What's the Tea: Real-time Topic Tagging of News Articles
Group 3 - CS543

Advith Chegu
Rutgers University
New Brunswick, NJ, USA
ac1771@scarletmail.rutgers.edu

Vipul Gharde
Rutgers University
New Brunswick, NJ, USA
vig4@scarletmail.rutgers.edu

Diksha Wuthoo
Rutgers University
New Brunswick, NJ, USA
dw659@scarletmail.rutgers.edu

***Abstract*—We have implemented Topic Tagging for news articles to classify different articles into different topics in real-time. We have used the K-means clustering algorithm to partition a curated dataset of over 40 million news articles into 'k' clusters. Using this clustering algorithm, we trained our model to group different news articles, and then apply this model to real-time tweets from various authorized Twitter news handles to predict the topics at any given time.**

***Index Terms*—News, topics, clustering, tagging, Twitter, word2vec, Natural Language Processing (NLP)**

I. Introduction

With the rise in the volume of real-time information, consumers are inundated with countless stories and news articles that make it difficult for them to sift through and search for their topics of interest. We wanted to provide a solution that will help busy consumers stay on top of current events without much effort or many clicks to do so. With this motivation in mind, we wanted to make an application that would allow curious users to know what topics are the most trending in the world and then find articles related to those topics. Our application provides users with real-time trending topics such as Politics, Finance, Sports, etc., helping them to view relevant news from the topic of their interest. We fetch real-time tweets from authorized and reputable news handles on Twitter and use those tweets to show the user what topics are the most trending.

II. Dataset Description

We are using multiple datasets for this project:

RealNews Dataset [1]: This dataset (taken from the paper [2]) is a single JSON file with the fields *article title*, *article text*, *summary*, *authors*, *published date*, *URL*, and *domain*, but we don't need to use all those fields for the clustering model. Since it is a huge dataset (over 100GB in size), we have used the *summary* field from the dataset for our implementation. The resulting dataset totals to around 35 million news articles.

News Aggregator Dataset [3]: This dataset from Kaggle is a CSV file with the fields *ID*, *TITLE*, *URL*, *PUBLISHER*, *CATEGORY*, *STORY*, *HOSTNAME* and *TIMESTAMP*, totaling to over 100MB and over 400k news articles. We have used the *TITLE* field.

A Million News Headlines [4]: This dataset from Kaggle is a CSV file with the fields *publish date* and *headline text*, totaling to over 60MB and over 1.2 million news articles. We have used the *headline text* field.

All the News 2.0 [5]: This dataset is a CSV file with the fields *date*, *year*, *month*, *day*, *author*, *title*, *article*, *url*, *section*, and *publication*, totaling to over 60MB and over 2.6 million news articles. We have used the *title* field.

India News Headlines Dataset [6]: This dataset from Kaggle is a CSV file with the fields *publish date*, *headline category* and *headline text*, totaling to over 250MB and over 3.6 million news articles. We have used the *headline text* field.

The combined dataset total to over 40 million news records.

III. Questions About the Data

With this dataset, we answer the question of what is being discussed in real-time. For example, if a user wants to see the most trending topics at a given moment, they can use our model to see what's popular, which will show the top 5 recently tagged topics. Here, 'popular' is defined as the topic clusters having the most number of counts of recent news articles. The user can then choose any one of those topics and view news articles from that particular topic. This will also save users a lot of time to know about the current events of the world.

IV. Implementation

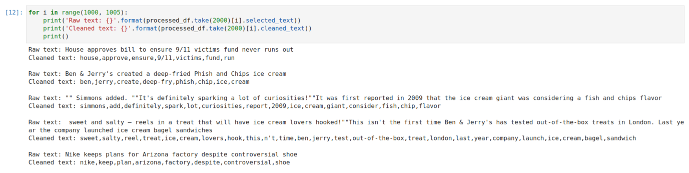
A. Data Preprocessing

For the RealNews dataset, we are mainly using the *summary* field for our clustering model. We found that some articles don't have a summary, so for those records we have used the *title* field from the dataset instead.

For the other datasets, we have used the specific columns as mentioned in the earlier section for preprocessing.

