Predicting Personal Attitudes Using Contextual Microblog Activity Logs

Md. Enamul Haque, Eddie C Ling, Aminul Islam, and Mehmet Engin Tozal
School of Computing and Informatics
University of Louisiana at Lafayette
Lafayette, LA 70504
Email: {enamul,eddie,aminul,metozal}@louisiana.edu

***Abstract*—In this paper, we present a corpus model to show how personal attitudes can be predicted from social media or microblog activities for a specific domain of events such as natural disasters. More specifically, given a user tweet and an event, the model is used to predict whether the user will be willing to help or show a positive attitude towards that event or similar events in the future. We present a new dataset related to a specific natural disaster event, i.e., Hurricane Harvey, that distinguishes user tweets into positive and non-positive attitudes. We build Term Embeddings for Tweet (TEmT) to generate features to model personal attitudes for arbitrary user tweets. Finally, we evaluate the effectiveness of our method by employing multiple classification techniques on the newly created dataset.**

I. Introduction

People have varying interests while using online platforms such as Twitter and Reddit. They may try to achieve specific or random goals based on both geographical and environmental contexts. For example, online users living near festival areas are more likely to share contemporary events happening around them. On the other hand, users with positive mindset are more likely to care for disastrous events such as floods and hurricanes. In such cases, the geographical parameter plays a vital role in deciding the user-level attitude toward the affected. In addition to that, individuals and organizations who are willing to show their support towards such events either by volunteering physically or financially, can spread their activity on social media, e.g., using Twitter. By doing so, others can easily find out the intended users who are willing to help during such events in the future. Figure 1 demonstrates a sample tweet that shows a user letting people know about food and shelter during Hurricane Harvey.

Fig. 1: An example tweet during Hurricane Harvey to let affected people know about shelters.

In this paper, we present a user attitude prediction model based on a specific natural disaster event. Particularly, we are interested to find out the answer for the following question. Given a user tweet about an event, what is the prospect that the user is willing to help or show positive attitude towards that event or similar events in the future assuming that people exhibit consistent behaviors? To the best of our knowledge, there is no such work that exactly matches with our problem domain. However, we present several related works from other domains in Section II. Most of these works either use wisdom of words or pretrained models to solve the relevant problems. Some of those approaches lack generalization and scaling in terms of different problem domains and data properties. We present a solution for user attitude prediction in disastrous events in such a way that it can be extended to other event domains as well.

In this work, we use Twitter activity logs and take Hurricane Harvey as our context. We first create labeled datasets related to Hurricane Harvey from Twitter. Next, we process the datasets using the *Skip-gram* model to generate the TEmT corpus model. Then, we translate the labeled tweets into a feature matrix using the TEmT corpus and generate a classification model based on the feature matrix. Lastly, each new tweet is converted into a feature vector, using the TEmT corpus and labeled according to the classification model. The proposed solution can be used for both offline and streaming data scenarios due to the pre-built corpus model.

This paper makes the following contributions.

- We create labeled datasets from Twitter for a specific disaster event, Hurricane Harvey, that will be helpful for user attitude prediction towards similar events in the future.
- We use Term Embeddings for Tweets (TEmT) corpus model, a specification of word embeddings [1], as a feature generation scheme to be used for the classification processes.

We employ TEmT corpus model to generate term embeddings for tweets on a manually collected dataset. Our experimental results demonstrate 0.738, 0.748, 0.971 and 0.845 for accuracy, precision, recall and F1 scores, respectively using Linear discriminant analysis for a preferable vector length of 20. In addition, we observe similar scores for Logistic regression and Decision tree classifiers.

The rest of this paper is organized as follows. Section II presents previous works that are closely related to our problem

domain. Section III describes the methodology of our proposed approach. Section IV presents the dataset preparation steps and experimental evaluations based on multiple classification methods. Finally, in Section V we conclude our work.

II. Related Work

One of the essential efforts on analyzing tweets [2] focuses on predicting future actions of a user. The authors employ supervised learning by targeting past events, ongoing events, and events that are likely to occur in the future to determine whether people participate in the events that they tweet about. Another work on analyzing tweets figures out if users agree or disagree with other topics [3]. They model inter-topic preferences of Twitter users and come up with a linguistic pattern design in which people agree and disagree about a specific topic.

Twitter data is also utilized in the medical field. In [4] the authors use twitter data to surveil and predict infectious diseases. Influenzas are common topics of communication and the authors use the symptom words such as "fever" and "headache" as a clue of an upcoming influenza outbreak.

