Ensemble Algorithms for Microblog Summarization

Soumi Dutta
IIEST Shibpur

Vibhash Chandra
IIEST Shibpur

Kanav Mehra
IIEST Shibpur

Asit Kumar Das
IIEST Shibpur

Tanmoy Chakraborty
IIIT Delhi

Saptarshi Ghosh
IIT Kharagpur

We investigate whether off-the-shelf summarization algorithms can be combined to produce better quality summaries. To this end, we propose ensemble schemes that can combine the outputs of multiple base summarization algorithms, to produce summaries better than what is generated by any of the base algorithms.

Summarization of Things in a cyber-physical society involve multi-dimensional Summarization of a wide variety of data, including textual data from various online and offline sources, sensor data, and so on.[1] Especially, crowdsourced textual data from social media sites like Twitter are nowadays important sources of real-time information on ongoing events, including socio-political events, natural and man-made disasters, and so on. On such sites, microblogs are usually posted so rapidly and in such large volumes, that it is not feasible for human users to go through all the posts. In such scenarios, summarization of microblogs (tweets) is an important task. A large number of extractive summarization algorithms have been proposed, both for general text summarization[2] and specifically for microblogs.[3] Few studies have also compared the performance of different summarization algorithm son microblogs.[4,5] In this work, rather than trying to come up with a new summarization algorithm, we investigate whether existing off-the-shelf summarization algorithms can be combined to produce better quality summaries, compared to what is obtained from any of the individual algorithms.

Motivation: We selected nine well-known extractive summarization algorithms and executed all of them on the same set of microblogs. All algorithms were made to generate summaries of 30 tweets. Table 1 shows the overlap between the summaries generated by the different base algorithms for the same dataset (tweets posted during Typhoon Hagupit). The entry (i, j), $1 \leq i, j \leq 9$ in Table 1 shows the number of common tweets included in the summaries generated by the two algorithms A_i and A_j. It is evident that there is very low overlap between summaries generated by various base algorithms. Similar trends are observed for all our datasets. Thus, different summarization algorithms usually select very different sets of tweets in the summaries for the same set of input tweets. The different summarization algorithms are likely to estimate the relative importance of tweets based on different factors, and hence various summaries are likely to bring out different aspects of the input dataset. This large variation observed in the set of tweets

selected by different summarization algorithms motivated us to devise ensemble techniques that can combine the views of multiple base algorithms to produce summaries better than what any of the base algorithms generate.

TABLE 1. Overlap of Tweets in the Summaries Generated by Different Base Algorithms for the Typhoon Hagupit Dataset

Algo	CR	CW	FS	LR	LS	LH	MD	SB	SM
CR	-	0	0	0	0	0	0	0	0
CW	0	-	0	0	4	3	1	2	1
FS	0	0	-	0	3	2	0	0	0
LR	0	0	0	-	0	0	0	1	2
LS	0	4	3	0	-	6	0	1	0
LH	0	3	2	0	6	-	2	1	0
MD	0	1	0	0	0	2	-	1	0
SB	0	2	0	1	1	1	1	-	2
SM	0	1	0	2	0	0	0	2	-

Present work: In this work, we propose two ensemble schemes – (i) EnGraphSumm, a graph-based unsupervised ensemble summarization algorithm, and (ii) Learn2Summ, a supervised ensemble summarization algorithm based on the Learning-to-Rank paradigm.[6] The intuition behind our proposed ensemble methods is as follows. Both the ensemble methods consider those tweets that have been selected by at least one base algorithm. The tweets selected by the base algorithms are grouped by the unsupervised method according to some measure of tweet similarity. The unsupervised method then selects one tweet from each group. In this way, the method attempts to select important tweets while reducing redundancy in the final summary. The supervised ensemble algorithm attempts to learn a ranking of tweets according to importance, based on the rankings computed by different base algorithms and the performance of the base algorithms over a training set. The idea is to combine different rankings of tweets and learn how to rank tweets better (for inclusion in the final summary). Motivated by the importance of microblog summarization during disaster events,[3] we perform experiments over microblogs related to four recent disaster events. We show that the proposed ensemble algorithms can combine the outputs of multiple base algorithms to produce summaries that are of better quality than what is obtained from any of the base algorithms. Specifically, our proposed ensemble algorithms achieve up to 8.4 percent higher Rouge-2 Recall scores on average compared to the best performing baseline. We demonstrate the summaries generated by the base algorithms and some of our ensemble algorithms over a small sample dataset in the supplementary information accompanying this paper (available at http://cse.iitkgp.ac.in/~saptarshi/docs/ DuttaEtAl-IS-ensemble-summ-SuppleInfo.pdf). To our knowledge, this is one of the first attempts at designing ensemble schemes for combining the outputs of multiple text-summarization algorithms. Importantly, we do not assume any particular property of the base algorithms; hence, any extractive algorithm can be used in the proposed ensemble framework. Though the present work focuses on summarization of textual data (microblogs), the proposed ensemble approaches are applicable for different types of summarization problems that are important in today's cyber physical society.[1] We envisage that this work will open up scope for several future works on ensemble summarization, similar to successful use of ensemble techniques on data mining tasks such as classification and clustering.[7]

