International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-13 Issue-1, December 2023

Enhancing Twitter Tweet Topic Understanding through Ensemble Learning

Suraj Shinde Patil, Muskan Jain, Harsh Ahuja, Akshay Mathur

Abstract. Since Twitter introduced the hashtag as a content grouping label on social media in 2007, the symbol and its associated usage as a classification label have seen widespread adoption throughout social media and other platforms. While the content of a post can be conveniently classified using said post's hashtags, classifying posts that do not contain hashtags proves to be a much more challenging problem. In this paper, we propose a system for identifying a post's hashtags using only the non-hashtag terms of the post and, by extension, address the issue of classifying the contents of posts that do not contain hashtags.

Keywords: Social media, Classification, hashtags, Twitter data

I. INTRODUCTION

With an overarching goal of making sense of the massive amount of Twitter data in the form of Tweets, our project focuses on characterising Tweets by their associated specific topics, ignoring the occurrences of hashtags. In addition to using standalone learning models, such as a bag-of-words model, neural networks on the raw frequencies, or similarity measures, we create an ensemble of features from a Tweet. By utilising information provided by other information extraction methods, such as sentiment analysis, our learning models will potentially yield better results and handle more robust instances of Tweets. This project has the additional utility of being integrated into other supervised learning models that require further dimensions and information about raw Tweets. The generated mapping from Tweet content to hashtags could also serve as the basis for a hashtag recommendation system for Twitter.

II. PREVIOUS WORK

We first surveyed recent literature and research analysing Twitter data to gain an understanding of the work that has been done in the field. Sections 2.1 to 2.3 provide a brief overview of 3 papers that significantly influenced and motivated our project.

Manuscript received on 06 November 2023 | Revised Manuscript received on 16 November 2023 | Manuscript Accepted on 15 December 2023 | Manuscript published on 30 December 2023.

** Correspondence Author (s)*
Suraj Shinde Patil, Department of Computer Science, Sanjay Ghodawat University, Kolhapur (Maharashtra), India. E-mail: surajshindepatil20@gmail.com, ORCID ID: 0009-0005-2486-0544
Muskan Jain, Department of Computer Science, Dr. A.P.J. Abdul Kalam Technological University, Lucknow (U.P.), India. E-mail: muskaanjain.2608@gmail.com, ORCID ID: 0009-0004-8657-9929
Harsh Ahuja, Department of Computer Science, Dr. A.P.J. Abdul Kalam Technological University, Lucknow (U.P.), India. E-mail: harshahuja1179@gmail.com, ORCID ID: 0009-0006-4810-4193
Akshay Mathur*, Department of Computer Science, Manipal University Jaipur, Jaipur (Rajasthan), India. E-mail: akshaym3119@gmail.com, ORCID ID: 0009-0000-7020-9385

Section 2.4 outlines the new ideas and goals behind our work.

A. Lee, Palsetia, Narayanan, Patwary, Agarwal, Choudhary

The basis for Lee et al. and its motivation is related to Twitter's Trending Topics product, which appears on the stream homepage for users. Most of the time, it is challenging to classify or understand what these topics are truly about, so the authors set out to categorise them into 18 distinct general categories. Lee et al. [1] used two different approaches to solve this problem, including the Bag-of-Words model and a network- based classification. To come up with the labels initially used for training, they used up to 3 human annotators, with the third being used if the first two did not agree on a category. Focusing on the Bag-of-Words model, the paper constructed word vectors from trending topic definitions and related Tweets, and used tf-idf weighting with a multinomial Naive Bayes classifier to classify the topics that appeared. We plan on using some of these same techniques to implement the topic classifier.

B. Ramage, Dumais, Liebling

Ramage et al. [2] also discuss two different features that Twitter and its users would find helpful: the process of user and figure discovery, as well as feed filtering based on one's interests. Their work involves using a partially supervised learning model, Labelled LDA, to process the multi-labelled fields in Twitter data and separate them into distinct dimensions. Labelling each document, or Tweet in this case, by a specific set of topics helps characterise the reading behaviours of its users. The paper considers four main behaviour classes: substance, status, style, and social context, with data that does not fall into one of these categories being separated. Some interesting observations were that specific Twitter features, such as mentions, hashtagging, emoticons, replies, retweets, and favourites, were all related to particular behavioural characteristics. Our project would help augment this approach by developing a method to label Twitter stream data and potentially contribute to their conclusions in our topic classification.