Furthermore, political stances have been extracted from tweets. A weakly supervised method is presented in [5] to study politician's stance over different issues as well as agreement and disagreement patterns among them. The authors model the dynamic nature of political discourse on Twitter and focus on a small set of politicians and issues. Similar to extracting political stances, tweets are used to predict election winners and losers by incorporating individual Twitter users' predictions [6]. First, a log-linear classifier is trained to detect positive veridicality. Then, the authors forecast uncertain outcomes by aggregating users' explicit predictions.

Moreover, tweets are analyzed to reduce gang-related violence in [7]. The authors developed a part-of-speech tagger and phrase table to identify tweets that convey grief and aggression.

A different social media platform, Pinterest, is used to understand user intents in terms of temporal range and goal specificity [8]. The authors develop a framework which combines survey-based methodology with observational analysis of user activity. They quantify users' intent and examine their subsequent behaviors. The authors report that users with specific goals search more and consume less content than users without specific goals.

Unlike the previous works, in this paper we propose a specification of word embeddings to build a corpus model for user attitude prediction based on what the user tweets are about. Our goal is to find Twitter users that may potentially help in an event of a disaster. We used users' past tweets to predict if they would be helpful or not during a nearby disastrous event.

III. Methodology

In this section, we present the problem statement and the solution along with its implementation.

A. Problem Statement

In this study, we consider natural disasters as an event domain or context. Our prime objective is to find the users who are more likely to help or show positive attitude towards natural disasters such as Hurricane Harvey. Considering the problem domain, the following objective function refers to the probability or score of a tweet to be labeled as positive attitude or not, given the tweet, corpus, and a classification model

$$\arg\max_{y \in Y} \; P(l = y \mid t, \mathcal{M}(\theta), \mathcal{C}) \tag{1}$$

where t indicates a new tweet instance, l is the label of t, $Y = \{0, 1\}$ is a binary label set with 1 being positive attitude and 0 being non-positive, $\mathcal{M}$ refers to `TEmT` corpus model, θ represents corpus model parameter(s), and $\mathcal{C}$ refers to a classification model.

We present our proposed approach with a minimal flowchart describing the step-by-step processes for predicting user attitude in Figure 2. The step numbers are labeled from 1 to 10 on each transition arrow between the process blocks. First, datasets are processed using *Skip-gram* model to generate `TEmT` corpus model. Then labeled tweets are processed at step 3 by `TEmT` model and saved as labeled feature matrix at step 4. Predefined classification algorithms are used on the labeled feature matrix to generate classification models at steps 5 and 6. At steps 7 and 8, new test tweets are processed again using `TEmT` corpus model to populate an unlabeled feature matrix. Finally, at steps 9 and 10, the unlabeled feature matrix is fed into the pretrained classification models for evaluation. Note that the proposed solution can easily be transferred to other platforms and domains.

B. The `TEmT` Model

We first describe *Skip-Gram*, CBOW [9], [1] and *Paragraph Vector* models [10] from which `TEmT` is developed. Next, we introduce the related theory along with its implementation. Finally, we present the classification algorithms that are used with `TEmT` during our experimental evaluations. We do this to accentuate the fact that the underlying vector representation model used for feature generation in `TEmT` is transferable to solve problems in other domains only by feeding domain specific datasets to the model.

Both continuous bag of words and skip-gram models, together called as *word2vec*, were proposed as efficient neural language models to learn embeddings (or vector representations) for words. The *word2vec* model creates dense vector representations of words which carries semantic meanings and are useful in a wide range of applications from sentiment analysis to on-line product recommendations. The most useful aspect of the word embeddings is that the vectors interpret the semantic meanings of the words. For example, words with similar/opposite connotations are inclined to have similar/opposite vectors considering the cosine distance measure. The working principle of the two models (CBOW and Skip-gram) are opposite of one another. In CBOW model, a word is predicted given a context as an input which can be a single or

