrences of the word 'fake news'. Fake news is more likely to purport the fact that the other 'news media outlets' are 'fake'.
    - Using document modeling on the data, our data got neatly divided into 4 topics - related to 'Trump', 'terrorism', 'military' and 'government'. Each document had a representative text for each topic. See 5.1
3. Learning outcomes
    - In the process of finding a way to work Distributed deep leaning on Spark, we came across many wrappers and learnt a lot in the process. Elephas is still the number one option, but we also came across an API developed by CERN to make Keras work on distributed environments.
    - Working with AWS, researching on papers to find the best feasible implementation for our problem was an interesting learning curve.
    - With our front-end application and live news querying, we realized the scale of Fake news and the need for architectures that are needed to handle Fake news on a large scale.

	Topic_Num	Topic_Perc_Contrib	Keywords	Representative Text
0	0.0	0.9057	government, report, official, accord, case, source, russian, story, claim, state	[charge, crook, careless, recommend, charge, already, recommendation, clinton, email, contain, c...
1	1.0	0.9932	medium, trump, news, event, time, people, political, week, show, call	[surprised, recently, discover, display, fake, time, magazine, feature, cover, star, fake, magaz...
2	2.0	0.8711	terrorist, police, shoot, attack, kill, group, man, protest, terror, incident	[video, second, load, black, box, loading]
3	3.0	0.9040	state, year, country, military, policy, new, crisis, public, american, world	[bernie, fan, look, berniesander, leave, behind, much, cost, pay, much, cost, plan, estimate, co...

Figure 5.1: Screenshot showing Four Documents, their keywords and the words most Representative of it

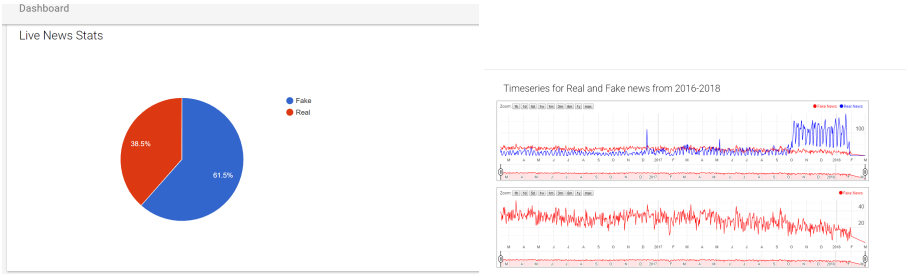

Figure 5.2: Pie chart showing live percentage of classified news in the MongoDB. Trends analysis of type of news over 2 years

Figure 5.3: Wordcloud showing Occurrences for words in Real and Fake news

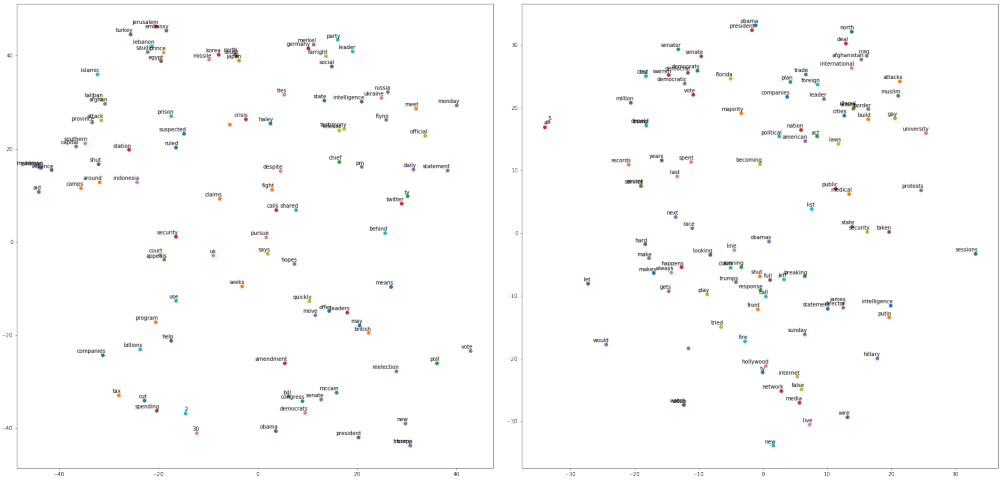

Figure 5.4: Similar occurring words in Fake and Real news analysed using T-SNE

$$Precision = \frac{TP}{TP+FP} = \quad 0.611$$

$$Recall = \frac{TP}{TP+FN} = \quad 0.69601$$

	(Actual label)	
(Predicted Labels)	Positive (Fake)	Negative (Real)
Positive	27996	17773
Negative	12227	42004

Table 5.1: Confusion Matrix for the Deep learning Model

6 Project Summary

Categories	Points	Description
Getting the data	2	Scraped and Combined existing Datasets
ETL	3	Did a lot of pre-processing to clean and process noisy data on Spark
Problem	1	Our Problem was well defined
Algorithmic Work	4	Researched on various techniques, tried Deep learning on Spark
Bigness and Parallelization	3	Can process and handle large datasets
UI	3	Flask Application with a MongoDB connection
Visualization	2	Used Google Charts, Highcharts, Explored Topic Modeling, Dimensionality reduction techniques for word clustering
Technologies	2	Explored AWS EMR, MongoDB, Keras and Elephas
Total	20	

Table 6.1: Project Summary

7 References

7.1 Datasets

- Scraped from InfoWars, worldtruthTV, WorldNewsdaily, BBC, CNN
- ISOT - Dataset by University of Victoria
- Fake New Corpus - open source dataset composed of millions of news articles mostly scraped from a curated list of 1001 domains from http://www.opensources.co/.
- Kaggle-Fake news, Getting Real about Fake News

7.2 Papers

- FAKEDETECTOR: Effective Fake News Detection with Deep Diffusive Neural Network
- 3HAN: A Deep Neural Network for Fake News Detection
- Fake News Detection Using A Deep Neural Network

7.3 Other Links

- CERN distributed deep learning
- Distributed Deep learning with Keras & Spark