C. Li, Ritter, Cardie, Hovy

The process of extracting events and classifying them based on Tweets by Li et al. [3] can be broken down into three separate areas of work: user-level analysis, identification and extraction of public events, and learning from these areas to acquire more data. The paper proposes a pipeline that combines these different areas to classify a user's Tweets into specific life events. Their process first takes a noisy input stream of user Tweets and attempts to categorise them into different categories of significant life events (many are not visible or captured).

To better recognize when a significant life event is captured in a Tweet, they leverage Tweet replies to determine when congratulations or condolences were exchanged. Next, Li et al. took these different categories and trained a classifier that would use words in the Tweet, named entity tags. The top words associated with each pre-determined category were used to determine which life event (if any) it belonged to. Different signals, such as the sentiment of a Tweet as described by Li et al., can prove to be very useful, a strategy we plan to employ rather than using the raw text alone for training.

D. Motivation and Goal for our Work

Among the large number of papers we reviewed, we found a consistent trend in which each group applied different approaches to the same problem. Most papers begin with an initial attempt to solve the problem using a language model based purely on information retrieval principles, such as n-gram likelihoods, Bag-of-Words representations, or similarity measures. These often concluded that many aspects or features in the data were largely unused or could not be feasibly used. The subsequent attempts to make sense of the data involved supervised learning in a neural network or uncovering relationships between features using methods such as SVM. Some papers recognise the benefits of using each of these techniques and create ensembles of different signals and successful strategies to enhance their accuracy. The goal of our project is to develop a straightforward tweet classification system that predicts the topic of a tweet based on its text content. We aim to model Twitter data using various natural language understanding (NLU) techniques similar to those covered above and methods discussed in CS 224U. One of the key takeaways from previous work is that ensemble systems typically yield improved performance compared to standalone models and learning algorithms. Previous ensemble algorithms, however, appear to utilise only models specific to a single general area of NLU, such as focusing on word representations or semantic parsing. Our system utilises models based on various regions of NLU, and in Section 5.2, we implement a new, data-driven ensemble algorithm designed to achieve optimal performance.

III. DATA COLLECTION AND PREPROCESSING

We begin by discussing our data collection in Section 3.1, and in Section 3.2, we describe our methodology for cleaning and preparing the dataset of Tweets.

A. Data Collection

Twitter contains a wealth of data on a highly diverse set of topics, and it has often been used as a primary source of data for natural language understanding and machine learn- ing research. Twitter, as a company, has also enabled open access to its data through company-supported APIs. As such, Twitter data is relatively easy to find, and many sources of high-quality Twitter data have been made available online. Our dataset, provided by the Stanford Linguistic Data Consortium (SLDC), comprises approximately 8 gigabytes of documents, including tweet content and metadata. The data is divided into 27 groups based on the presence of specific hashtags.[5] The overall choice of groups provides a nice amount of variation in content. For our system, we used the identifying hashtag as the topic label.

Given the scope of our project and the available hardware, we chose to primarily examine a smaller portion of the data, comprising 6,000 Tweets from each of the 27 topic groups. This yields a total of 162,000 Tweets. This smaller subset of the raw data allowed us to work more flexibly and efficiently. To verify the appropriateness of 5, the complete list of groups (with the # symbol removed) is: android, basic, coffee, dontjudgeme, earthquake, egypt, election, freedom, god, haiti, happy, harrypotter, healthcare, immigration, indonesia, ipod, love, mubarak, obama, obamacare, question, sotu, teaparty, tsunami, usa, win, wiunion. Our partition size was initially generated from some of our intermediate results on larger subsets of the entire dataset. The results were sufficiently close to indicate that our primary working set was large enough to capture the bulk of language-based variance among the Tweets. From this point on in our paper, any mention of the "dataset" is referencing this subset.

