Overview of the FIRE 2017 track: Information Retrieval from Microblogs during Disasters (IRMiDis)

Moumita Basu
UEM Kolkata, India; IIEST Shibpur, India

Saptarshi Ghosh
IIT Kharagpur, India; IIEST Shibpur, India

Kripabandhu Ghosh
IIT Kanpur, India

Monojit Choudhury
Microsoft Research, India

ABSTRACT

The FIRE 2017 Information Retrieval from Microblogs during Disasters (IRMiDis) track focused on retrieval and matching of needs and availabilities of resources from microblogs posted on Twitter during disaster events. A dataset of around 67,000 microblogs (tweets) in English as well as in local languages such as Hindi and Nepali, posted during the Nepal earthquake in April 2015, was made available to the participants. There were two tasks. The first task (Task1) was to retrieve tweets that inform about needs and availabilities of resources; these tweets are called need-tweets and availability-tweets. The second task (Task2) was to match need-tweets with appropriate availability-tweets.

CCS CONCEPTS

•Information systems →Query reformulation;

1 INTRODUCTION

Various important information is posted on online social media like Twitter at the times of disaster events such as floods and earthquakes. However, this important information is immersed within a lot of conversational content such as prayers and sympathy for the victims. Hence automated methodologies are needed to extract the important information from the deluge of tweets posted during such an event [3]. In this track, we focused on two types of tweets that are very important for coordinating relief operations in a disaster situation:
(1) **Need-tweets:** Tweets which inform about the need or requirement of some specific resources such as food, water, medical aid, shelter, mobile or Internet connectivity, etc.
(2) **Availability-tweets:** Tweets which inform about the availability of some specific resources. This class includes both tweets which inform about potential availability, such as resources being transported or despatched to the disaster-struck area, as well as tweets informing about the actual availability in the disaster-struck area, such as food being distributed, etc.

The track had two tasks, as described below.
Task 1: Identifying need-tweets and availability-tweets: Here the participants were asked to develop methodologies for identifying need-tweets and availability-tweets. Note that this task can be approached in different ways. It can be approached as a retrieval or search problem, where two types of tweets are to be retrieved. Differently, the problem of identifying need-tweets and availability-tweets can also be viewed as a classification problem, e.g., where tweets are classified into three classes - need-tweets, availability-tweets, and others.
Task 2: Matching need-tweets and availability-tweets: An availability-tweet is said to match a need-tweet, if the availability-tweet informs about the availability of at least one resource whose need is indicated in the need-tweet. In this task, the participants were asked to develop methodologies for matching need-tweets with appropriate availability-tweets.

Table 1 shows some examples of need-tweets and availability-tweets from the dataset that was made available to the participants (described in the next section). Note that the dataset contains tweets not only in English but also in local languages such as Hindi and Nepali, and also code-mixed tweets, as shown in Table 1.

2 THE TEST COLLECTION

In this track, our objective was to develop a test collection containing code-mixed microblogs for evaluating

- Methodologies for extracting two specific type of actionable situational information – needs and availabilities of various types of resources (need-tweets and availability-tweets), and
- Methodologies for matching need-tweets and availability-tweets

In this section, we describe how the test collection for both the tasks of IRMiDis track was developed.

2.1 Tweet dataset

As part of the same track in FIRE 2016, we had released a collection of $50,018$ English tweets related to the devastating earthquake that occurred in Nepal and parts of India on 25^{th} April 2015[1] [2]. We also utilized this collection to evaluate several IR methodologies developed by ourselves and others [1, 2]. We re-use these tweets in the present track. Additionally, in the present track, we collected tweets in Hindi and Nepali (based on language identification by Twitter itself) using the Twitter Search API [4], using the keyword 'नेपाल', that were posted during the same period that of the English tweets. A total 90K tweets were collected, and after removing duplicates and near-duplicates as before [1, 2], we obtained a set of *16,903 tweets*. Hence, a set of *66,921 tweets* tweets was obtained – containing $50,018$ English tweets and $16,903$ tweets in Hindi, Nepali or code-mixed tweets – which was used as the test collection for the track.