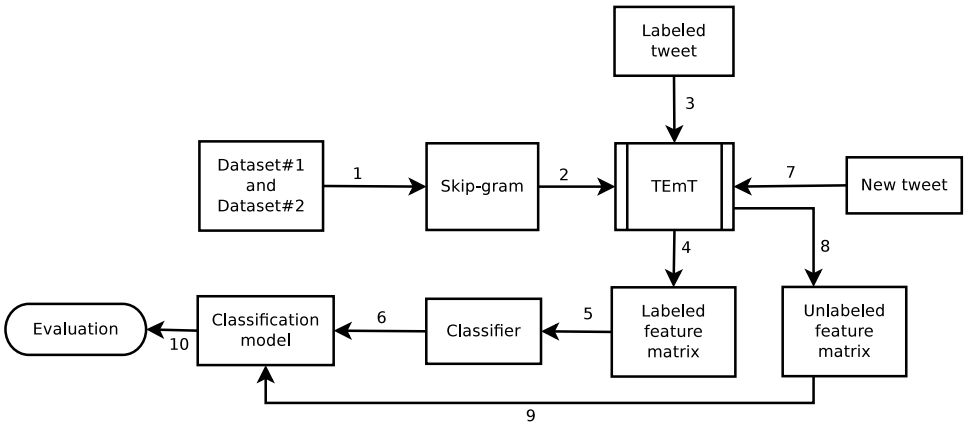


Fig. 2: A simple overview of the experimental flow of our approach to predict domain specific personal attitudes using feature generation and supervised learning.

multiple sentences. In Skip-gram model, however, the context is predicted given a word as an input.

Paragraph vector model [10] is a learning framework that learns representations of texts of any length such as sentences, paragraphs, or documents. It is also known as *doc2vec* which is an extension of *word2vec* for learning embedding vectors for documents. The *doc2vec* model creates unique vector representations for paragraphs with the help of *word2vec* model. The word vectors in a paragraph are averaged and/or concatenated to obtain the paragraph vector. Likewise, we use the coordinate wise average method to combine the tweet embeddings where each word corresponds to a term in the tweet.

The primary goal to build `TEmT` is to create contextual term embeddings (or vector representations) for tweets which can be used as features during classification processes. Note that `TEmT` generates a corpus model from the tweet collection which can be used with different classifiers. In addition, `TEmT` corpus model can be updated based on the arrivals of new terms which are not in the tweet collection.

C. `TEmT` Theory

We use the *Skip-gram* model to generate a unified `TEmT` corpus model for different tweets. The *Skip-gram* model (`word2vec`) requires the labeled tweets to construct the corpus model using a neural network. On the other hand, the vector aggregation concept from `doc2vec` is used to generate the final feature representation from the output of the hidden layer of the *Skip-gram* model.

Figure 3 shows the architecture to create a vector representation of a tweet term with respect to its surrounding terms with window size c. The double layer neural network consists of a hidden and an output layer. The hidden layer generates a k-dimensional vector representation of tweet term t_k. As our goal in this step is to generate the vector representations of tweet terms, we are more interested in the output of the hidden layer. The input vector is transformed into "one-hot" representation that refers to a vector of length T consisting of all zeros and a one at the i-th position. T refers to the vocabulary size, and the i-th position is one for the i-th term in the vocabulary. As a result the i-th row of the hidden layer will be selected, which is the vector representation of the i-th input term. Once Skip-gram model is trained on the whole corpus of terms, the vector representations are obtained from the output of the hidden layer for the corresponding terms. The vector length k refers to the feature size which is an important parameter for our `TEmT` corpus model. For a tweet of size [1] $|S|$, the `TEmT` corpus model generates a feature vector $\boldsymbol{v}$ of size k by applying vector averaging using Equation 2.

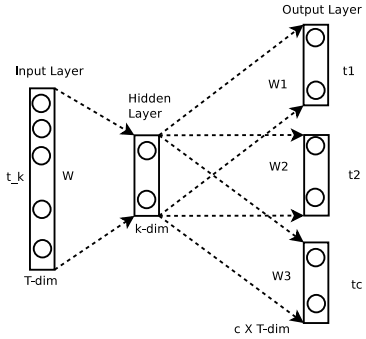


Fig. 3: Skip-gram model predicting context terms given an input tweet term (k-th term of a tweet). This architecture provides a vector for all the context terms of length c. W represents the weights.

$$\boldsymbol{v} = \frac{1}{|S|} \sum_{i=1}^{|S|} \texttt{TEmT}[t_i] \tag{2}$$

The training objective of the Skip-gram model is to find the term representations that are useful for predicting the context terms in a tweet. More formally, given a stream of training terms $t_1, t_2, t_3, \ldots, t_L$, the objective is to maximize

[1]The average term length observed in our experiments is 14.12. The size $|S|$ refers to the effective term length of a particular tweet.

the average log probability:

$$\frac{1}{L}\sum_{m=1}^{L}\sum_{-c\leq j\leq c, j\neq 0}\log p(t_{m+j}|t_m) \tag{3}$$

where c is the size of the training context or window which is a function of the center term t_m and L is the number of input terms. The training time to build the model grows with the increase in context size.

