Int J Adv Eng Sci Appl Math
https://doi.org/10.1007/s12572-017-0197-2

Using social media for classifying actionable insights in disaster scenario

Samujjwal Ghosh[1] · P. K. Srijith[1] · Maunendra Sankar Desarkar[1]

Abstract Micro-blogging sites are important source of real-time situational information during disasters such as earthquakes, hurricanes, wildfires, flood etc. Such disasters cause miseries in the lives of affected people. Timely identification of steps needed to help the affected people in such situations can mitigate those miseries to large extent. In this paper, we focus on the problem of automated classification of the disaster related tweets to a set of pre-defined categories. Some example categories considered are resource availability, resource requirement, infrastructure damage etc. Proper annotation of the tweets with these class information can help in timely determination of the steps needed to be taken to address the concerns of the people in the affected areas. Depending on the information types or categories, different feature sets might be useful for proper identification of posts belonging to that category. In this work, we define multiple feature sets and use them with various supervised classification algorithms from literature to study the effectiveness of our approach in annotating the tweets with their appropriate information categories.

Keywords Disaster management · Information retrieval · Social media · Text categorization

✉ Samujjwal Ghosh
cs16resch01001@iith.ac.in

P. K. Srijith
srijith@iith.ac.in

Maunendra Sankar Desarkar
maunendra@iith.ac.in

[1] IIT Hyderabad, Sangareddy, India

1 Introduction

Micro-blogging websites like Twitter, Weibo etc. are extremely useful [1] during disasters, mass emergencies and crisis situations. At the time of massive disasters like earthquakes, floods, hurricanes etc. there is often significant damage to the infrastructure. This generally cripples the traditional communication networks and media such as television, newspaper, radio channels etc. Social media comes to the rescue [2] at a crucial time like this because of its easy accessibility (through satellite, wireless and mobile) and vast outreach. During various events like Earthquakes in Nepal (in 2015), Italy (in 2012), Guatemala (in 2012), wildfires in Colorado (in 2012), Australia (in 2013), typhoons in Philippines (in 2012, 2013), lots of people used the Twitter social media to exchange situational information, thereby enhancing situational awareness. Through social media, people who are in the affected areas can post messages that reflect the exact situation in those areas, the damages caused and the repair status, the needs of people, the status of rescue and relief operations etc. On the other hand, individuals, groups of peoples or organizations can express their willingness to help or mention the exact way in which they can be helpful in mitigating the effect of the disaster.

Although social media has immense potential to handle crisis situations, much of its potential is yet to be realized. Only recently, some work has started to leverage the use of social media for responses during emergency situations. As tweets posted during disaster may contain various kinds of information, it might be useful to automatically identify the exact nature of information that is present in a given tweet. This will help in identifying different actionable insights regarding the disaster scenario to various groups of people as well as different government and non-governmental

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197 LE TYPESET
MS Code : AEAM-D-17-00027 CP DISK

organizations. The information contained in a tweet may be regarding infrastructure damage, asking for medical help, reporting medical resources like medicines, medical toolkits available, requesting for resources such as food, water, blankets, clothes etc. Given a tweet posted during a disaster scenario, we want to identify, what type of information is present in the tweet. The different information types like resources available, medical resources needed, infrastructure damage etc. are viewed as different categories. In several cases, a single tweet may contain information about multiple information categories. We view the problem of automated identification of information type from a tweet as a multi-class multi-label classification problem and study the applicability of different algorithms towards this task. The major challenges for this task appears due to the short length of the tweets and the informal ways (frequent use of smiley, abbreviations, out-of-vocabulary words etc.) in which they are written. We define multiple feature sets for this scenario and evaluate the performances of several off-the-shelf classifiers for this task. The performances of the algorithms are measured against a benchmark dataset containing tweets posted during the Nepal Earthquake in 2015. The contributions of the work are given below:

- We identify different feature sets for representing the tweets. Along with tf–idf features we use few features derived from the tweet collection and also few manual features for this purpose.
- We use different classifiers for determining the categories to which the test tweets belong. The performance of each classifier, for different feature sets is analyzed in detail. We also evaluate the effect of adding the extra features in detail.

The structure of the rest of this paper is as follows. We discuss related work from literature in Sect. 2. We further discuss about the use of social media during disasters with specific attention to a dataset containing information on actionable insights in Sect. 3. Then, in Sect. 4, we define the problem of tweet classification. A description of the classifiers and various features used in the work are presented in Sect. 5. Our experimental set up is discussed in 6. Experimental results are presented and discussed in Sect. 7. We conclude the paper with a brief discussion of our findings in Sect. 8.

2 Related work

In this section we look at recent work from literature that deals with social media in disaster scenarios. We also look at various approaches that attempt to categorize short texts into several predefined categories.

2.1 Mining social media posts related to a disaster

In [3], the authors use a set of seed keywords for retrieving a set of micro-blogs that might be relevant for a particular disaster. Then they used binary classification algorithms to mark posts from this initial collection as relevant or irrelevant. Unigrams and bigrams from tweets are used as features to create a lexicon (list of terms) related to disasters. Positive class represents tweets that are directly or indirectly related to crises and irrelevant tweets are put into negative class. Then they used the positive class only to create the lexicons. The work presented in [4] discusses use of word2vec model for efficient retrieval of disaster related micro-blogs. The authors found that using word2vec improved their model over traditional features. The works presented in [5] retrieves disaster related tweets by using seed keywords and additional expanded queries using WordNet. In [6], the authors determine strategies for obtaining ground truth annotation for disaster related micro-blogs. They argue that people miss many relevant tweets to label when creating ground truths. Support vector machine with linear kernel was used select disaster related posts for which "ground truth" information would be obtained. In [7], the authors classify the tweets obtained for the Hurricane Sandy to different categories such as sentiment, action, presentation etc. For classification, they used Support Vector Machine and Naive Bayes algorithm. They used various information like unigrams, POS Tags, Named Entities, url, retweet information etc. as features to represent the tweets. The work presented in [5] calculated relevance score using cosine similarity between tweets and topics.