[1]https://en.wikipedia.org/wiki/April_2015_Nepal_earthquake

Examples of need-tweets	Examples of availability-tweets
नुनाकोट जिल्ला थानसिंग गाविसमा अहिलेसम्म कुनै राहत सामाग्री तथा उद्दारटोली नपुगेको खबरले दुखी बनायो,तेतातिर पनि सम्बन्धित पक्ष…	Nepal earthquake: Spiritual group sends relief materials to victims [url]
नेपाल में दवाओं की किल्लत, एयरपोर्ट पर हजारों की भीड़ - आज तक #World [url]	स्वास्थ्य मन्त्रालय र WHO सँगको संयोजनमा करीब छ दर्जन चलचित्रकर्मीहरु औषधी र खाद्यन्न वितरण तथा जनचेतना कार्यक्रममा #earthquake #Nepalifilms
after 7days of earthquake! people are still crying, sleeping in rain, lack of food and water! hope it was dream but this all happens to us!	RT @abpnewshindi: विमान में खाना, पानी और कंबल नेपाल के लिए भेजे गए हैं . एस. जयशंकर #NepalEarthquake लाइव देखें- [url]
Nepal earthquake: Homeless urgently need tents; Death toll above 5,200 Read More... [url]	#grgadventure donating our tents and sleepig bags for victims of the #nepal #earthquake [url]

Table 1: Examples of need-tweets and matching availability-tweets, posted during the 2015 Nepal earthquake

Topic for Retrieval	**Hindi tweets**	**Nepali tweets**	**English tweets**
Need-Tweets	31	82	558
Availability-Tweets	238	206	1326

Table 2: Summary of the gold standard used in IRMiDis

The data was ordered chronologically based on the timestamp assigned by Twitter, and *released in two stages*. At the start of the track, the chronologically earlier posted 20K tweets were released (training set), along with a sample of Need-tweets and Availability-tweets in these 20K tweets (development set). The participating teams were expected to use the training and development sets to formulate their methodologies. Next, about two weeks before the submission of results, the set of chronologically later posted 46K tweets were released (test set). The methodologies were evaluated based on their performance over the test set.

2.2 Developing gold standard for retrieval

The gold-standard for both tasks was generated by 'manual runs'. To develop the gold standard set of need-tweets and availability-tweets, a set of three human annotators having proficiency in English, Hindi and Nepali were involved. Additionally, annotators were a regular user of Twitter, and had previous experience of working with social media content posted during disasters. The gold standard development involved similar three phases as described in [1, 2] – first each annotator individually retrieved need-tweets and availability-tweets, then there was mutual discussion among the annotators to resolve conflicts, and finally there was a pooling step over all the runs submitted to the track.

The summary of the number of need-tweets and availability-tweets present in the final gold standard corresponding to three different languages is reported in Table 2.

2.3 Developing gold standard for Matching

To develop the gold standard for matching, the same human annotators were involved. The annotators were asked to inspect the gold standard for need-tweets and availability-tweets, and to manually find out the set of need-tweets for which at least one matching availability-tweet exists. The annotators were also asked to find matching availability-tweets for each need-tweet. Additionally, pooling was used over the participant runs to identify relevant matches which the annotators might not have found.

3 TASK 1: IDENTIFYING NEED-TWEETS AND AVAILABILITY-TWEETS

11 teams have participated in Task1 and 18 runs were submitted. A summary of the methodologies used by each team is given in the next sub-section.

3.1 Methodologies

We now summarize the methodologies adopted in the submitted runs.