We pass these datasets for preprocessing which includes the removal of stopwords, lemmatization, etc. We start with tokenizing the summary for each record using the *nltk* library which gives us an array of words. After tokenization, we convert all the words to lowercase and remove the stopwords like 'a', 'an', 'the', which don't contribute to the semantics of the sentence. For the removal of those words, we have used the already provided set of *STOPWORDS* in the *gensim* library. We also remove punctuation marks and any words that are of length 2 or less. After that, we pass the whole dataset for lemmatization, for which we have used the *WordNetLemmatizer* of the *nltk* library. Lemmatization is a text normalization technique commonly used in Natural Language Processing that helps us to group different forms of the same word and transform them to root form while preserving the semantics. This is the final step of preprocessing the dataset.

We then store the resulting dataset into JSON files.



```
[12]: for i in range(1000, 1005):
          print('Raw text: {}'.format(processed_df.take(2000)[i].selected_text))
          print('Cleaned text: {}'.format(processed_df.take(2000)[i].cleaned_text))
          print()

      Raw text: House approves bill to ensure 9/11 victims fund never runs out
      Cleaned text: house,approve,ensure,9/11,victims,fund,run

      Raw text: Ben & Jerry's created a deep-fried Phish and Chips ice cream
      Cleaned text: ben,jerry,create,deep-fry,phish,chip,ice,cream

      Raw text: "" Simmons added. ""It's definitely sparking a lot of curiosities!""It was first reported in 2009 that the ice cream giant was considering a fish and chips flavor
      Cleaned text: simmons,add,definitely,spark,lot,curiosities,report,2009,ice,cream,giant,consider,fish,chip,flavor

      Raw text:  sweet and salty – reels in a treat that will have ice cream lovers hooked!""This isn't the first time Ben & Jerry's has tested out-of-the-box treats in London. Last ye
      ar the company launched ice cream bagel sandwiches
      Cleaned text: sweet,salty,reel,treat,ice,cream,lovers,hook,this,n't,time,ben,jerry,test,out-of-the-box,treat,london,last,year,company,launch,ice,cream,bagel,sandwich

      Raw text: Nike keeps plans for Arizona factory despite controversial shoe
      Cleaned text: nike,keep,plan,arizona,factory,despite,controversial,shoe
```

Fig. 1. Difference between the raw data and the cleaned data

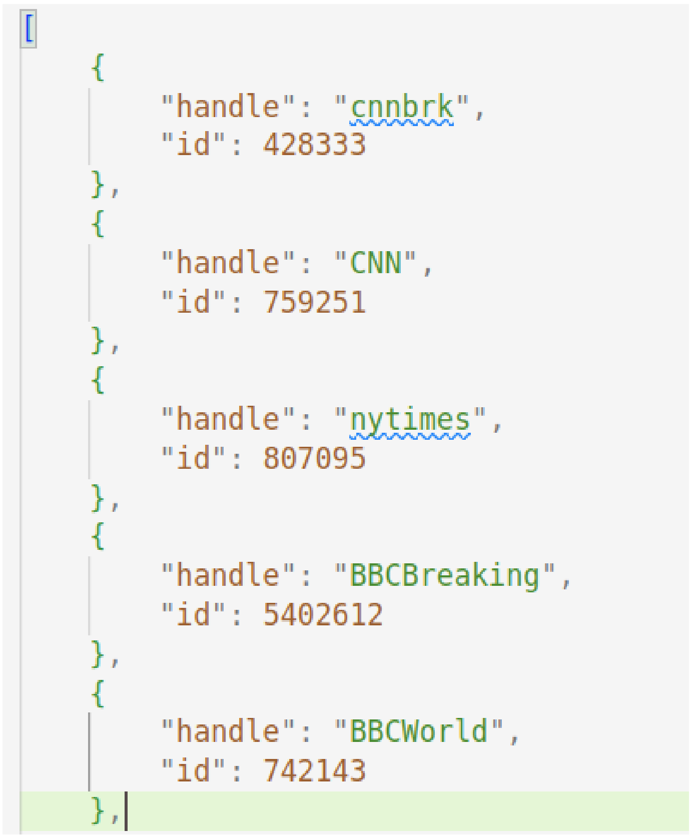
B. Model Building

For creating the k-means clustering model, we first have to convert all the data into a vector format, which is currently in textual form. For the transformation, we have used the *word2vec* library to generate the embeddings. *Word2vec* is a neural network model that can learn meaningful relations and then encode the relatedness into vector similarity. After generating the *word2vec* model, we then apply the transform method on the training set which gives us the data in vector format. After getting the data in the correct format, we pass it to the K-means clustering model to build it and run the model for a maximum of 10 iterations. After this step, we get the cluster centers from the model, which we have labeled into different categories/topics, and then save the model.