B. Tweet Tokenization

Users generate the vast majority of content on Twitter, and oftentimes, Tweets do not follow formal language conventions. As a result, it is a common occurrence for Tweets to contain words or tokens that are generally not considered to be standard forms of language. Developing a system to accurately parse and extract informative tokens from "noisy" Tweets thus becomes integral to extracting a suitable language corpus. It can have a significant impact on the performance of language-based analysis of Twitter data. Our system for tokenizing Tweets draws heavily from the Twitter preprocessing methods by Pennington et al. in [4]. Several non-standard language tokens commonly found in tweets are filtered or modified during the tokenisation process. The first of these is the hashtags themselves; formally, these are tokens that contain a phrase that is preceded by a "#" symbol, such as "#win". For our classification system, hashtags are removed from the Tweet body during preprocessing. This is done because our topic groups are divided based on the presence of specific hashtags, as described in Section 3.1. The string "RT" also appears in many Tweets due to retweets. These are also removed because they provide no information. Our system modifies emotion faces (emoticons) such as ":)" and encodes them with unique identifier tags. Furthermore, words typed out using only capital letters, such as "NEWS", and words spelt using unnecessarily repeated instances of letters, such as "heyyyyy", are labelled with similar tags. Finally, all numerical values are replaced with a numerical value tag. However, URLs and references to other Twitter users (handles, denoted by the @ symbol) are left as they appear in the original Tweet. Finally, standard punctuation symbols, such as periods and exclamation points, are removed from the Tweet context. Following the initial processing of the Tweet body, a list of tokens is obtained by simply splitting the Tweet on whitespace. The flexibility of this tokenisation method enables the extraction of a high level of context information, even from Tweets containing a large amount of unorthodox language.

For example, consider the Tweet.

"on 1 Fav Source+5 others like CNET News-Why Google Android is winning http://bit.ly/aW9QWWn yayyyy :')!!!!!!!!"

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Processing this Tweet returns the tokens:

[' on', '<number>', 'fav', 'source<number>', 'others', 'like', 'cnet<allcaps>', 'news-why', 'google', 'android', 'is', 'winning', 'http://bit.ly/aW9QWWn', 'yay<elong>', '<smile>']

Of particular note, the "yayyyy" token is encoded in a manner that allows it to match with any instance of the word "yay" followed by any number of unnecessary "y"s. In In addition, all the capital letter formatting of the word "CNET' and the smiling face near the end of the Tweet body are captured in corresponding tokens.

IV. LEARNING MODELS AND FEATURES

We now discuss the technical details of the models and features used in implementing our system. Our project was primarily coded in Python, utilising the popular open-source scikit-learn module for assistance. In Section 4.1, we discuss the construction of our Word-Tweet matrix, a data structure analogous to a general word-document matrix. In Sections 4.2 to 4.6, we discuss our various approaches to modelling tweets in our dataset.

A. Word-Tweet Matrix

The first step in our Tweet analysis procedure is the construction of a Word-Tweet matrix. To construct such a matrix, the Tweet tokenization process described in Section 3.2 was first applied to all Tweets in the dataset. Then, we counted the number of times each unique word token appeared in our dataset. We removed all tokens that did not appear at least 50 times, as rare words are unlikely to contribute to the generation of accurate models. The remaining frequently occurring tokens are then placed into a term set, and each token is assigned a unique ID. A term frequency vector t of a Tweet is then defined as a vector containing a feature for each of the terms or tokens in the frequently occurring term set. The ith element of the vector, t_i, is equivalent to the number of times the term with id i appears in the Tweet. The Word-Tweet matrix is then constructed such that the term frequency vector of each Tweet in the dataset is stored as a column in the matrix. This is achieved through a second pass over the dataset, where the term frequency vectors of each Tweet are calculated and stored. Any Tweets that have over 50% of their tokens not appearing in the generated term set are not placed into the Word-Tweet matrix, as these Tweets have had a significant amount of their content stripped and thus cannot be classified meaningfully. The Word-Tweet matrix is a beneficial data structure that provides the basis for our various models, such as raw term frequency vectors and term frequency-inverse document frequency (tf-idf) vectors.