- **iitbhu_fmt17**: This team participated from Indian Institute of Technology (BHU) Varanasi, India. It submitted the following two *Automatic* (i.e. no manual step involved) runs. Both the runs used google translator API to convert the code-mixed tweets.
  - *iitbhu_fmt17_task1_1*: It used Apache Lucene, a open source Java-based text search engine library[2]. Training data was indexed using Standard Analyzer and frequency of each token is training set recorded. Query is generated by the disjunction of tokens with frequency more than or equal to a threshold value. Tweets are categorized according to the score return by Lucene search engine.
  - *iitbhu_fmt17_task1_2*: It treated the task as a classification task, and used SVM algorithm. Undersampling was employed. A threshold of 0.2 in the predicted score by the SVM classifier was set to classify a tweet as relevant.
- **DataBros**: This team participated from Indian Institute of Information Technology, Kalyani, India. It submitted one *automatic* run described below:
  - *iiests_IRMiDis_FIRE2017_1*: The bag of words model was used with TfidfVectorizer to collect the features including unigram and bigrams. Recursive Feature Elimination (RFE) algorithm with LinearSVM was used

[2]https://lucene.apache.org/

to compute the ranking weights for all features and sort the features according to weight vectors. In addition, Decision Tree Classifier is applied to classify the data.

- **Bits_Pilani_WiSoc**: This team participated from Birla Institute of Technology and Science, Pilani, India. It submitted two *automatic* runs. Both the runs were generated by using word embeddings and then fastText classification algorithm to classify the tweet to its appropriate category. The fastText classifier was trained on the labeled data and the previously created word embeddings.
  - *BITS_PILANI_RUN1*: Created word embeddings using Skip-gram model.
  - *BITS_PILANI_RUN2*: Created word embeddings using CBOW model.
- **Data Engineering Group**: This team participated from Indraprastha Institute of Information Technology, Delhi, India. It submitted one *automatic* run – *DataEngineeringGroup_1* described as follows:
  - *DataEngineeringGroup_1*: This run used Stanford CoreNLP library [3] for the POS tagging along with the lemma identification of all the words in the tweet set. Features were constructed using both the words present in the tweets and its POS tag. Logistic Regression model was used for this classification task.
- **DIA Lab - NITK**: This team participated from, National Institute of Technology, Karnataka, India. It submitted one *automatic* run described as follows:
  - *daiict_irlab_1*: This run used Doc2vec model to transform tweets into embedding vectors of size 100. To convert the code-mixed tweets, the ASCII transliterations of unicode text (tweet) was used. The frequency of each token available in a tweet is also used as the feature. These embeddings was the input for multilayer preceptron (a feed forward Artificial Neural Network model) for classification and w-Ranking Key Algorithm was used to rank the tweets.
- **FAST-NU**: This team participated from, FAST National University Karachi Campus, Pakistan. It submitted one *automatic* run described below:
  - *NU_Team_run01*: This run extracted textual features using tf*idf scores. All non -English tweets are translated using Google Translator API into English equivalent text. The logistic regression based classifier is used for classification.
- **HLJIT2017-IRMIDIS**: This team participated from Heilongjiang Institute of Technology, China. It submitted three *automatic* runs. The task was viewed as a classification task in all the runs and the feature selection was based on logistic regression method.
  - *HLJIT2017-IRMIDIS_task1_1*: SVM of Liner kernel classifier was used.
  - *HLJIT2017-IRMIDIS_task1_2*: AdaBoost classifier was used.
  - *HLJIT2017-IRMIDIS_task1_3*: SVM of Nonlinear kernel classifier was used.
- **HLJIT2017-IRMIDIS_1**: This team participated from Heilongjiang Institute of Technology, China. The task was viewed as a classification task in all the runs used words as a feature.
  - *HLJIT2017-IRMIDIS_1_task1_1*: LibSVM classifier was used.
  - *HLJIT2017-IRMIDIS_1_task1_2*: LibSVM classifier was used.
  - *HLJIT2017-IRMIDIS_1_task1_3*: Linear Regression model was used.
- **Iwist-Group**: This team participated from, Hildesheim University, Germany. It submitted one *automatic* run *Iwist_task1_1* that is described as follows. Pole-based overlapping clustering algorithm was used to measure the degree of relevance of the tweet. For ranking the tweets Euclidean distance was used as a similarity measure and the object closer to a pole was ranked higher.
- **Radboud_CLS Netherlands**: This team participated from Radboud University, the Netherlands and submitted the following two *semi-automatic* runs described as follows. Code-mixed tweets were preprocessed and translated to English using Google translator.
  - *Radboud_CLS_task1_1*: A lexicon and a set of handcrafted rules were used to tag the relevant n-grams. Then the class labels were automatically assigned to the tagged output. The output was initially ranked using combined score of human-estimated confidence of specific class label and tag pattern. However, the final ranking was generated by ordering the tweets within these ranked sets according to their tweet ID.
  - *Radboud_CLS_task1_2*: This run used a tool Relevancer for initial clustering of the tweets tagged as English or Hindi. English clusters were annotated and used as training data for the support vector machines (SVM) based classifier.
- **Amrita CEN 1**: This team participated from Amrita school of Engineering, Coimbatore, India. It submitted one *semi-automatic* run *AU_NLP_1* described as follows. The training data was tokenized. Classifier was trained using the word count as feature. For ranking the tweets cosine similarity was used.