C. Model Prediction

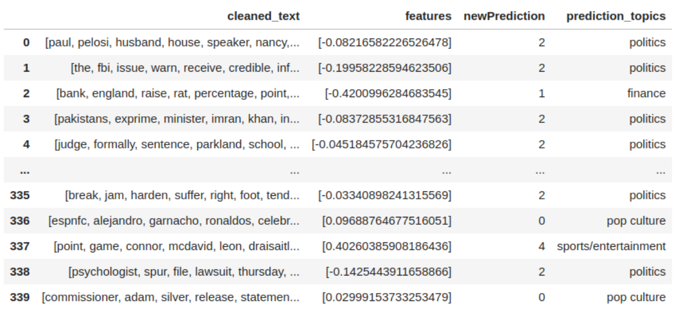
Now as our K-means model is created, we have to use this model to predict real-time data and classify them into specific topics. We have selected 34 authorized and reliable Twitter news accounts to fetch the latest 10 tweets from each of those accounts. For retrieving the tweets, we have used the Tweepy package which provides APIs to fetch the data.

After retrieving the tweets, we then do the preprocessing steps as we did before for the entire dataset which includes tokenization, removal of stopwords, punctuation, and lemmatization. The raw data from the tweets also include some links at the end of each tweet which we remove using the regex filter. Now that we have the cleaned words, we then use the same *word2vec* model (created in the model building step) to get the embeddings for test data so that we can pass it to the clustering model for prediction. After getting the data in vector form, we pass it to the model to predict and classify the tweets into different clusters.



```
[
    {
        "handle": "cnnbrk",
        "id": 428333
    },
    {
        "handle": "CNN",
        "id": 759251
    },
    {
        "handle": "nytimes",
        "id": 807095
    },
    {
        "handle": "BBCBreaking",
        "id": 5402612
    },
    {
        "handle": "BBCWorld",
        "id": 742143
    },
```

Fig. 2. JSON of Twitter news accounts and their Twitter IDs



	cleaned_text	features	newPrediction	prediction_topics
0	[paul, pelosi, husband, house, speaker, nancy,...	[-0.08216582226526478]	2	politics
1	[the, fbi, issue, warn, receive, credible, inf...	[-0.19958228594623506]	2	politics
2	[bank, england, raise, rat, percentage, point,...	[-0.4200996284683545]	1	finance
3	[pakistans, exprime, minister, imran, khan, in...	[-0.08372855316847563]	2	politics
4	[judge, formally, sentence, parkland, school, ...	[-0.045184575704236826]	2	politics
...	...	...	...	...
335	[break, jam, harden, suffer, right, foot, tend...	[-0.03340898241315569]	2	politics
336	[espnfc, alejandro, garnacho, ronaldos, celebr...	[0.09688764677516051]	0	pop culture
337	[point, game, connor, mcdavid, leon, draisaitl...	[0.40260385908186436]	4	sports/entertainment
338	[psychologist, spur, file, lawsuit, thursday, ...	[-0.1425443911658866]	2	politics
339	[commissioner, adam, silver, release, statemen...	[0.02999153733253479]	0	pop culture

Fig. 3. Predicted clusters of the test data

D. Tools and Technologies

We are using *PySpark* which supports numerous libraries and packages related to Natural Language Processing (NLP) for tokenization, lemmatization, and stop words removal to format our training and testing dataset. We have used *wordnet*, *nltk* for all the NLP-related techniques. We also used *Word2vec* which is a text-to-vector converting library that helps us to represent text as an n-dimensional vector. Our K-means clustering algorithm is also provided in a built-in ML package in *PySpark*. We used the PySpark ML package as it provides a distributed way to compute Word2vec embeddings making the task run faster compared to a single machine. [7] [8] [9] [10]

E. Viszualization

We took latest 340 tweets and passed it to the clustering model which predicted these categories to be the most trending topics on Twitter.