2.2 Categorization of short texts

In [8], Caragea et. al, created a disaster related information retrieval system called, *Enhanced Messaging for the Emergency Response Sector* (EMERSE). This system classifies and aggregates tweets and text messages about the Haiti disaster relief to deliver relevant information to appropriate personal. The EMERSE system works using feature abstraction technique where a set of features are grouped together to form abstract features. Abstraction is specifically done by selecting m-size partitions of the overall vocabulary. Munro et. al [9] focused on classifying medical text messages, written in "Chichewa" language, that were received by a clinic in Malawi and have shown that incorporating morphological and phonological variation could improve classification performance. In the work [10] rather than using bag of words approach, the authors created domain specific features from authors' profiles by filtering certain information is present or not. Their main argument is "authorship" of micro-blogs plays a crucial role in differentiating useful information when classifying

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197 LE TYPESET
MS Code : AEAM-D-17-00027 CP DISK

short texts. They used Naive Bayes classifier for their experiments. [11] analyzes the effect of different smoothing techniques in Naive Bayes algorithm on the classification performance. Various other work on classification of short tests are presented in [12–14]. In all these works, the authors used standard off-the-shelf classification algorithms like Naive Bayes, SVM etc. However, depending on the data and the classes under consideration they employ different feature sets in conjunction with tf–idf or word n-gram based features. The data and class information (actionable insights in disaster scenario) considered in our work are quite different from the short text classification works considered in these papers from literature. Hence we need to look for specific features that are suitable for the task of classifying actionable insights.

In the next section we will look into the various uses of social media to assess actionable steps in disaster related scenario.

3 Usage of social media during disasters

As mentioned in the Sect. 2, there are several efforts in classifying the information available in the micro-blogs posted during disasters. In [3], the classes are whether a given tweet is relevant to a disaster/crisis under consideration or not. In [7], the possible classes to which each disaster related-tweet can belong to are: reporting, sentiment, information, action, preparation and movement. Although these classes are useful, but they do not directly identify any "specific actionable insight." Fortunately, recently a dataset has been released [15], where each tweet is annotated with specific information from which definite action plans can be prepared. We center our work on that dataset for identifying actionable insights from disaster-related micro-blogs. We now give a detailed description of the dataset.

The dataset, called FIRE 2016 Micro-blog Track dataset [15], contains labeled data for disaster related tweets. The labeled data contains a collection of 2139 tweets categorized into 7 classes. The class details were given in TREC format. Description of each class contains four fields: the class ID, title (small title to denote the class), desc (short description of the class) and narr (detailed narrative of which text should be considered for this class). Example of a class description in TREC format is given below.

```
< num > Number: FMT1
< title > What resources were available
< desc > Identify the messages which describe the
availability of some resources.
< narr > A relevant message must mention the availability
of some resource like food, drinking water, shelter,
clothes, blankets, human resources like volunteers,
resources to build or support infrastructure, like tents,
water filter, power supply and so on. Messages informing
the availability of transport vehicles for assisting the
resource distribution process would also be relevant.
However, generalized statements without reference to any
resource or messages asking for donation of money would
not be relevant.
```

Each of these classes represents different types of information to be retrieved from the tweets. Also, sometimes one particular tweet might contain information related to more than one class. In such a scenario, all the applicable classes will be assigned to that tweet. If we count all such tweets which belongs to two or more classes only once, then total number of unique data points comes to 1551. Table 1 shows the class numbers and their class titles. In the original dataset the classes have numbers as $FMT1, FMT2, \ldots, FMT7$. We use the class numbers as 1, 2, ..., 7 respectively.

Next section gives the formal definition of multi-label multi-class tweet classification into predefined classes.

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197 LE TYPESET
MS Code : AEAM-D-17-00027 CP DISK

Table 1 Class numbers and their titles

Class number	Class title
1	Resources available
2	Resources required
3	Medical resources available
4	Medical resources required
5	Requirements/availability of resources at specific locations
6	Activities of various NGOs/government organizations
7	Infrastructure damage and restoration reported

4 Problem definition

Classifying tweets according to the type of information they contain is a multi-class multi-label problem. The definition of the problem under consideration is given below:

> Let, $T = \{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_N\}$ be a set of N tweets and $C = \{1, 2, \ldots, M\}$ be a set of M classes. Given a set of mappings of the from $\{\boldsymbol{t}_i, c_i^1, \ldots, c_i^k\}$ where tweet $\boldsymbol{t}_i \in T$ and class labels $c_i^1, \ldots, c_i^k \in C$, our goal is to find the class labels for a new tweet $\boldsymbol{t}_{new}$.

In this work, we also define multiple feature sets encoding variety of information that might be useful to represent the data, specifically when the goal is to identify specific actionable insights mentioned in the class descriptions.

In next section we discuss about the methods we followed to solve the problem.

5 Methodology

We model the problem as a multi-class multi-label supervised classification problem [16], where we want to predict multiple class labels for each instance. We study the usefulness of various multi-class classification algorithms such as decision tree, Naive Bayes, support vector machine and ensemble learning algorithms for this problem. These can be easily adapted to multi-label setting. Given the feature vectors of training tweets and their corresponding class labels as supervision each of the methods develops a classification model. This model can then be used for predicting the class labels for a new unclassified tweet. A brief definition of different classifiers and features used are given below.

5.1 Learning algorithms

5.1.1 Naive Bayes classifier

Naive Bayes is a generative classifier and make use of Bayes theorem in order to predict the class corresponding to a sample [17]. They make a class conditional independence assumption over the features. A Naive Bayes classifier is modeled as follows using Bayes theorem.

$$p(c_i^k|\boldsymbol{t}_i) \propto \prod_{l=1}^{D} p(\boldsymbol{t}_{il}|c_i^k)p(c_i^k)$$

Here, $p(\boldsymbol{t}_{il}|c_i^k)$ is the class conditional density and $p(c_i^k)$ is the prior distribution over the classes. The training involves estimating the parameters of the class conditional density from the data. The prior over the class can be uniform or based on class-ratio in the data. The estimated class conditional density is combined with the prior using Bayes theorem to make predictions for test data.

5.1.2 Support vector machines

Support vector machines (SVMs) are popular machine learning algorithms useful for binary classification [18]. But they can be easily extended to multi-class classification using one-vs-rest or one-vs-one strategy. SVMs learn a decision boundary which separates training instances from the two classes with the largest margin (large margin principle). When training instances are non-separable it tries to balance margin width and misclassification through a hyper-parameter (soft-margin classification). SVMs are characterized by Kernel functions which decides if the decision boundary is linear or non-linear. SVMs with a non-linear kernel can be seen as mapping training instances to higher dimensional space and learning a linear decision boundary in that space. This maps to a non-linear decision boundary in the original space. While, SVMs with linear Kernel learn a linear decision boundary. Linear SVM for binary classification (assuming $c_i \in \{+1, -1\}$) is formulated as follows.

$$\text{minimize}_{\boldsymbol{w},\mathrm{b},\xi}$$

$$\frac{1}{2}\boldsymbol{w}^\top \boldsymbol{w} + C\sum_{i=1}^{N} \xi_i$$

$$c_i(\boldsymbol{w}^\top \boldsymbol{t_i} + b) \geq 1 - \xi_i; \quad \xi_i \geq 0 \qquad \forall i = 1, \ldots, N$$

Here, the hyper-parameter C balances the margin width and misclassification error. Typically in SVMs, the minimization problem is solved as a maximization problem in the dual space. However, efficient algorithms exist to solve the problem in primal space for linear SVMs.