B. Raw Term Frequency Vectors

The first classification component we used involved constructing several simple classifiers that directly classified on a bag-of-words language model, where the feature vector of each Tweet simply contained the counts of the tokens appearing in the Tweet. Term frequency vectors for each Tweet were obtained by normalizing the columns of the Word-Tweet matrix generated using Section 4.1. These frequency vectors were then mapped to their corresponding hashtags and used as data points for training our models' classification. Using the raw term frequency vector model as a backbone, we focused on three classification algorithms: logistic regression, k-nearest neighbours, and a 3-layer neural network.

C. Term Frequency-Inverse Document Frequency Vectors

Another possible method for accurately classifying Tweet hashtags involves first constructing term frequency-inverse document frequency (tf-idf)- based word vectors that encode information about the similarities and differences between different tokens in the corpus. In the TF-IDF word vector model, the IDF of a token t is given by the formula:

$$\text{idf}(t) = \log\left(\frac{\#\text{ of total documents}}{\#\text{ of documents containing } t}\right)$$

The tf-idf value for a token t and a Tweet d is then defined as:

$$\text{tf-idf}(t, d) = \text{tf}(t, d) \times \text{idf}(t)$$

Here, $\text{tf}(t, d)$ represents the relative number of occurrences of t in the Tweet d concerning all other Tweets in the dataset. The tf-idf word vector for the token t is then defined as the vector containing the tf-idf scores of t with every Tweet in the corpus. Using the Word-Tweet matrix constructed in Section 4.1, these word vectors can be computed straightforwardly. Following the generation of the tf-idf word vectors, Tweet representations can be constructed as a combination of the word vectors for the tokens in the Tweet. In our model, the mean of the word vectors for tokens contained in the Tweet is used as a vector representation of the Tweet.

D. Feature Vectors Derived from GloVe Word Representations

Another approach we tested was building feature vectors for each Tweet by leveraging GloVe word representations referenced in [4]. Our implementation was based on a similar one provided in the CS 224U distributed Word Reps code lab. Using the frequent tokens (or words) identified during our construction of the Word-Tweet matrix, we constructed a Word-Word matrix using all of the Tweets in our dataset. Following the generation of the GloVe vectors, we attempted various methods for representing Tweets using these vectors. Using the Word-Tweet matrix, we extracted the significant words and their frequencies from each Tweet. Due to our earlier filter on which Tweets we would consider, we know that each Tweet has many words with corresponding GloVe vectors. After extensive testing, it was found that the most straightforward approach yielded the best results; the feature vector for a Tweet was created by simply summing the GloVe vectors for each distinct word that appeared. This approach worked marginally better than other related methods, such as weighting the GloVe vectors by the corresponding word frequency and averaging the sum of the GloVe vectors.

E. Sentiment Learning

One signal that involves information extraction and additional training on a separate Tweet dataset is sentiment. The predicted sentiment of a particular Tweet could be beneficial for identifying the class of specific topics in our dataset by providing a signal that could have its weight in the classification determined by each learning model. For example, the "win" topic should have a robust correlation with positive sentiment, and the "tsunami" topic should generally be perceived as very harmful.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

To extract this information for input when operating on our test dataset, we trained a sentiment scorer on the Sentiment140 Twitter dataset to classify the sentiment of our topic dataset. This was achieved by creating a tf-idf feature vector, as before, on the training data using the sentiment score labels as the target. The learned model was then used to create a pipelined process that generates a sentiment label during both training and testing on the topic dataset. The initial training yielded a test accuracy of 0.712 using logistic regression and up to 0.739 when employing a neural network, which was sufficiently high for us to incorporate into our topic classification model as a training signal. This semi-supervised strategy might incorrectly label the sentiment of specific Tweets. Still, we hoped that for the much larger proportion of Tweets, the labelling would be accurate and create a valuable signal for our model to learn from. In this model, we used this signal as an additional feature, concatenated with the TF-IDF vectors.