The basic Skip-gram formulation defines $p(t_{m+j}|t_m)$ using softmax function:

$$p(t_O|t_I) = \frac{\exp\left({v'_{tO}}^{\top} v_{tI}\right)}{\sum_{t=1}^{T}\exp({v'_t}^{\top} v_{tI})} \tag{4}$$

where v_t and v'_t are the "input" and "output" vector representations of term t, and T is the vocabulary size. Computing $\nabla \log p(t_{m+j}|t_m)$ (gradient) in Equation 3 of basic Skip-gram model is computationally expensive. However, in our approach, the computational cost is limited due to the smaller vocabulary size. In the original Skip-gram model, a hierarchical softmax and negative sampling are used to minimize the cost.

D. TEmT Implementation

In this section we present the details of the TEmT feature generation approach using Algorithms 1 and 2. Note that, two types of training inputs are mentioned in the algorithms. First, corpus training vocabulary is needed to generate TEmT corpus model. Second, training and test tweets are required to generate feature matrix from TEmT corpus models as input for the classification algorithms. During feature matrix generation, tweet types (positive/non-positive) are known only for the training tweets.

Algorithm 1 TEmT corpus model generation

Input: input tweets ▷ corpus training vocabulary
Output: *model* ▷ corpus model
Initialization : vocab = ∅, model = ∅
1: generate terms from tweets
2: **for** each tweet term **do**
3: vocab = vocab ∪ term
4: **end for**
5: model = build skip_gram model using Equation 3 and 4 from vocab.
6: **return** *model*

Algorithm 1 explains how TEmT corpus model is generated for the training tweet vocabulary. The detailed process is described in section IV. Selected tweets are treated as input to the algorithm. The tweets are then preprocessed to remove the unnecessary information such as stop words. Once the tweets are preprocessed, lines 2-4 creates a vocabulary from all the unique terms present in the training tweet corpus. Line 5 generates corpus model from the tweet vocabulary using the Skip-gram model. Finally, line 6 returns the TEmT corpus model.

Algorithm 2 Feature matrix generation

Input: input tweets ▷ training or test tweets
Output: feature matrix, $\boldsymbol{V}$
Initialization: n, k, $\boldsymbol{V} = \mathbf{0}_{n,k+1}$, $\boldsymbol{v} = \mathbf{0}_k$
1: **for** $i = 1$ to n **do**
2: $\boldsymbol{t}$ = i^{th} tweet
3: l = label of i^{th} tweet {0,1} ▷ known for training tweets only
4: $model$ = TEmT corpus model
5: s = length of $\boldsymbol{t}$
6: **for** $j = 1$ to s **do**
7: $\boldsymbol{v}$ = $\boldsymbol{v}$ + $model[\boldsymbol{t}[j]]$
8: **end for**
9: $\boldsymbol{v}$ = $\boldsymbol{v}$ / s
10: $\boldsymbol{V}_{i,1:k}$ = $\boldsymbol{v}$
11: $\boldsymbol{V}_{i,k+1}$ = l
12: **end for**
13: **return** $\boldsymbol{V}$

Algorithm 2 generates a feature matrix from the training tweets. The number of the tweets (n), feature length (k), feature vector ($\boldsymbol{v}$), and feature matrix ($\boldsymbol{V}$) are initialized during the initialization phase. We use $\mathbf{0}_{n,k+1}$ to refer a matrix with n rows and $k+1$ columns having all the elements initialized to zero. $\boldsymbol{t}$ refers to a term vector of dimension s. $\boldsymbol{v}$ defines a k-dimensional vector of zeros ($\mathbf{0}_k$). Lines 1-12 generate the feature matrix of dimension $n \times (k+1)$. Line 2 converts the i^{th} tweet into a term array. Line 3 finds out the corresponding label of the i^{th} tweet. Note that the test tweet types can not be directly known at this step. Line 4 is used to select the TEmT corpus model that was built using Algorithm 1. Line 5 calculates the tweet length to support varying tweet sizes. Lines 6-8 create a feature vector for each term of the i^{th} tweet using the corpus model, $model$, and aggregate them into vector $\boldsymbol{v}$. $model[\boldsymbol{t}[j]]$ at line 7 refers to the k-dimensional vector representation of the j-th term of the i-th tweet. The algorithm aggregates the vectors by element-wise vector addition. Each term of the i^{th} tweet is represented as a vector of length k. Line 9 averages the aggregated vector representation of the i^{th} tweet by element wise division using the tweet size, s. To populate the feature matrix $\boldsymbol{V}$, line 10 copies the i^{th} vector of length k to i^{th} row of matrix $\boldsymbol{V}$. $\boldsymbol{V}_{i,k+1}$ position is filled by the label of i^{th} tweet collected at line 3. This process is repeated for all tweets in the dataset. In case of test tweets, the same procedure is applied to generate a feature matrix representing the test tweets. The only difference for the test tweet dataset is that we remove the $k+1$-th column. At the end, the algorithm returns a feature matrix $\boldsymbol{V}$.