5.1.3 Decision trees

Decision tree is a supervised non-parametric machine learning algorithm which is simple and easily interpretable [19]. It constructs a tree where each node in the tree learns some decision rule. We use the classification

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197
MS Code : AEAM-D-17-00027
☐ LE ☑ CP ☐ TYPESET ☑ DISK

and regression tree (CART) algorithm to train the decision tree. The algorithm constructs a binary tree where each node in the tree splits the data recursively into two partitions based on some feature in the data and a threshold for that feature. The algorithms decide the best feature and threshold pair for each node based on an impurity score such as Gini, entropy etc. For instance, Gini impurity score of node i in the decision tree is defined as

$$G_i = 1 - \sum_{k=1}^{M} p_{ik}^2,$$

where p_{ik} denotes the proportion of class k instances in the ith node. The feature and threshold pair is chosen such that it minimizes the impurity score in the resulting split. This results in splits with majority of the samples belonging to one class. Decision trees are prone to over-fitting. By limiting the depth of the decision tree, this can be overcome to some extent. Predicting for test data involves traversing the decision tree to the leaf node and outputting the majority class in the leaf node.

5.1.4 Random forest

Random forest is an ensemble approach consisting of multiple decision trees each trained on a sub-sample of the data [20]. In addition to introducing randomness in the training data, it also adds randomness in feature selection. Instead of selecting the best feature among all the features for a split at a node, it only considers the best feature among the random subset of features. Random forest is found to achieve better generalization performance by avoiding over-fitting through model averaging. Prediction is done by choosing a class with majority of the voting from individual decision trees.

5.1.5 Adaboost

Adaboost is an ensemble classifier which iteratively trains a base classifier in order to overcome the errors made by the base classifier in the previous iterations [21]. Adaboost achieves this by maintaining a weight to each sample. The weights are updated after every iteration so that misclassified samples gets more weight than the correctly classified sample. This forces the base classifier to give more importance on misclassified samples during training. It combines the base classifiers learned over multiple iterations with a weight inversely proportional to their error rates to form an ensemble of classifiers. The weight α_j associated with a classifier j is defined as

$$\alpha_j = \eta log \frac{1 - r_j}{r_j}$$

where r_j is the weighted error rate computed from the weights associated with training data instances and η is the learning rate. The weights of misclassified instances are then updated using α_j. The ensemble of classifiers is then used for making predictions. The class which receives a majority of the weighted votes is taken as the predicted class. We considered a decision tree as the base learner.

5.1.6 Gradient boosting

Gradient boosting is another ensemble learner similar to Adaboost. It iteratively learns a base learner which corrects the error made by the previous learner in the ensemble [22]. Unlike Adaboost, it trains the base learner on the residual of errors made by the previous learner and does not maintain sample weights. We use regression trees as the base learner and are trained on the negative gradient of the multinomial deviance loss function.

The algorithms can be easily adapted to a multi-label classification setting during prediction. For instance, the probability values provided by Naive Bayes for each class will be useful for selecting the classes associated with a test instance. One could either select the top K classes according to probability values or keep a threshold to select the classes. Probability over the predicted classes can be obtained from decision tree as well by considering the ratio of instances belonging to that class in a leaf node. This is useful in obtain multiple labels, again following the same process as in a Naive Bayes classifier. One-vs-all approach was used for extending SVM to multi-class classification and it lends itself multiple labels for a test instance.

5.2 Features used

Performance of any classification algorithm depends heavily on the features used in representing the tweets. Here, we mention the different features that are used for representing the tweets in our work.

5.2.1 Tf–idf features

Tf–idf stands for Term Frequency—Inverse Document Frequency. It is the most common feature used in literature for text classification. In our experiments we mainly worked with unigrams and bigrams. Although we did some experiments with trigrams, no improvements over bigrams were found. We do not discuss about the trigram features here.

- *tf–idf (unigrams)*: This feature set is the simplest one which consists of tf–idf values of each term of the tweet texts.

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197 LE TYPESET
MS Code : AEAM-D-17-00027 CP DISK

Table 2 Tweet before and after pre-processing, belongs to classes 1 and 3 in our data set

Before	4 Tonne relief materials carrying food, medicines earth excavation equipments have been sent to Nepal from India http://t.co/4xdYcsEcrx
After	4 Tonne relief materials carrying food medicines earth excavation equipments sent nepal india urlurl

Table 3 BOIW and k-NN vectors with normalization for tweet in Table 2

Class	1	2	3	4	5	6	7
BOIW vector	1/15	1/15	2/15	0/15	0/15	0/15	0/15
k-NN vector	13/20	1/20	6/20	0/20	0/20	3/20	0/20

– *tf–idf (unigrams+ bigrams)* In this set, along with unigrams we have taken into account bigrams. After the addition of bigrams number of features almost increased by threefold.

5.2.2 Derived features

We call these features *derived*. As the name suggests we derive these features by applying some operations on the tweets. These features do not depend on any manual selection. Descriptions of the derived features are given below.

– *Bag of important words* BOIWs' are class specific set of words which represents each class uniquely. We used TF–IDF scores to calculate the importance of these set of words for a given class. Once the BOIWs were generated, actual feature was created for each tweet by taking the intersection of the words in the tweets with BOIWs. Although TF–IDF scores were separately used as feature, we believe the terms in these class-specific BOIWs will have high discriminative powers for each class, and are given separate attention. Given a tweet we count how many terms from the tweet are present in the BOIWs for each class. To normalize this count value we divide the count by k_1. In our experiments we used $k_1 = 15$. We selected this value for k_1 by varying the value of k_1 manually, starting from 10 and increased by multiple of 5 till 25. After 15 we didn't observe any significant change in classifier performance. An example of top 10 entries from BOIW for a class are shown in Table 5.
– *k-Nearest Neighbor votes* To generate this feature we selected top k_2 similar tweets from training set similar to each new tweet t_k using cosine similarity measure. If a closest tweet t_i belongs to class c_j then we may consider that t_i is voting for class c_j to be the target class for t_k. We count the number of votes given to each class by the k_2 selected tweets. We divided the vote counts by k_2 to normalize. In our experiments we found $k_2 = 20$ to be most appropriate by manual testing. Table 3 shows the vote counts for tweet in Table 2 without normalization. For this tweet, 13 out of 20 neighbors votes for this to be in class 1 *Resources available* and 6 neighbors votes for class 3 *Medical resources available*. Out of 20 neighbors 13 belongs to class 1, 1 belongs to class 2, 6 belongs to class 3 and 0 belongs to class 4, etc, the generated feature will be [13/20, 1/20, 6/20, 0/20, 0/20, 3/20, 0/20]. Please note that total vote count can cross k_2 because of multi-label nature of the tweets.
– *Length features* Lastly, we added length related features like number of characters with and without space in the tweet, count of words in the tweet.