F. Open IE Relation Extraction

Open Information Extraction (OpenIE) System is an information extraction open-source tool created by the University of Washington that takes a sentence and breaks it up into relational clauses [5]. We saw this as a potentially helpful addition to our learning models, which would help differentiate between the importance of specific words in the Tweet or allow us to exploit the relationship between words. Each *n*-ary extraction will emphasize certain words or phrases within the Tweet, allowing us to develop an intuitive weighting scheme that promotes words in our respective word vector representations. For example, an analysis of the text "The U.S. president Barack Obama gave his speech on Tuesday to thousands of people." will yield the relation (Barack Obama is the president of the US) among others. This allows us to use this information and potentially weigh the subject and relational terms more than the rest of the phrase or Tweet. Rather than creating a new feature for each of these different relations, we weighed the word representations instead based on the most likely relation outputted by OpenIE. This decision helps eliminate the problem of having extremely sparse training and test data in our feature vectors. Additionally, this allows for human intuition to play a role in determining what relations – and therefore words – could potentially be significant. In this model, these weights are incorporated into the TF-IDF vectors.

V. LEARNING ALGORITHMS AND RESULTS

We now discuss the results of training various learning algorithms on our different models. Individual model results and analysis are given in Section 5.1, while Section 5.2 discusses our overall ensemble system.

A. Individual Model Results

As discussed in Section 4, we modelled our data set using a variety of different approaches: raw term frequency vectors (which we will refer to as FREQ), raw tf-idf vec- tors (TFIDF), feature vectors derived from GloVe word representations (GLOVE), tf-idf vectors with an additional sentiment scoring feature (SENT), and tf-idf vectors based off of relation extraction weighting (REL).

Table 1. Classification Accuracy for Individual Models

Model	FREQ	TFIDF	GLOVE	SENT	REL
Logistic Regression	0.58507	0.61264	0.48970	0.56820	0.58764
3-Layer Neural Net.	0.58913	0.60989	0.48333	0.58214	0.59063
*k*NN - Minkowski	0.53517	0.59128	0.44745	0.55190	0.53726

We trained on these models using three standard learning algorithms: logistic regression, a k-nearest neighbours classifier using the Minkowski distance, and a 3-layer shallow neural network. The neural network implementation was adapted from the CS 224U distributed word reps code lab. For testing data, we used approximately 500 additional Tweets from each topic group that were not included in the original subset, which consisted of 6000 Tweets from each group. Each classifier outputted the predicted topic label for each Tweet in the test dataset, and test accuracy was measured by simply calculating the percentage of test Tweets that were correctly classified. Table 1 summarizes the results.

Word Representation Vectors: As shown in Table 1, our best-performing classifiers, which utilised only basic word representations, achieved a test accuracy of approximately 60%. Treating randomly predicted labels as a baseline yields a baseline of roughly 3.7%. We can thus conclude that our model achieved a significant improvement in prediction accuracy over the baseline.

When examining the performance of the three-word representation vector models used to classify Tweets into hashtags, raw term frequency and tf-idf-based word vectors yielded relatively strong results. In contrast, GloVe vector representations yielded significantly lower test accuracy. This can be explained by the general focus of GloVe vectors on the relationships between words as opposed to the relationship between words and documents. The language used in our Twitter dataset tended to be informal and relatively simplistic. Consequently, the vast majority of words that appeared in the corpus had very different meanings from one another, and the occurrence of strict synonyms was infrequent. To illustrate, using a cosine similarity metric, the four words whose GloVe vectors were most similar to the word "happy" are: "apparently", "OS", "directly", and "socialism". From a linguistic point of view, none of these words is very similar to the word "happy", which suggests that the classification performed using the GloVe vectors was very noisy. In contrast, word vector representations such as tf-idf, which focus more on the relationship between tokens and Tweets, can be expected to perform much better with this dataset.