The low dimensional vector generation architecture is also depicted in Figure 4. The figure shows a toy example tweet with five terms $t_1, t_2, \ldots, t_5$. For each of the terms t_i, the corresponding vector v_i of length k is generated using the TEmT corpus model. The vectors are averaged to get the final feature vector of dimension k for all the five terms present

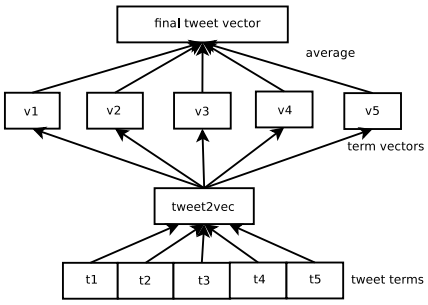


Fig. 4: A framework for learning term vectors. This diagram is shown for an example tweet with five terms $(t_1, t_2, t_3, \ldots, t_5)$ and their corresponding vectors $(v_{t_1}, \ldots, v_{t_5})$ of length k which is chosen based on empirical results.

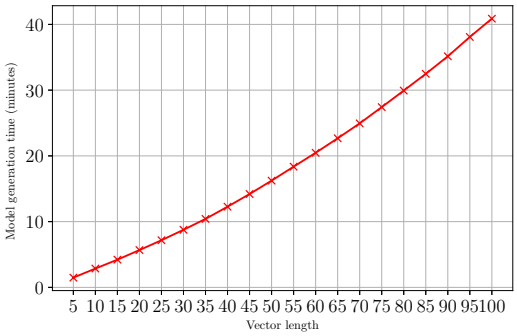


Fig. 5: TEmT corpus model generation time for different vector lengths.

in the tweet. For this particular example, we explain how the averaging is performed using coordinate wise computation. As each term has feature vector length k, the vector representation of term t_1 is $v_{t_1} = [v_{11}, v_{12}, \ldots, v_{1k}]$. Similarly for term t_5, we have $v_{t_5} = [v_{51}, v_{52}, \ldots, v_{5k}]$. Thus the coordinate-wise average of these five terms for a tweet is $\boldsymbol{v}^k$, where $\boldsymbol{v}^k = \frac{1}{5}\sum_{i=1}^{5}[v_{i1}, v_{i2}, \ldots, v_{ik}]$. It should also be mentioned that the original paper [1] refers to concatenation as another method of aggregation.

E. TEmT Classification

We consider TEmT in the following supervised classification setting. We create labeled tweets $\boldsymbol{V} = (\mathbb{X}, \mathbb{Y})$ with the help of TEmT corpus model where $\mathbb{X}$ is the feature vectors and $\mathbb{Y}$ is the class labels. Our tweet dataset is labeled as 0 or 1 based on the user attitude towards Hurricane Harvey, resulting $\mathbb{Y} \in \{0, 1\}$. This setting makes our tweet classification task a binary classification problem. We use three classification algorithms, namely Logistic regression (LR), Decision Tree (DT) and Linear discriminant analysis (LDA) for the parameter $\mathcal{C}$ in Equation 1 that provides final classification decision for test tweets. Note that our proposed model can accommodate both streaming and offline data as test tweets.

IV. Experiments

In this section, we first present the dataset preparation steps and then experimental evaluations based on different classification techniques. Please note that, European Union's General Data Protection Regulation (GDPR) limits web scraping without user consent. Hence, our approach might not be applicable under GDPR.