5.2.3 Manual features

We created a set of manual features to augment our feature set. For each class we carefully selected a set of words that might be strong indicators for certain classes. We will call this feature set as *manual* in rest of the paper. Given below are the set of words and reasons for selecting them:

– *List of units* Presence of units like "litre, liter, kg, kilogram, gram, kilometer, meter, pack, packet, sets, ton, percentage, lac, lakh, million, thousand, hundred" signifies that the tweet might contain resource related information.
– *Availability related verbs* Words like "treat, send, sent, sending, supply, offer, distribute, treat, mobilize, mobilized, donate, donated, dispatch, dispatched" were selected as they represent verbs related availability.

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197
MS Code : AEAM-D-17-00027
☐ LE ☐ TYPESET
☑ CP ☑ DISK

Table 4 Manual feature vector for tweet in Table 2

Units	Available	Required	Medical	Location	URL	Phone
0	1	0	1	0	0	0

Table 5 Top 10 entries from BOIW of class 7 infrastructure damage and restoration reported in training set

Destroyed	Houses	Damaged	Building	Temples
Durbar	Monuments	Completely	Square	Collapsed

Table 6 Counts of manual features in **Training** set

Class number	Units	Available	Required	Medical	Location	URL	Phone
Class 1	135	178	17	121	24	18	15
Class 2	25	39	116	51	15	18	24
Class 3	34	98	7	242	24	8	8
Class 4	8	19	49	52	5	7	9
Class 5	12	23	14	34	12	4	3
Class 6	57	116	7	78	24	16	15
Class 7	7	10	9	6	19	10	10

Table 7 Counts of manual features in **Test** set

Class number	Units	Available	Required	Medical	Location	URL	Phone
Class 1	56	84	10	50	11	12	20
Class 2	6	14	46	18	2	4	4
Class 3	12	38	9	104	15	10	9
Class 4	1	12	17	27	3	1	2
Class 5	1	11	7	16	6	2	4
Class 6	29	49	6	47	9	6	2
Class 7	2	3	0	1	9	0	0

- *Requirement related verbs* This is a collection of words like "need, requirement, require, ran out, shortage, scarcity" which signify requirements.
- *Medical words* Presence of medical related words denotes high chance for the tweet to belong to medical related classes. We used words like "medicine, hospitals, medical, doctor, injection, syringe, ambulance, antibiotic."
- *Locational words* To address the location specific tweets keywords like "in, at" were chosen as it was seen that these words were followed by location names.

We also appended plural versions of these words in our word set. This processing was done before tweet preprocessing described in Sect. 6 as these bag of words contained stop-words like in, at etc. We created these features by counting the presence of above mentioned terms for each tweet. To normalize the values, we divided the count by number of words present in the tweet (Table 4).

Tables 6 and 7 shows the counts for each of the above mentioned manual features in the training and test sets respectively. It can be noted in the tables that counts in the test set follows the proportions of counts in training set.

5.2.4 Feature sets

We used the tf–idf features unigram and bigram as two base feature set and created 8 feature sets by using different combinations of derived features 5.2.2 and manual features 5.2.3. Each of the combinations is described below:

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197 LE TYPESET
MS Code : AEAM-D-17-00027 CP DISK

- FS_1 **(unigrams):** This feature set consists of tf–idf values of unigrams only.
- FS_2 **(unigrams + bigrams):** This feature set contains tf–idf values of unigrams along with bigrams.
- FS_3 **(unigrams + manual):** To create this feature set we combined unigrams FS_1 with manually created features.
- FS_4 **(unigrams + bigrams + manual):** In this set, we added unigrams and bigrams FS_2 with manual features.
- FS_5 **(unigrams + derived):** Combination of unigrams FS_1 with derived features creates this set.
- FS_6 **(unigrams + bigrams + derived):** We combined unigrams and bigrams FS_2 with derived features to create this set.
- FS_7 **(unigrams + derived + manual):** We added derived features with FS_3 for this set.
- FS_8 **(unigrams + bigrams + derived + manual):** Derived features along with FS_4 creates this set.

Next section discusses about the setup we used for our experiments.

6 Experimental setup

We divided the dataset described in Sect. 3 into train and test sets with 70% and 30% of the total data points respectively. Table 8 shows the counts of data points we used for Train and Test set. Our code can be found at https://github.com/samiith/tweet_classification.git.

6.1 Preprocessing

Before working with the data we cleaned the data by performing the below-mentioned steps in sequence.

1. *Acronym expansion*: Tweets are informal and people use various acronyms. In this step we parse each tweet and try to find a match with our acronym dictionary. We used the acronym given in [23] and merged it with our own generated acronyms.
2. *Removal of smiley and non-ASCII characters*: Another prevalent problem with informal text is with smiles, we removed all smiley and non-ASCII characters.
3. *Case folding*: we converted all the texts to lower case.
4. *Stop-words and punctuation removal* After all the above mentioned steps were done we removed any word from the tweet which is present in the nltk stopwords[1].
5. *Special character removal*: We removed special characters like '#', '@' without removing the corresponding hashtags or user mentions. Also, there are some special words like "rt" which means the tweet is actually a retweet, "via" and "amp" though this is not a stop-word but it contains no value whatsoever. We removed all the above mentioned words.
6. *URLs and Phone numbers handling* : URLs' or phone numbers present in any tweet was replaced by "urlurl" and "phonenumber" respectively.

Below we mention a tweet in original form and after preprocessing was done:

Table 8 Counts of tweets for each class

Class number	Train data count	Test data count
Class 1	401	175
Class 2	210	81
Class 3	231	100
Class 4	75	36
Class 5	135	53
Class 6	252	119
Class 7	178	74
Total	1482	638

6.2 Algorithm setup

We have used the algorithms mentioned in Sect. 5 for this task. Naive Bayes classifier was used considering the likelihood of features as Gaussian with prior based on class-ratio. For Support vector machines we used linear kernel with hyper-parameter $C = 1$ in our experiments since they are found to be most useful for classification of text data with high dimensional features. One-vs-rest[2] strategy was followed to handle multi-class classification problem whenever required. In case of decision tree, Gini impurity score was used for each node without any tree depth with minimum sample in leaf nodes as 2. The tree will be expanded until all nodes are pure. Our random forest classifier works with 10 decision trees with each tree as same configuration as above mentioned Decision Tree classifier. Adaboost classifier uses decision tree as its base estimator using "SAMME" algorithm and learning rate = 1. A limit of maximum 50 estimator was put in place. "Deviance" loss function was used for gradient boosting classifier with regression tree weak learners using "Friedman mean squared error" measure (Table 2).