Sentiment Feature As seen in the results from Table 1, the sentiment feature did slightly worse for all three learning models. This can be attributed to the fact that we used our sentiment dataset to train the model, which labelled our training and test data for topic classification. Additionally, our topic dataset may not provide the highest magnitude sentiments. Fortunately, adding this feature to the ensemble model resulted in a slight increase in accuracy (see Section 5.2.2), likely providing a more accurate signal for specific Tweets that contained strong sentiment. Sentiment intuitively seems like it should produce an improvement, given that each training and test score is correctly labelled better than the current accuracy of our model. Future work could entail giving more substantial

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

confidence scores a higher weight as a feature, rather than the signal being simply a binary "positive" or "negative" sentiment value.

Thus, Tweets with a more vital score would result in a higher weight being given to that feature, while more neutral scores would revert to the original word representation features. This would help classify topics not very "opposite" in sentiment, such as "Egypt" and "Indonesia".

OpenIE Relation Extraction Table 2 shows that the addition of relation extraction also performed slightly worse for all three learning models, although it did outperform the sentiment feature. Examining the relations extracted by OpenIE provides some plausible explanations. For example, many of the Tweets in our dataset were written in non-English languages, such as Japanese and Russian, and therefore, OpenIE was unable to extract any relations from these Tweets. Thus, the effect of incorporating information from extracted relations was not as widespread as we had hoped.

Since relation extraction relies on human intuition in interpreting language, this addition is expected to make a positive contribution. Indeed, for Tweets corresponding to the `Obama` tag, about 81% of them had a relation extracted corresponding to "Obama", referencing facts such as "[is the] President" or "[lives in the] White House". However, after testing, it turns out that not all these Tweets were correctly classified. Future work can focus on incorporating relation extraction information more effectively. Perhaps a side model can be trained to determine which types of relations or which words or subjects are the most important, allowing for easy classification of any Tweets that contain these phrases.

B. Ensemble Model

As mentioned in Section 2.4, previous literature suggests that ensemble systems have the potential to improve performance. After analysing the strengths and weaknesses of each model in Section 5.1, we predicted that an ensemble system incorporating different areas of NLU could work very well. Since the models described above employ different features, the strengths of each could compensate for the weaknesses of other models.

Implementation. To start our implementation, we focused on the logistic regression output of each model. Logistic regression outputs a probability score for each possible candidate label, and the predicted classification (when used standalone) is simply the label with the highest probability score. Our ensemble system aggregates the probability scores of all candidate topics for each Tweet. We employed a data-driven approach to identify the optimal set of weights; our algorithm systematically tests various weights for each model to determine the set that yields the best performance. Thus, the aggregation is a sum of weighted probability scores. For Tweet t, let MODEL(t) denote the probability scores given by training logistic regression on MODEL and predicting the label for t. Then, to calculate ENSEMBLE(t), we calculate:

$$\text{ENSEMBLE}(t) = \alpha \times \text{FREQ}(t) + \beta \times \text{TFIDF}(t) + \gamma \times \text{GLOVE}(t) + \delta \times \text{SENT}(t) + \epsilon \times \text{REL}(t)$$

where $(\alpha, \beta, \gamma, \delta, \epsilon)$ are the specific weights for each model's scores. Algorithm 1 (on the next page) provides the pseudocode for our algorithm.

Algorithm 1: Ensemble system.

```
α, β, γ, δ, ϵ ← 0
for all Tweets t do
    Compute FREQ(t), TFIDF(t),GLOVE(t), SENT(t), and REL(t)
end for
for all α from 0 to 1, every time increment by 0.1 do
    for all β from 0 to 1, every time increment by 0.1 do
        for all γ from 0 to 1, every time increment by 0.1 do
            for all δ from 0 to 1, every time increment by 0.1 do
                for all ϵ from 0 to 1, every time increment by 0.1 do
                    Compute ENSEMBLE(t) for all Tweets t
                    Evaluate classification accuracy of (α, β, γ, δ, ϵ)
                end for
            end for
        end for
    end for
end for
return Best (α, β, γ, δ, ϵ)
```

Results: After running our algorithm to determine the best weights for ENSEMBLE(t), we found that the weight set

$$\alpha, \beta, \gamma, \delta, \epsilon) = (0.5, 0.3, 0.9, 0.1, 0.4)$$

Provided the best performance, yielding a classification accuracy of 0.64373. Although not by a significant amount, this accuracy surpasses the performance of any individual algorithm discussed in Section 5.1. (The highest individual algorithm was TFIDF at 0.61264 for logistic regression.) As discussed earlier, the intuition as to why ENSEMBLE may work better is that each model contributes different strengths to the overall system. Thus, we can potentially identify a broader range of features than can be collected with just one stan- dalone model. To illustrate this, we examine the classification accuracies obtained by varying one weight parameter while holding all others fixed at their optimal values.