3.2 Evaluation Measures and Result

We now report the performance of the methodologies submitted to the Task1 of FIRE 2017 IRMiDis Track. We consider the following measures to evaluate the performance – (i) **Precision at 100 (Precision@100)**: what fraction of the top ranked 100 results are actually relevant according to the gold standard, i.e., what fraction of the retrieved tweets are actually need-tweets or availability-tweets, (ii) **Recall at 1000 (Recall@1000)**: fraction of relevant tweets (according to the gold standard) that are in the top 1000 retrieved tweets, and (iii) **Mean Average Precision (MAP)** considering the full retrieved ranked list.

[3]https://nlp.stanford.edu/software/tagger.shtml

Run Id	Type	Precision @100	Recall @1000	MAP	Method summary
iitbhu_fmt17_task1_2	Automatic	0.7900	0.6160	0.4386	SVM classifier Undersampling was employed
iiests_IRMiDis_FIRE2017_1	Automatic	0.7850	0.3542	0.2639	TfidfVectorizer, LinearSVM, , Decision Tree Classifier
Bits_Pilani_1	Automatic	0.6800	0.2983	0.2073	POS tagging, word embeddings, Skip-gram model, fastText classifier
Bits_Pilani_2	Automatic	0.7300	0.2634	0.1993	POS tagging, word embeddings, CBOW model, fastText classifier
DataEngineeringGroup_1	Automatic	0.5400	0.2896	0.1304	POS tagging, Lemma identification, Logistic Regression model
HLJIT2017- IRMIDIS_1_task1_3	Automatic	0.6850	0.1662	0.1208	words as features Linear Regression model
iitbhu_fmt17_task1_1	Automatic	0.5600	0.1570	0.0906	Query generation by token disjunction more than threshold frequency, Apache Lucene
HLJIT2017-IRMIDIS_1_task1_2	Automatic	0.3650	0.1176	0.0710	Words as a feature LibSVM classifier
HLJIT2017-IRMIDIS_task1_3	Automatic	0.4450	0.1642	0.0687	Logistic regression based feature selection, SVM Nonlinear kernel classifier
DIA_Lab_NITK_task1_1	Automatic	0.3850	0.1437	0.0681	Doc2vec, Multilayer preceptron, w-Ranking Key
HLJIT2017-IRMIDIS_task1_2	Automatic	0.5500	0.1094	0.0633	Logistic regression based feature selection, AdaBoost classifier
HLJIT2017-IRMIDIS_1_task1_1	Automatic	0.3050	0.0636	0.0317	Words as a feature LibSVM classifier
Iwist_task1_1	Automatic	0.0350	0.0916	0.0291	POS tagging, Cosine similarity, Greedy approach search
HLJIT2017-IRMIDIS_task1_1	Automatic	0.1250	0.1414	0.0286	Logistic regression based feature selection, SVM Liner kernel classifier
NU_Team_run01	Automatic	0.0700	0.0478	0.0047	tf*idf scores, Logistic regression based classifier
Radboud_CLS_task1_1	Semi-automatic	0.7400	0.3731	0.2458	Linguistic approach, Tagged n-grams, Automatically assigned class labels
Radboud_CLS_task1_2	Semi-automatic	0.5500	0.2189	0.1736	Relevancer for initial clustering, SVM based classifier, Cosine similarity
AU_NLP_1	Semi-automatic	0.0800	0.0645	0.0199	Tokenization, Word count as feature, Classification, Cosine similarity