[1] https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/packages/corpora/stopwords.zip

[2] http://scikit-learn.org/stable/modules/multiclass.html#one-vs-the-rest.

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197
MS Code : AEAM-D-17-00027
LE TYPESET
CP DISK

6.3 Metrics used

We used Precision, Recall, F_1, Macro-F_1 and Micro-F_1 as evaluation metrics to evaluate the performances of the algorithms. The measures Precision, Recall, F_1 can be used to evaluate the performances of the classifiers on individual classes. Macro-F_1 and Micro-F_1 are used to evaluate the performance of the classifier for the entire data, with test instances from all the classes considered together. Definitions of these evaluation measures are given below.

- *Precision*: For a given class, out of all the tweets that are assigned that class label, what fraction of them actually belong to that class, is called the precision. If *Actual* denotes the set of tweets that belong to that class and *Predicted* denotes the set of tweets that are assigned to that class by the algorithm, then Precision for the class can be computed as:

$$Precision = \frac{|Actual \cap Predicted|}{|Predicted|}$$

- *Recall*: For a given class, out of all the tweets that are from that class, what fraction of them are predicted to belong to that same class is called the Recall. If *Actual* denotes the set of tweets that belong to that class and *Predicted* denotes the set of tweets that are assigned to that class by the algorithm, then Recall for the class can be computed as:

$$Recall = \frac{|Actual \cap Predicted|}{|Actual|}$$

- F_1 *score*: F_1 score for a class is the harmonic mean of Precision and Recall obtained for the class. F-measure for the class can be computed as:

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

- *Macro-F_1* **score**: In Macro-F_1 we take summation of F_1 scores of all the classes together and divide by the number of classes. Thus giving equal weights to each class. It gives the performance measure of the model across all classes. Macro-F_1 for binary classification can be computed as:

$$Macro - F_1 = \frac{1}{q}\sum_{j=1}^{q} F_1(TP_j, FP_j, TN_j, FN_j)$$

where the TP_j, FP_j, TN_j, FN_j are the true positive, false positive, true negative and false negative counts respectively for only the j^{th} label. In a multi-label classification scheme, there is no concept of in-class versus out-of-class. In this case, Macro-F_1 is computed by treating each label as a binary classification scheme, then computing its F_1, and then average over all labels.

- *Micro-F_1* **score**: In Micro-F_1 we calculate the F_1 for individual classes and take their summation. Micro-F_1 favors classes with more data points. Micro-F_1 for binary classification can be computer as:

$$Micro - F_1 = F_1\left(\sum_{j=1}^{q} TP_j, \sum_{j=1}^{q} FP_j, \sum_{j=1}^{q} TN_j, \sum_{j=1}^{q} FN_j\right)$$

where the TP_j, FP_j, TN_j, FN_j are the true positive, false positive, true negative and false negative counts respectively for only the j^{th} label. For Micro-F_1, we make M binary predictions (where M is the number of labels) for each of the N data points. We then compute F_1 over those $M * N$ predictions.

In next section, we show our results and discuss what were the factors for such result.

7 Results

In this section, we discuss the results of our experiments with different choices of feature sets and algorithms.

7.1 Results for the complete collection: all classes considered together

We used Macro-F_1 and Micro-F_1 scores to determine the overall performance of each algorithm and feature set combination for the complete test dataset. The values for these measures are shown in Tables 9 and 10 respectively. It can be observed from the tables that SVM Linear with Feature set FS_7 (unigrams + derived + manual) outperforms all other feature set and algorithm combinations. SVM Linear with feature set FS_3 (unigrams + manual) and FS_6 (unigrams + bigrams + derived) appear as close competitors. Although for Macro-F_1, feature set FS_3 perform equally well as FS_6 (unigrams + bigrams + derived), it achieves slightly higher score than FS_3 according to Micro-F_1 score.

7.1.1 Comments about the feature sets

As mentioned in Sect. 5.2, the performance of the classification algorithms are generally found to depend a lot on the features used. In our work, along with tf–idf based features, we have used two different groups of features, namely, derived features and manual features. We analyze

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197 LE TYPESET
MS Code : AEAM-D-17-00027 CP DISK

Table 9 Macro-F_1 scores

Algorithms	FS_1	FS_2	FS_3	FS_4	FS_5	FS_6	FS_7	FS_8
Adaboost	0.6293	0.6020	0.6391	0.5955	0.6306	0.6312	0.6304	0.6073
Decision tree	0.5712	0.5789	0.6030	0.5899	0.5712	0.5869	0.6064	0.5770
Gradient boosting	0.6025	0.6166	0.6016	0.6073	0.5987	0.6189	0.6040	0.6087
NB Gaussian	0.4995	0.5477	0.4995	0.5477	0.4995	0.5477	0.4995	0.5477
Random forest	0.5180	0.4430	0.5704	0.5145	0.4981	0.4266	0.5258	0.4355
SVM Linear	0.6547	0.6425	0.6551	0.6443	0.6601	0.6551	**0.6625**	0.6528

Table 10 Micro-F_1 scores

Algorithms	FS_1	FS_2	FS_3	FS_4	FS_5	FS_6	FS_7	FS_8
Adaboost	0.6638	0.6477	0.6744	0.6466	0.6722	0.6677	0.6600	0.6439
Decision tree	0.6096	0.6099	0.6308	0.6095	0.6108	0.6163	0.6294	0.6048
Gradient boosting	0.6389	0.6552	0.6456	0.6490	0.6457	0.6588	0.6445	0.6556
NB Gaussian	0.5327	0.5878	0.5327	0.5878	0.5327	0.5878	0.5327	0.5878
Random forest	0.5838	0.5243	0.6458	0.6068	0.5824	0.5105	0.6281	0.5228
SVM Linear	0.7011	0.6939	0.7055	0.6964	0.7081	0.7081	**0.7089**	0.7047

the experimental results thoroughly to see the usefulness of these groups of features.