For example, to roughly see the contribution of FREQ, we vary weight α and hold all other weights fixed, as shown in Figure 1.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

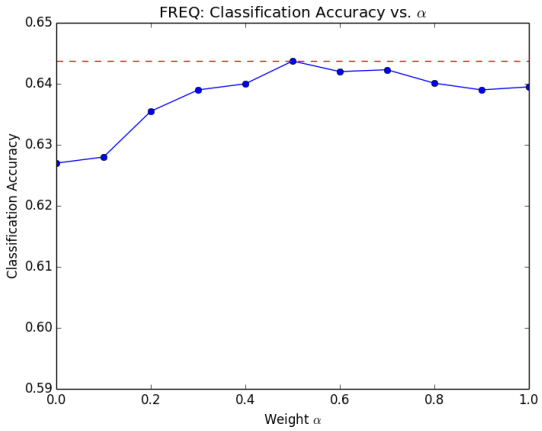

Fig. 1. Classification Accuracy While Varying A.

As expected, the graph for FREQ peaks at $\alpha = 0.5$, the optimal value discovered earlier. Compared to the case where it is nonexistent ($\alpha = 0$), the optimal addition of the FREQ model increases our test accuracy by approximately 0.016. We can compare this to a similar graph of a model that did not perform very well, such as SENT, shown in Figure 2.

Again, as expected, the graph for SENT has a peak at its optimal value $\alpha = 0.1$. This time, however, we notice that the net contribution given by this model is larger at around 0.023. This is interesting because SENT was one of the poorer-performing standalone models. This suggests that sentiment analysis – a very different approach to topic classification compared to those offered by standard word representations – provides more new information that complements what has already been parsed, yielding better accuracy. Thus, in the face of the vast complexity of Twitter content and language, our results suggest that an ensemble model that attempts to capture a wide variety of features using different models may be a practical approach.

Building on the idea that different models identify different features, other models may excel at classifying different topics. Thus, future work on this ensemble model could implement a weighting scheme that varies based on which topics are under consideration, allowing for a more fine-tuned classification.

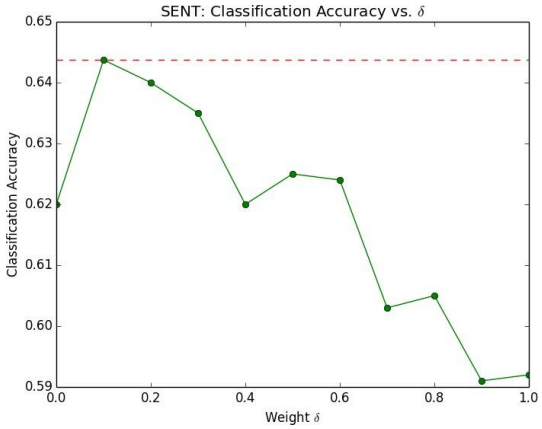

Fig. 2. Classification Accuracy While Varying Δ.

VI. CONCLUSION

Our project explored various word representations for the Tweet data and tested each on a range of models to determine which would yield the best test accuracies. Additionally, we utilised key points from various works in the area that had been previously determined and applied them to our task of classifying Tweets without relying on the main hashtag. Overall, the logistic regression and neural network models yielded the best results, with the TF-IDF vector representation appearing to capture the most information. Sentiment analysis and incorporation as a signal initially did not produce a positive outcome. Still, when combined with other models in predicting test data, it led to improved performance and classification of some Tweets. Relation extraction also showed promise as an addition to word representations. Finally, our ensemble model successfully incorporated the positive aspects of each model and signal, producing a more comprehensive model that captures elements of the training data that a single model could not cover.