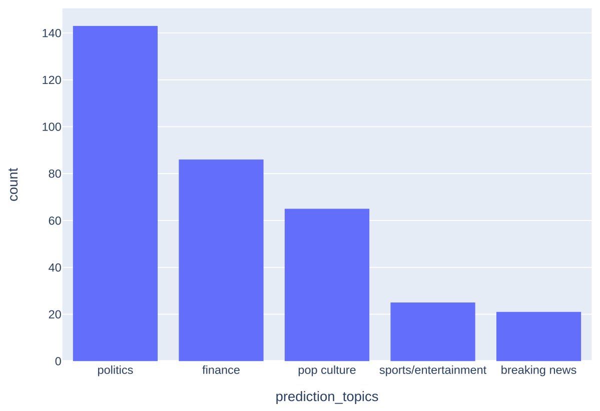


Fig. 4. Chart showing the top 5 trending topics

V. Conclusion and Future Work

In this project, we implemented an application to cluster related short texts together into labeled topics. We do the clustering by first using *word2vec* to create a singleton vector for each document and then using k-means to find the clusters. We label the clusters based on the training data, and then use the Twitter API to grab the latest 340 tweets and do the same preprocessing on the tweets. Instead of fitting the k-means model to the tweets, we predict which cluster each tweet belongs to.

Some setback we faced was the reduced dimensionality of our document vectors. The method that we used resulted in singleton vector values for each document which severely limits the efficiency of the k-means clustering algorithm that generates the topics. Due to this, certain categories were collapsing against each other and thus, causing a few overlapping results. We plan to fix this issue by using a more robust vectorization technique involving deep learning such as a Feed Forward Neural Net Language Model or a Recurrent Neural Net Language Model. Because *PySpark* does not have a native implementation of these algorithms unlike *word2vec*, we will have to find a creative solution to implement these while keeping our dataset moderately large to maintain accuracy.

We would also like to deploy our API onto a cloud service provider such as AWS and continually update our model using streaming k-means. This way the accuracy remains high and we stay up to date with the latest news. By deploying our model, it can also be used widely by anyone on the internet enabling people to call our API for any creative use.

References

[1] RealNews Dataset — Papers With Code. [Online]. Available: https://paperswithcode.com/dataset/realnews

[2] Defending Against Neural Fake News. [Online]. Available: https://arxiv.org/pdf/1905.12616v3.pdf

[3] News Aggregator Dataset — Kaggle. [Online]. Available: https://www.kaggle.com/datasets/uciml/news-aggregator-dataset

[4] A Million News Headlines — Kaggle. [Online]. Available: https://www.kaggle.com/datasets/therohk/million-headlines

[5] All the News 2.0 — 2.7 million news articles and essays from 27 American publications - Components. [Online]. Available: https://components.one/datasets/all-the-news-2-news-articles-dataset/

[6] India News Headlines Dataset — Kaggle. [Online]. Available: https://www.kaggle.com/datasets/therohk/india-headlines-news-dataset

[7] Text Clustering using Python and Spark — by Davide Gazzè — Medium. [Online]. Available: https://medium.com/@davide.gazze/text-clustering-using-python-and-spark-69a3696a4320

[8] Creating Word2Vec embeddings on a large text corpus with pyspark — by Abhishek Singh — Medium. [Online]. Available: https://medium.com/@the.data.yoga/creating-word2vec-embeddings-on-a-large-text-corpus-with-pyspark-469007116551

[9] K-Means clustering with Apache Spark — by .eranga — Medium. [Online]. Available: https://medium.com/rahasak/k-means-clustering-with-apache-spark-cab44aef0a16

[10] Understanding Word2Vec With PySpark - Gabriel Fair. [Online]. Available: https://gabefair.github.io/posts/2018/12/Understanding-Word2Vec-With-Pyspark/