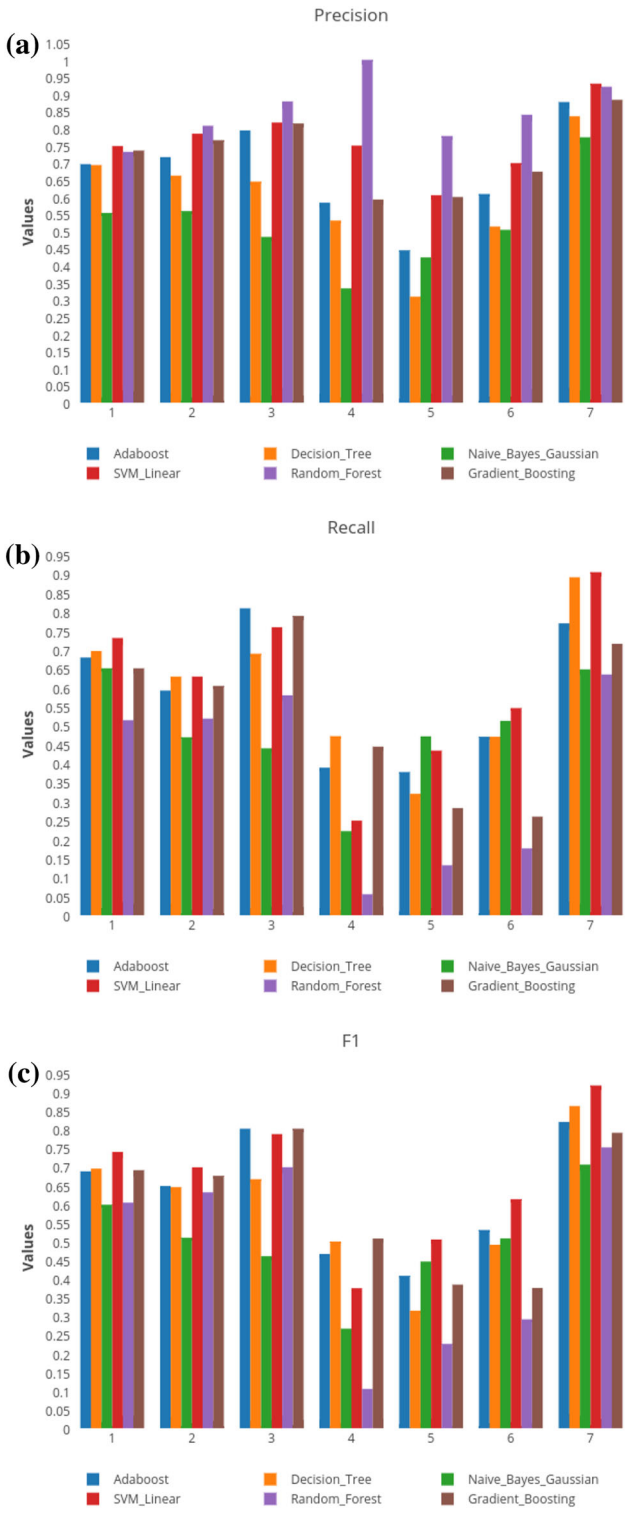
The feature sets FS_1 (unigrams), FS_2 (unigrams + bigrams), FS_5 (unigrams + derived) and FS_6 (unigrams + bigrams + derived) do not have manual features in them. If we add the manual features with each of these feature sets, we get the feature sets FS_3 (unigrams + manual), FS_4 (unigrams + bigrams + manual), FS_7 (unigrams + derived + manual) and FS_8 (unigrams + bigrams + derived + manual) respectively. If we refer to the Macro-F_1 and Micro-F_1 scores again, we see that the performance generally improves whenever we add the manual features. If we compare FS_6 (unigrams + bigrams + derived) with FS_8 (bigram + derived + manual), the performance improves in around 50% of the cases. However, use of FS_3 leads to better scores than FS_1 for *almost* all the algorithms. Improvements in almost all algorithms are observed when we use FS_4 instead of FS_2, and also if we use FS_7 vs FS_4. This observation regarding manual features seems to indicate the usefulness of the manual features considered in our work.

Similarly, the feature sets FS_1 (unigrams), FS_2 (unigrams + bigrams), FS_3 (unigrams + manual) and FS_4 (unigrams + bigrams + manual) do not have derived features in them. If we add derived features to each of these feature sets, we get the feature sets FS_5 (unigrams + derived), FS_6 (unigrams + bigrams + derived), FS_7 (unigrams + derived + manual) and FS_8 (unigrams + bigrams + derived + manual) respectively. Again we refer to Tables 9 and 10 to understand the usefulness of these derived features. We see that, although in majority of the cases the feature set with derived features is working better than its counterpart without the derived features, the differences are not high. If we compare the values in Micro-F_1 table, we see that FS_5 performs better than FS_1 for 4 of the 6 algorithms. For random forest, both the feature sets lead to same result. Similarly, results with FS_6 are better or same as the results with FS_2 for 5 of the 6 algorithms. For FS_3 vs FS_7 and FS_4 vs FS_8, we see that the improvements are not always there, and even if there is improvement, it is not much. This seems to indicate that if manual features are present, the derived features are generally not able to provide much boost to the performances of the algorithms. This is kind of expected, as the manual features are added keeping human expertise and intuition in the loop, and derived features (e.g. bag of important words, k-Nearest Neighbor votes) rely a lot on the statistics of the data. Hence derived features are supposed to be noisy for smaller dataset. However, the performance improvement obtained in around 50% of the cases indicate that derived features might be useful for the task if a larger dataset is available.

7.1.2 *Comments on the performance of individual classifiers*

In terms of the Micro-F_1 and Macro-F_1 scores, SVM Linear consistently outperformed all other algorithms for almost all the cases. The Precision, Recall and F_1 scores for

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197
MS Code : AEAM-D-17-00027
☐ LE ☑ CP ☐ TYPESET ☑ DISK



◀**Fig. 1** Performance of the different algorithms for different classes, while using Feature Set FS_7 (unigram + manual + derived). In the individual figures, each stack of bars is for a particular class. Class 1 is Resources available, Class 3 is Resources required, Class 3 is Medical resources available, Class 4 is Medical resources required, Class 5 is Requirements/Availability of resources at specific locations, Class 6 is Activities of various NGOs/Government organizations, Class 7 is Infrastructure damage and restoration reported **a** Precision scores obtained by different classes for Feature set FS_7 **b** Recall scores obtained by different classes for Feature set FS_7 **c** F_1 scores obtained by different classes for Feature set FS_7

all the classes for the feature set FS_7 is shown in Fig. 1a–c respectively. From the graphs, it is clear that random forest has highest Precision for all the classes. However, it has very poor Recall, for all the classes. It indicates that random forest is not able to put a tweet in appropriate class most of the times. On other hand decision tree Classifier performs reasonably well. Random forest is an ensemble of multiple decision trees. Each of these underlying decision trees (base learners) have a different sample of the original dataset. It appears that the classification function learned by these individual decision trees suffer from the sparseness of the data, and when a combination of the individual trees' decisions are taken into account for the final decision of the test tweet's class, the random forest classifier is getting confused.

On the other hand, SVM performs well consistently. For almost all the classes, it achieves a second-best precision (after Random Forest) and also has high recall for all the classes. This might be attributed to the generalization property of SVM. All other methods show moderate performance throughout, for all the classes and all evaluation metrics.