A. Data set description

We collected all English tweets within 15 miles radius from the center point of Houston, Texas with coordinates 29.7604 N, 95.3698 W, to be specific. The tweets have dates ranging from January 1, 2017 to October 16, 2017 to capture both relevant and non-relevant tweets. We split the tweets into two datasets according to their dates. Dataset#1 has tweets ranging from January 1, 2017 to July 16, 2017. Dataset#2 has tweets from July 17, 2017 to October 16, 2017. We first identify the tweets that are related to Hurricane Harvey using Dataset#2. We did this by querying Dataset#2 with five keywords which are flood, disaster, hurricane, Texas, Houston, and Harvey to get tweets that contain any of those five keywords. We then use the resulting tweets to find the top keywords that appear in those tweets. We ranked each unique word according to their frequencies and listed them from high to low. We had three graduate students who looked through all the unique words and chose all the keywords that might be related to Hurricane Harvey to be able to label the tweets later. Then, we compared the three lists of keywords and chose only the unique words that appear in at least two of the three lists. Finally, we ended up with the keywords presented in Table I.

Next, we temporarily label a tweet as 1 if any of the 35 words appear in the tweet and 0 otherwise. Temporary *label_x = 1* indicates that the tweet is related to the event and *label_x = 0* indicates that the tweet is unrelated to the event. We then look for users who has tweeted about the event. We did this by searching for users that has at least one tweet with *label_x = 1*. If a user tweeted about the event, we add a final *label_y = 1*. For a user who did not tweet about the event, we add a final *label_y = 0*. Next, we create a list of users that has *label_y = 1*. We compare this list of users with tweets in Dataset#1. If a user from the list has tweets in Dataset#1, all the tweets from the user in Dataset#1 have *label_y = 1*. We present a couple of sample tweets in Table II that show the labeling information.

Now, all the tweets in both Dataset#1 and Dataset#2 have *label_x* and *label_y*. We concatenate both Dataset#1 and Dataset#2 and randomize the order of tweets which were previously ordered by dates. We remove links (URLs), stop words, punctuations (without hashtags) and emojis from all tweets. Before performing tweet vectorization using Algorithm 2, we tokenized all tweets. Statistics of the two datasets are presented in Table III.[2]

B. Empirical Analysis of TEmT

We present TEmT corpus model generation time in minutes for varying vector lengths starting from length 5 to 100 with

[2]The datasets, TEmT models, and implementation code are made available online at: https://drive.google.com/drive/folders/1t8lkjQN6YEW3JwW_xWL4GMxAtXyXPtCz?usp=sharing

TABLE I: Keywords used for tweet collection and labeling.

#hurricaneharvey	#harvey	flood	hurricane	#hurricane
#prayforhouston	#houstonstrong	harvey	#houstonflood	flooded
water	#harvey2017	safe	#hurricaneharvey2017	#flood
need	center	#flooding	#prayfortexas	#texasstrong
victims	storm	affected	prayers	#harveyrelief
#help	rescue	#hurricanharvey	pray	helping
praying	donations	#rain	relief	help

TABLE II: A few sample tweets that are labeled as positive, 1, or non-positive, 0.

Sample tweets	Label
There are other channels to go through to help, if you can not find a more personal way to… https://www. instagram.com/p/BYYqNbnFHK8/ #1	1
This for all the people affected in Texas by hurricane Harvey SN I have friends and a small… https://www. instagram.com/p/BYaQ6vog4R6/	1
Pasadena Rodeos final night. #patgreen #aintnokneelinhere @Pasadena… https://www. instagram.com/p/BZsHaoyBzATP ekBLgV2iH3ay7_rkI2WJcrPrc00/	0
I love details like this...not just a flower or a candle—a mood. A motif. Just lovely. • In… https://www. instagram.com/p/BaU8H_plH-z/	0

TABLE III: Summary statistics pertaining to the collected tweets.

Dataset	% of +ve attitude	Unique user count	Tweet count	Max tweet length	Min tweet length	Average tweet length
Dataset #1	57.87%	18269	153994	50	1	14.41
Dataset #2	35.88%	32980	378120	49	1	14.12

intervals of 5 in Figure 5. The model generation time strictly increases as the vector length grows. Our experimental results involve the classification of tweets using `TEmT` generated features. We use Logistic regression (LR), Decision Tree (DT) and Linear discriminant analysis (LDA) for all classification steps with 10-fold cross validation. Additionally, we present the empirical results demonstrating the accuracy, precision, recall, and F1 scores with respect to different vector lengths.