BASE SUMMARIZATION ALGORITHMS

For the present work, we consider the following nine algorithms as base summarization algorithms – (1) Clusterrank (CR),[8] (2) COWTS (CW),[3] (3) Frequency Summarizer (FS),[9] (4) LexRank(LR),[10] (5) LSA (LS),[11] (6) LUHN (LH),[12] (7) Mead (MD),[13] (8) SumBasic(SB),[14] (9) SumDSDR(SM).[15] These algorithms generally estimate an importance score for each textual unit in the input (e.g., sentence in a document, or a tweet in a set of tweets), and include the textual units in the summary in decreasing order of this score, until a pre-defined length of the summary is reached.

Note that, apart from the ones described above, other extractive summarization algorithms can also be used in our ensemble frameworks. We selected the above algorithms because their implementations are readily available. The supplementary information contains brief descriptions of the algorithms, and the availability of their implementations.

UNSUPERVISED ENSEMBLE SUMMARIZATION

In this chapter, we describe unsupervised ensemble frameworks for summarization. We describe a baseline ensemble algorithm and our proposed graph-based algorithm EnGraphSumm. Let A_1, A_2, ……,A_N be the base summarization algorithms, where N is the number of algorithms considered. For a given set of microblogs, we first run each A_i and obtain summaries of a fixed length (K tweets). Then, we use the ensemble techniques on these summaries, to obtain an ensemble summary of the same length (K tweets). Each base algorithm A_i selects a set of tweets to include in the summary; let S_i denote the set of tweets included in the summary output by A_i. The ensemble techniques consider the set of tweets, $S = \bigcup_N^{i=1} S_i$,i.e., the set of tweets that have been selected by at least one base algorithm. Additionally, for a particular tweet t, let A(t) denote the set of base algorithms which have selected t to include in their summary.

Baseline: Voting Approach

In this simple strategy, each tweet t Є S is assigned a score score(t) = |A(t)|, i.e., the number of base algorithms that have selected this particular tweet to include in the summary. Basically, each base algorithm is considered to 'vote' for the tweets that it includes in the summary. The tweets in S are ranked in decreasing order of this score, and the top K tweets are chosen to be included in the ensemble summary. If necessary, ties among tweets having the same score are broken by random selection.

EnGraphSumm: Proposed Ensemble Algorithm

We now describe the algorithm EnGraphSumm which consists of three steps. The pseudo code of the algorithm can be found in the supplementary information.

Step 1 Constructing a Tweet-Similarity Graph

EnGraphSumm first constructs an undirected graph G where the tweets in S are the nodes, and two nodes are connected by an edge if the two corresponding tweets are 'similar.' Several methods can be used to measure the similarity sim(t_1, t_2) of two tweets t_1 and t_2. Specifically, we experiment with the following two methods:

1. Text similarity (TextSim): We consider two tweets to be similar if they contain similar words or terms. We represent a tweet as a bag (set) of words, after removing a standard set of English stop words, punctuation symbols, URLs, @usermentions, etc. To measure the similarity sim(t_1, t_2), we compute the Jaccard similarity between the bags (sets) of wordsin the two tweets. This similarity measure lies in the range[0, 1] where the maximum value 1 indicates two very similar tweets.

2. Vector similarity (VecSim): Here we consider two tweets to be similar if they have been selected by the same base algorithms to be included in the summaries. We represent a tweet as a binary vector of size N (the number of base summarization algorithms), where the i^{th} term of the vector is 1 if the tweet was included in the summary output by algorithm A_i, 0 otherwise. The similarity $sim(t_1, t_2)$ between two tweets is measured by the inner-product (term-wise product) of the two corresponding vectors. Effectively, $sim(t_1, t_2) = |A(t_1) \cup A(t_2)|$ lies in the range [0,N], and a value of P Є [0,N] indicates that the two tweets were selected by P base algorithms in common.

3. Embedding similarity (EmbedSim): Here we construct embeddings of tweets using the popular text embedding tool Word2Vec,[16] and measure the similarity between these embeddings. Specifically, we train Word2vec on the given set of tweets (after stop word removal), using the continuous bag of words (CBOW) model along with hierarchical softmax, and the following parameter values – vector size: 100, context size: 5, learning rate: 0.05, iteration: 500. Word2vec gives a vector for each term (term-vector) which captures the semantic context of the term. The vector for a tweet (tweet-vector) is obtained by computing the average (mean) of the term-vectors in the tweet. The similarity between two tweets is measured as the cosine similarity of the tweet-vectors of the two tweets. This method can be considered as an improved version of TextSim, which can identify similarity between two tweets that are semantically similar but use different terms.