We also notice that Naive Bayes algorithm has the least impact of adding new features. A significant F_1 score improvement is there from unigram to unigram + bigram features. However addition of manual features and derived features had no extra effect over unigrams + bigrams.

7.2 Results for individual classes

The F_1 scores obtained by the different algorithms for the classes 1 to 7 are shown in Tables 11, 12, 13, 14, 15, 16, 17 respectively. We summarize this information in Table 18 to indicate the algorithm and feature set combination that obtained best value of according to the F_1 score.

It can be observed from Table 18 that SVM Linear outperforms all other algorithms for all classes except class 4. For classes 4, 6 and 7 SVM performed best with feature set FS_7 which is a combination of tf–idf with manual and derived features. decision tree crosses SVM by large margin for class 4, possibly due to lack of sufficient of training examples. Class 4 has the lowest number of training examples (75). Due to the less number of training data, all algorithms perform poorly for this class.

In next section we conclude our findings and how our approach can be modified in future to achieve better result.

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197
MS Code : AEAM-D-17-00027
LE TYPESET
CP DISK

Table 11 F_1 scores for class 1: resources available

Algorithms	FS_1	FS_2	FS_3	FS_4	FS_5	FS_6	FS_7	FS_8
Adaboost	0.6951	0.6786	0.6878	0.6745	0.7118	0.6938	0.6878	0.6492
Decision tree	0.6363	0.6594	0.6927	0.6480	0.6571	0.6666	0.6908	0.6426
Gradient boosting	0.6940	0.6932	0.6932	0.6845	0.7062	0.6909	0.7030	0.6936
NB Gaussian	0.5984	0.6318	0.5984	0.6318	0.5984	0.6318	0.5984	0.6318
Random Forest	0.6644	0.5953	0.6821	0.6495	0.6558	0.6298	0.6840	0.5866
SVM Linear	0.7405	0.7289	0.7383	0.7305	**0.7500**	0.7433	0.7398	0.7383

Table 12 F_1 scores for class 2: resources required

Algorithms	FS_1	FS_2	FS_3	FS_4	FS_5	FS_6	FS_7	FS_8
Adaboost	0.6760	0.6962	0.7066	0.7320	0.6760	0.6962	0.6486	0.7320
Decision tree	0.6193	0.5766	0.6625	0.6265	0.6538	0.6540	0.6666	0.6097
Gradient boosting	0.6891	0.7083	0.6938	0.6928	0.6573	0.7034	0.6712	0.7006
NB Gaussian	0.5100	0.5507	0.5100	0.5507	0.5100	0.5507	0.5100	0.5507
Random forest	0.6461	0.5217	0.7297	0.6814	0.5299	0.5203	0.6515	0.5641
SVM Linear	0.6986	0.7123	0.7074	**0.7466**	0.6849	0.7123	0.6986	0.7432

Table 13 F_1 scores for class 3: medical resources available

Algorithms	FS_1	FS_2	FS_3	FS_4	FS_5	FS_6	FS_7	FS_8
Adaboost	0.8020	0.7826	0.7900	0.8061	0.8121	0.7826	0.8059	0.7821
Decision tree	0.7342	0.7511	0.6969	0.7537	0.7272	0.7393	0.7029	0.7326
Gradient boosting	0.7650	0.7708	0.7817	0.7860	0.7826	0.7812	0.7860	0.7821
NB Gaussian	0.4607	0.5833	0.4607	0.5833	0.4607	0.5833	0.4607	0.5833
Random forest	0.6707	0.6233	0.7441	0.7708	0.7058	0.5827	0.7978	0.6745
SVM Linear	0.7724	0.8108	0.7789	0.7916	0.7830	**0.8235**	0.7875	0.7916

Table 14 F_1 scores for class 4: *Medical resources required*

Algorithms	FS_1	FS_2	FS_3	FS_4	FS_5	FS_6	FS_7	FS_8
Adaboost	0.5263	0.4210	0.4482	0.3396	0.4062	0.4545	0.4912	0.4126
Decision tree	0.3157	0.3928	0.5151	0.5230	0.3225	0.3846	**0.5588**	0.4657
Gradient boosting	0.5090	0.4705	0.4333	0.4642	0.4363	0.4615	0.5161	0.4642
NB Gaussian	0.2666	0.2978	0.2666	0.2978	0.2666	0.2978	0.2666	0.2978
Random forest	0.1538	0.0540	0.1999	0.1052	0.1538	0.0540	0.0540	0.0200
SVM Linear	0.3750	0.3478	0.3404	0.3404	0.3750	0.3478	0.3750	0.3404

Table 15 F_1 scores for class 5: *Requirements / availability of resources at specific locations*

Algorithms	FS_1	FS_2	FS_3	FS_4	FS_5	FS_6	FS_7	FS_8
Adaboost	0.3555	0.3544	0.4166	0.3529	0.4395	0.3863	0.4200	0.3516
Decision tree	0.3396	0.4561	0.3333	0.3275	0.3238	0.4210	0.3269	0.3333
Gradient boosting	0.3846	0.3947	0.3846	0.3684	0.3589	0.3947	0.3684	0.3243
NB Gaussian	0.4464	0.4666	0.4464	0.4666	0.4464	0.4666	0.4464	0.4666
Random forest	0.3943	0.2222	0.3076	0.2258	0.2258	0.2461	0.2295	0.1875
SVM Linear	**0.5057**	0.4691	**0.5057**	0.4634	0.5054	0.4938	0.5054	0.4938

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197
MS Code : AEAM-D-17-00027
☐ LE ☑ CP ☐ TYPESET ☑ DISK

Table 16 F_1 scores for class 6: *Activities of various NGOs / government organizations*

Algorithms	FS_1	FS_2	FS_3	FS_4	FS_5	FS_6	FS_7	FS_8
Adaboost	0.5100	0.4088	0.5463	0.4086	0.5151	0.5388	0.5308	0.4536
Decision tree	0.4810	0.3465	0.4818	0.3823	0.4588	0.3671	0.4669	0.3723
Gradient boosting	0.3558	0.4505	0.4071	0.4293	0.4497	0.4835	0.3832	0.4640
NB Gaussian	0.5083	0.5480	0.5083	0.5480	0.5083	0.5480	0.5083	0.5480
Random forest	0.3225	0.2448	0.4634	0.3116	0.3647	0.2638	0.4000	0.2820
SVM Linear	0.5882	0.5492	0.6048	0.5583	0.6124	0.5940	**0.6132**	0.5911

Table 17 F_1 scores for class 7: *Infrastructure damage and restoration reported*