Accuracy of a classifier is defined as the total number of accurate predictions over the total number of samples. Figure 6a shows the classification accuracy of `TEmT` based feature generation approach. In the figure the accuracy improvement is not significant between adjacent vector lengths. Moreover, in some cases the accuracy rates slightly drop between two consecutive vector lengths. However, the classification accuracy improves for larger vector lengths, in general. Note that the model generation time also increases as the vector length increases. Since our dataset is not symmetric accuracy is not enough to measure the performance of our models. In the following we look at precision, recall and F1 scores as well.

Figure 6b shows the precision scores. The figure presents the number of the instances being predicted correctly (true positives) divided by the number of all positively predicted instances, including the false positives. Figure 6b shows that the precision scores present a similar behavior to that of the accuracy scores. Increasing vector length slightly impacts the precision scores. On the other hand, the figure shows that DT classifier performs better than LR and LDA in terms of precision. Please note that in terms of accuracy, LR and LDA classifiers outperform the DT classifier.

Figure 6c shows the recall scores. Recall measures the sensitivity defined as the number of the instances being predicted correctly (true positives) divided by the number of all positive instances in the dataset, including the false negatives. Although there is a slight decrease in recall up to vector length 20 regarding LR and LDA classifiers, the recall remains more or less stable as the vector length increases further. Figure 6c shows that the recall scores present a similar behavior. Increasing vector length slightly impacts the recall scores. On the other hand, the figure shows that LR and LDA classifiers perform better than the DT classifier in terms of recall. Please note that in terms of precision, the DT classifier outperforms the LR and LDA classifiers.

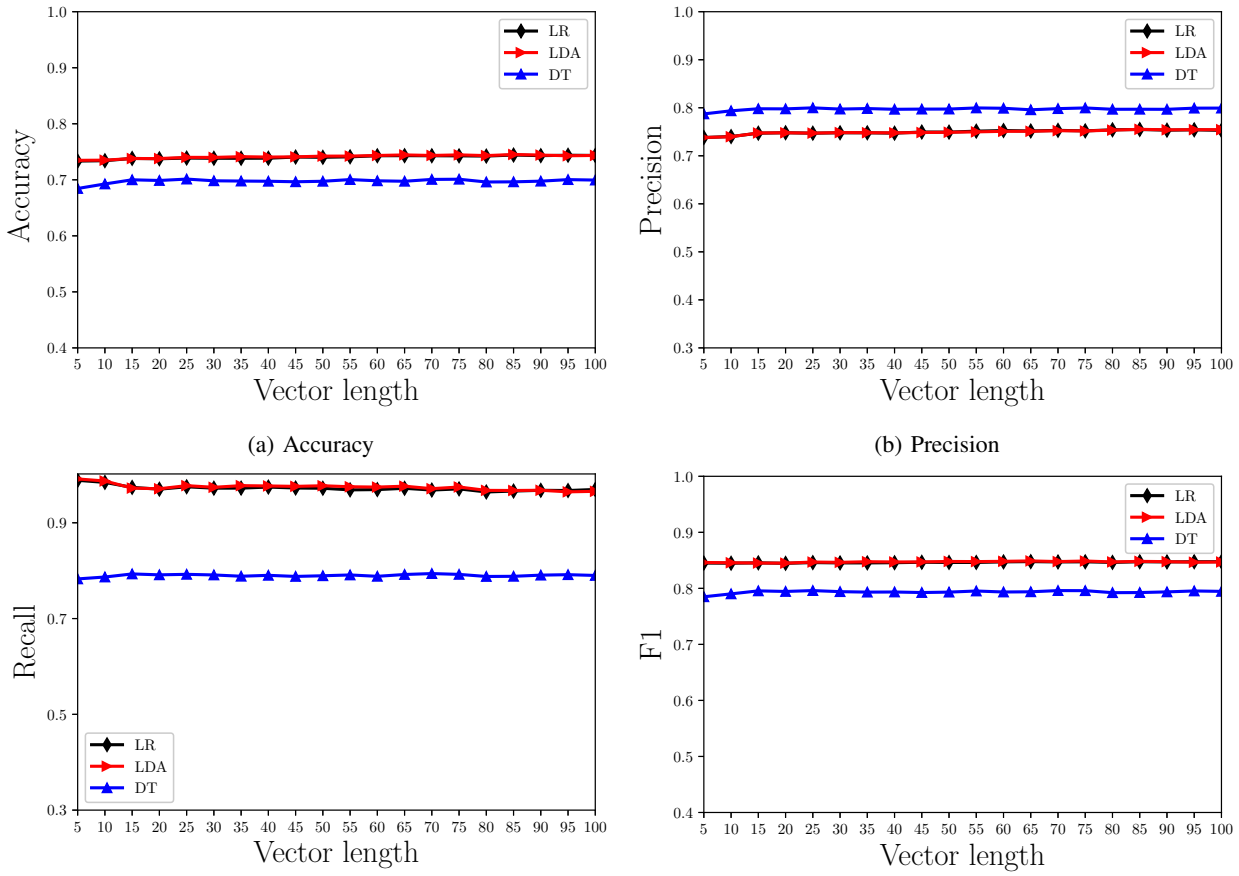
Due to this discrepancy, we look at the F1 scores in Figure 6d. F1 is defined as the harmonic mean of precision and recall and it presents a balance between the precision and recall scores. Similar to the other measures F1 is not significantly affected by the varying vector lengths. Moreover, LR and LDA classifiers perform better than DT in terms of F1 score.