DECLARATION STATEMENT

Funding	No, I did not receive.
Conflicts of Interest	No conflicts of interest to the best of our knowledge.
Ethical Approval and Consent to Participate	No, the article does not require ethical approval or consent to participate, as it presents evidence.
Availability of Data and Materials	Not relevant.
Authors Contributions	All authors have equal participation in this article.

REFERENCES

1. K. Lee, D. Palsetia, R. Narayanan, M. Patwary, A. Agrawal, A. Choudhary. Twitter Trend- ing Topic Classification. IEEE, 2011. https://doi.org/10.1109/ICDMW.2011.171
2. D. Ramage, S. Dumais, D. Liebling. Characterizing Microblogs with Topic Models. AAAI, 2010. https://doi.org/10.1609/icwsm.v4i1.14026
3. J. Li, A. Ritter, C. Cardie, and E. Hovy. Primary Life Event Extraction from Twitter Based on Congratulations/Condolences Speech Acts. EMNLP, 2014.
4. J. Pennington, R. Socher, C. Manning. GloVe: Global Vectors for Word Representation. EMNLP, 2014. https://doi.org/10.3115/v1/D14-1162
5. Open Information Extraction System, v4.0. University of Washington. Accessed 6/1/2015word at https://github.com/knowitall/openie.

AUTHOR'S PROFILE

Suraj Shinde Patil is a final-year Computer Science & Engineering student at Sanjay Ghodawat University. During his third year, he undertook a one-month internship at ITNium Technologies, focusing on Cloud Computing. Proficient in Python, C++, and Machine Learning, he excels in algorithms and problem-solving. Notable projects include the development of a Netflix Show Recommendation System and a Real-Time Human Detection and Counting System. Outside academics, Suraj is passionate about team sports, such as football and cricket, and actively participates in weekly matches with local teams. He is also an engaged member of the Computer Society of India (CSI) club, contributing to tech-related discussions and activities.

Muskan Jain is a Computer Science and Engineering student at Dr. A. P. J. Abdul Kalam Technical University. She has a strong enthusiasm for AI/ML and has delved into various domains, including computer vision, image analysis, and mobile app development using Flutter. During her academic journey, she has shown a keen interest in Artificial Intelligence and has completed several projects in this field. Notable projects include the development of a chatbot utilising natural language processing and a Sentiment Analysis tool for Twitter data. Muskan is also proficient in Machine Learning and has worked on several projects, including developing a Movie Recommendation System and a Face Recognition System. In addition to her academic pursuits, Muskan is an avid swimmer and enjoys participating in swimming competitions. She is also an active member of the Robotics Club, where she contributes to discussions and participates in tech-related activities.

Harsh Ahuja is a student at Dr. A. P. J. Abdul Kalam Technical University pursuing a Computer Science and Engineering degree. His academic interests are centred on Convolutional Networks, Computer Vision, and Web Development, reflecting his passion for cutting-edge technologies and a strong foundation in these fields. Harsh has completed several projects in the field of Web Development, including the development of a Social Media Platform using Flask and a Web Application for Online Shopping. He is also proficient in Machine Learning and has worked on several projects, including the development of a Speech Recognition System and a Face Recognition System. In addition to his academic pursuits, Harsh is an avid badminton player and enjoys participating in badminton tournaments. He is also an active member of the Robotics Club, where he contributes to discussions and participates in tech-related activities.

Akshay Mathur is a full-stack developer serving as Software Developer 1 at Mercer Mettl in Gurgaon. He earned his degree in Computer Science Engineering from Manipal University, Jaipur. Before his current role at Mercer, Akshay completed an internship as a Product Developer at Comviva, where he contributed to the Digital Business Solutions team by developing web pages for their CRM product. He excels in algorithms and problem-solving, with proficiency in Java, ReactJS, Python, and Machine Learning. He has completed projects in domains such as natural language processing and full-stack development for interactive websites. His interests include listening to music and playing outdoor sports, such as badminton and table tennis.