Algorithms	FS_1	FS_2	FS_3	FS_4	FS_5	FS_6	FS_7	FS_8
Adaboost	0.8400	0.8724	0.8783	0.8552	0.8533	0.8666	0.8285	0.8701
Decision tree	0.8724	0.8701	0.8387	0.8684	0.8552	0.8758	0.8322	0.8831
Gradient boosting	0.8201	0.8285	0.8175	0.8260	0.7999	0.8175	0.7999	0.8321
NB Gaussian	0.7058	0.7555	0.7058	0.7555	0.7058	0.7555	0.7058	0.7555
Random forest	0.7741	0.8396	0.8656	0.8571	0.8507	0.6896	0.8636	0.7540
SVM Linear	0.9027	0.8794	0.9103	0.8794	0.9103	0.8714	**0.9178**	0.8714

Table 18 List of best algorithm and feature set for each class

Class number	Best performance (Algorithm + Feature set)
Class 1	SVM linear + FS_5 (unigrams + derived)
Class 2	SVM linear + FS_4 (unigrams + bigrams + manual)
Class 3	SVM linear + FS_6(unigrams + bigrams + derived)
Class 4	Decision tree + FS_7 (unigrams + manual + derived)
Class 5	SVM linear + FS_1 (unigrams) and FS_3 (unigrams + manual)
Class 6	SVM linear + FS_7 (unigrams + manual + derived)
Class 7	SVM linear + FS_7 (unigrams + manual + derived)

8 Conclusion and future work

In this paper we discussed our approach of classifying disaster related tweets according to the insights they might contain. It is evident that the derived and manual features proposed in the work help the algorithms in achieving better performances over traditional term frequency related features. We have also showed through detailed analysis that SVM classifier performs much better than other off-the-shelf classifiers like Naive Bayes, random forest, decision tree, Adaboost and gradient boosting.

In future we plan to use Deep Learning based techniques which can eliminate feature engineering. There is need to cater to specific class related situations like one class can be a proper subset of other class. One such example can be seen in our dataset as in case of *Resources required* and *Medical resources required*. It is clear that if a tweet belongs to the latter must also belong to former but not vice versa. It might be interesting to see the performance of the classification if we consider possible hierarchical relationships between the classes.

If the tweets for specific insight categories can be identified, then one can summarize the tweets to prepare a short summary of the situation and activities being performed in the affected regions. It would be interesting to see whether we can monitor the progress of rescue and relief operations using a time-line of tweets.

References

1. How social media is changing disaster response. https://www.scientificamerican.com/article/how-social-media-is-changing-disaster-response/
2. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes Twitter users: real-time event detection by social sensors. In: Proceedings of the 19th International Conference on World Wide Web, pp. 851–860. ACM (2010)
3. Olteanu, A., Castillo, C., Diaz, F., Vieweg, S.: CrisisLex: A Lexicon for Collecting and Filtering micro-blogged Communications in Crises. Association for the Advancement of Artificial Intelligence, Menlo park (2014)
4. Basu, M., Roy, A., Ghosh, K., Bandyopadhyay, S., Ghosh, S.: Micro-blog retrieval in a disaster situation: a new test collection for evaluation. SMERP, ECIR (2017)
5. Singla, R., Modha, S., Majumder, P., Mandalia, C.: Information Extraction from micro-blog for disaster related event. SMERP, ECIR (2017)

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197
MS Code : AEAM-D-17-00027
☐ LE ☑ CP ☐ TYPESET ☑ DISK

6. Soni, R., Pal, S.: Micro-blog retrieval for disaster relief: how to create ground truths?. SMERP, ECIR (2017)
7. Stowe, K., Paul, M., Palmer, M., Palen, L., Anderson, K.: Identifying and categorizing disaster-related tweets. In: Proceedings of The Fourth International Workshop on Natural Language Processing for Social Media, Austin, TX, November 1 (2016)
8. Caragea, C., McNeese, N., Jaiswal, A., Traylor, G., Kim, H.-W., Mitra, P., Wu, D., Tapia, A.H., Giles, L., Jansen, B.J., Yen, J.: Classifying text messages for the Haiti earthquake. In Proceedings of the 8th International ISCRAM Conference, Lisbon, Portugal (2011)
9. Munro R., Manning C.: Subword variation in text message classification. Paper presented at the Human Language Technologies: The Annual Conference of the North American Chapter of the ACL (2010)
10. Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., Demirbas, M.: Short text classification in Twitter to improve information filtering. In: SIGIR 10, Geneva, Switzerland, July 1923 (2010)
11. Yuan, Q., Cong, G., Thalmann, N.M.: Enhancing naive bayes with various smoothing methods for short text classification. In: Proceedings of the 21st International Conference Companion on World Wide Web- WWW '12 Companion, pp. 645–646 (2012)
12. Thelwall, M., Buckley, K., Paltoglou, G., Cai, D., Kappas, A.: Sentiment strength detection in short informal text. J. Am. Soc. Inf. Sci. Technol. **61**(12), 2544–2558 (2010)
13. Yin, C., Xiang, J., Zhang, H., Wang, J., Yin, Z., Kim, J. U.: A New SVM method for short text classification based on semi-supervised learning. In: 2015 4th International Conference on Advanced Information Technology and Sensor Application (AITS), Harbin, pp. 100–103 (2015)
14. Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., Demirbas, M.: Short text classification in Twitter to improve information filtering. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 841-842. ACM (2010)
15. Ghosh, S., Ghosh, K.: Overview of the FIRE 2016 micro-blog track: information extraction from micro-blogs posted during disasters. In: Working notes of FIRE 2016-Forum for Information Retrieval Evaluation, Kolkata, India, December 7–10, 2016, CEUR Workshop Proceedings. CEUR-WS.org (2016)
16. Tsoumakas, Grigorios, Katakis, Ioannis: Multi-label classification: an overview. Int. J. Data Warehous. Min. **3**(3), 1–13 (2007)
17. Ng, A.Y., Jordan, M.I.: On discriminative vs. generative classifiers: a comparison of logistic regression and naive Bayes. NIPS (2002)
18. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
19. Breiman, L.: Classification and Regression Trees, The Wadsworth and Brooks–Cole Statistics—Probability Series. Chapman & Hall, London (1984)
20. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
21. Schapire, R., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Mach. Learn. **37**, 297–336 (1999)
22. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. **29**, 1189–1232 (2000)
23. Imran, M., Mitra, P., Castillo, C.: Twitter as a lifeline: human-annotated Twitter corpora for NLP of crisis-related messages. In: Proceedings of the 10th Language Resources and Evaluation Conference (LREC), pp. 1638-1643. Portoro, Slovenia (2016)

Journal : Large_Springer-India12572 Dispatch : 21-11-2017 Pages : 14
Article No. : 197 ☐ LE ☐ TYPESET
MS Code : AEAM-D-17-00027 ☑ CP ☑ DISK