Considering the accuracy, recall, precision, F1, and model generation time with respect to different vector lengths, we suggest using vector length 20 with the LDA classifier. Models with larger vector lengths achieve slightly better scores in all performance measures. However, the gain is not significant after vector length of 20.

Finally, Table IV summarizes the performance measures with respect to different classifiers. In addition we present the actual scores associated with the vector length 20 in the table.

V. Conclusions

We posed the problem of user attitude prediction for a specific domain from microblog activity logs as a supervised, binary class learning. In our formulation, the classes corresponds to either positive or non-positive attitudes. We



(a) Accuracy

(b) Precision

(c) Recall

(d) F1

Fig. 6: Classification accuracy, precision, recall and F1 scores for different classification algorithms based on varying vector lengths.

TABLE IV: Accuracy (Acc), precision (Prec), recall (Rec), and F1 scores for different classifiers.

Classifier	Parameters	Avg Acc	Acc@20	Avg Prec	Prec@20	Avg Rec	Rec@20	Avg F1	F1@20
LDA	solver = SVD	0.741±0.006	0.738	0.749±0.009	0.748	0.974±0.013	0.971	0.847±0.002	0.845
LR	solver=liblinear	0.740±0.006	0.738	0.750±0.009	0.748	0.972±0.011	0.970	0.846±0.002	0.845
DT	none	0.698±0.008	0.699	0.797±0.006	0.798	0.790±0.005	0.791	0.794±0.005	0.794

employed a specification of word embeddings, `TEmT` corpus model, using Twitter data relevant to Hurricane Harvey. Our proposed method consists of four major steps: data collection and preprocessing, model building, feature generation, and classification. One important positive aspect of our method is that the model building step is done only once before the feature generation, providing increased flexibility and computational cost reduction. Our experimental results demonstrate 0.738, 0.748, 0.971 and 0.845 for accuracy, precision, recall and F1 scores, respectively using Linear discriminant analysis for a preferable vector length of 20. Finally, we plan to investigate other event domains or contexts to demonstrate the transferability of our approach.

REFERENCES

[1] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[2] K. C. Sanagavarapu, A. Vempala, and E. Blanco, "Determining whether and when people participate in the events they tweet about," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2017, pp. 641–646.

[3] A. Sasaki, K. Hanawa, N. Okazaki, and K. Inui, "Other topics you may also agree or disagree: Modeling inter-topic preferences using tweets and matrix factorization," *arXiv preprint arXiv:1704.07986*, 2017.

[4] I. Hayate, S. Wakamiya, and E. Aramaki, "Forecasting word model: Twitter-based influenza surveillance and prediction," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 76–86.

[5] K. Johnson and D. Goldwasser, "" all i know about politics is what i read in twitter": Weakly supervised models for extracting politicians' stances from twitter," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2966–2977.

[6] S. Swamy, A. Ritter, and M.-C. de Marneffe, "" i have a feeling trump will win.....": Forecasting winners and losers from user predictions on twitter," *arXiv preprint arXiv:1707.07212*, 2017.

[7] T. Blevins, R. Kwiatkowski, J. Macbeth, K. McKeown, D. Patton, and O. Rambow, "Automatically processing tweets from gang-involved youth: Towards detecting loss and aggression," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2196–2206.

[8] J. Cheng, C. Lo, and J. Leskovec, "Predicting intent using activity logs: How goal specificity and temporal range affect user behavior," in *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 2017, pp. 593–601.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[10] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1188–1196.