Table 3: Comparison among all the submitted runs in Task 1 (identifying need-tweets and availability-tweets). Runs are ranked in decreasing order of MAP score.

Table 3 reports the retrieval performance for all the submitted runs in Task1. Each of the measures (i.e. Precision@100, Recall@1000, Map) are reported by taking an average over both the topics need-tweets and availability-tweets.

It is seen that classification-based approaches performed better than the other methodologies based on word-embeddings or searching tools like Apache Lucene, as is evident from the scores in Table 3.

4 TASK 2: MATCHING NEED-TWEETS AND AVAILABILITY-TWEETS

In Task2, 5 teams participated and 10 runs were submitted. We first describe the runs, and then report the comparative evaluation.

4.1 Methodologies

We now describe the submitted runs.

- **DataBros** : This team participated from Indian Institute of Information Technology, Kalyani, India. It submitted one *automatic* run. This run used POS (Parts of Speech) tagging and matching-score was obtained from the number

Team Id	Precision@5	Recall	F-Score	Type	Method summary
DataBros	0.2482	0.3888	0.3030	Automatic	POS tagging, Common noun overlapping
Data Engineering Group	0.2081	0.2904	0.2424	Automatic	POS tagging, Cosine similarity, Brute force search
Data Engineering Group	0.1758	0.3677	0.2379	Automatic	POS tagging, Cosine similarity, Greedy approach search
HLJIT2017-IRMIDIS	0.1819	0.1546	0.1671	Automatic	Indri, Dirichlet smoothing, KL distance sorting model
HLJIT2017-IRMIDIS	0.2033	0.1405	0.1662	Automatic	Indri, Dirichlet smoothing, KL distance sorting model
HLJIT2017-IRMIDIS	0.2051	0.0913	0.1264	Automatic	Indri, Dirichlet smoothing, KL distance sorting model
HLJIT2017-IRMIDIS_1	0.0882	0.2178	0.1256	Automatic	Indri, Dirichlet smoothing, Correlation calculation
HLJIT2017-IRMIDIS_1	0.0825	0.1475	0.1058	Automatic	Indri, Dirichlet smoothing, Correlation calculation
HLJIT2017-IRMIDIS_1	0.0889	0.0211	0.0341	Automatic	Indri, Dirichlet smoothing, Correlation calculation
Radboud_CLS Netherlands	0.3305	0.4450	0.3793	Semi-automatic	n-grams, Resource tagging

Table 4: Comparison among all the submitted runs in Task 2 (matching need-tweets and availability-tweets). Runs are ranked in decreasing order of F-score.

of overlapping of common nouns between Need-tweets and Availability-tweets.

- **Data Engineering Group**: This team participated from Indraprastha Institute of Information Technology, Delhi, India. It submitted two *automatic* runs described as follows:
  - Both the runs used POS tag of nouns and similarity between Need-tweets and Availability-tweets were measured by cosine similarity. However, for the first submitted run the similarity threshold was set as 0.7 as inferred on the basis of experimentation. Thus, brute force approach was followed in searching.
  - In the second submitted run, greedy approach was followed and the search stopped as soon as it finds the first five or lesser availability tweets with a cosine similarity score greater than our set threshold of 0.7.
- **HLJIT2017-IRMIDIS**: This team participated from Heilongjiang Institute of Technology, China. It submitted three *automatic* runs. The task was viewed as an IR task. All the runs used the open source retrieval tool Indri language model based on the Dirichlet smoothing for retrieval and KL distance as the sorting model.
- **HLJIT2017-IRMIDIS_1**: This team participated from Heilongjiang Institute of Technology, China. It submitted three *automatic* runs. The task was viewed as an IR task. Need-tweets used as a query set and Availability-tweets used as a collection of documents. All the runs used Indri open-source retrieval tool and the Dirichlet smoothing language model to solve the matching problem. However, the three runs submitted by this team differ in preprocessing step.
- **Radboud_CLS Netherlands**: This team participated from, Radboud University, Netherlands, and submitted the *semi-automatic* run *Radboud_CLS_task1_1*. This method used the tagged output obtained in the processing the tweets for Task 1 using a linguistic approach. For every Need-tweet all the word *n*-grams were tagged as identifying a resource; the approach attempt to find an exact match in the Availability-tweets and ranked the Availability-tweets accordingly.

4.2 Evaluation Measures and Result

The runs were evaluated against the gold standards generated by manual runs. Additionally, the annotators (same as used to develop the gold standard) checked many of the need-availability pairs matched by the methodologies (after pooling), and judged whether the match is correct.

We have used the following IR measures to evaluate the runs.
(i) **Precision@5:** Let n be the number of need-tweets correctly identified (i.e., present in the gold standard) by a particular matching methodology. For each need-tweet, we consider the top 5 matching availability-tweets as matched by the method. The precision of a particular matching methodology is the fraction of pairs that are matched correctly by the methodology (out of the $5 \times n$ pairs).
(ii) **Recall:** The recall of matching is the fraction of all the need-tweets (present in the gold standard) which a methodology is able to match correctly.
(iii) **F-Score:** F-score of a matching methodology is the harmonic mean of the precision and recall.

Table 4 shows the evaluation performance of each submitted run, along with a brief summary. For each type, the runs are arranged in the decreasing order of the *F-Score*. It is evident that the methods which considered noun overlapping or cosine similarity between need-tweets and availability-Tweets to obtain matching-score (post POS tagging) outperformed the other methodologies.

5 CONCLUSION AND FUTURE DIRECTIONS

The FIRE 2017 IRMiDis track successfully created a benchmark collection of code-mixed microblogs posted during disaster events. The track also compared the performance of various methodologies in retrieving and matching two pertinent and actionable types of information, namely need-tweets and availability-tweets. We hope that the test collection developed in this track will help the research community in the development of a better model for retrieval and matching in future.

In this year's track we considered a static collection of code-mixed microblogs. However, in reality, microblogs are obtained in a continuous stream. The challenge can be extended to retrieve relevant microblogs from the live streaming of microblogs dynamically. We plan to explore this direction in the coming years.

ACKNOWLEDGEMENTS

The track organizers thank all the participants for their interest in this track. We also thank the FIRE 2017 organizers for their support in organizing the track.

REFERENCES

[1] M. Basu, K. Ghosh, S. Das, R. Dey, S. Bandyopadhyay, and S. Ghosh. 2017. Identifying Post-Disaster Resource Needs and Availabilities from Microblogs. In *Proc. ASONAM*.

[2] M. Basu, A. Roy, K. Ghosh, S. Bandyopadhyay, and S. Ghosh. 2017. Microblog Retrieval in a Disaster Situation: A New Test Collection for Evaluation. In *Proc. Workshop on Exploitation of Social Media for Emergency Relief and Preparedness (SMERP) co-located with European Conference on Information Retrieval.* 22–31. http://ceur-ws.org/Vol-1832/SMERP_2017_peer_review_paper_3.pdf

[3] M. Imran, C. Castillo, F. Diaz, and S. Vieweg. 2015. Processing Social Media Messages in Mass Emergency: A Survey. *Comput. Surveys* 47, 4 (June 2015), 67:1–67:38.

[4] Twitter-search-api 2017. Twitter Search API. (2017). https://dev.twitter.com/rest/public/search