While constructing the graph G, we consider a similarity threshold sim_{th}, and an edge is added between two nodes (tweets) t_1 and t_2 if $sim(t_1, t_2) \geq sim_{th}$. The choice of sim_{th} is described later.

Step 2 Identifying Groups of Similar Tweets (Nodes in the Graph)

Once the graph G is constructed, EnGraphSumm identifies groups of similar nodes (tweets). Several approaches can be taken to find groups of similar nodes, out of which we adopt the following:

1. Identifying connected components (ConComp): Since an edge between two nodes in G indicates that the two tweets are similar, identifying connected components in G is a straightforward way of detecting groups of similar tweets.

2. Identifying communities (Community): Various community detection algorithms for graphs can be used to identify subgroups of nodes such that the nodes within the group are more densely connected to each other than to nodes outside this group. We use the popular Louvain algorithm,[17] to find communities in G.

Note that text-clustering algorithms can also be applied to identify groups of tweets (nodes) whose textual content is similar. We leave this direction for future work.

Step 3 Selecting a Representative Tweet from a Group

Once a group of similar tweets (nodes) is identified, EnGraphSumm selects one tweet (node) from each group as a representative of the group and adds it to the summary. This selection can be made in several ways, out of which we try the following:

1. Tweet with the maximum length (MaxLen): Here we select the tweet having the maximum string length, with the intuition that including a longer tweet would make the summary more informative.

2. Node with the maximum degree (MaxDeg): Out of an identified group of nodes, we select the node having the highest degree. This approach follows the intuition that a high degree node in the similarity graph G is similar to more nodes in the group, and hence is a better representative of the group.

3. Tweet with maximum TF-IDF score (maxSumTFIDF): In this approach, we attempt to select more 'informative' tweets. To this end, we use the popular Information Retrieval measures 'term frequency' (TF) and 'inverse document frequency' (IDF). A particular tweet t is considered as a bag (set) of words, as described in the TextSim approach. For each word w Є t, we

compute (i) term frequency of w in t, and (ii) the IDF of w in the whole set of tweets. Both TF and IDF are log-normalized. Finally, the TF-IDF score of the tweet t is taken as the sum of TF-IDF scores of all words in t. Out of an identified group of similar tweets, we select the tweet with the highest TF-IDF score.

4. Tweet with maximum BM25 score (MaxSumBM25): Several prior works in Information Retrieval have observed that the BM25 ranking model,[18] performs better than the TF-IDF model in the case of short text-like tweets. Hence in this version, the BM25 score18 is computed for each distinct word, and then the score of each word in a tweet is summed up to get the BM25 score for the tweet. Out of an identified group of nodes, we select the node (tweet) having the highest BM25 score.

Selection of a Similarity Threshold

Our experiments with various microblog datasets demonstrated that it is difficult to choose a suitable value for the threshold similarity sim_{th} that would work well for different datasets. Hence we took the following approach. For a given sim(), we consider its range [sim_{min}, sim_{max}]. We initialize the threshold $sim_{th} = sim_{max}$, and construct the graph G accordingly. Then we identify groups of similar tweets, and select a representative tweet from each group of size larger than one, considering groups in decreasing order of their size. Larger groups are given preference, since a larger number of similar tweets implies that the common topic of those tweets is more important. Note that once a representative tweet from a group is selected, the other tweets in the group are removed, so as to prevent redundancy in the summary. In the next iteration, we reduce sim_{th} by a step sim_{dec}, and construct G again. This process is repeated until sim_{th} becomes equal to the lowest possible similarity value sim_{min}. At this stage, if K tweets have not yet been included in the summary, then the remaining tweets are ranked in decreasing order of their string lenth, and the requisite number of top (longest) tweets are selected.

For the TextSim function (Jaccard similarity), $sim_{min} = 0$,$sim_{max} = 1$, and we consider $sim_{dec} = 0.1$. For the VecSimsimilarity function, $sim_{min} = 0$, $sim_{max} = N$ (the number of base summarization algorithms), and we consider $sim_{dec} = 1$.The detailed pseudocode of the algorithm EnGraphSummcanbe found in the accompanying supplementary information.

SUPERVISED ENSEMBLE SUMMARIZATION

This Chapter discusses supervised ensemble algorithms. Here we assume that the datasets are divided into two parts –

1. Training set: For these datasets, we assume we already know (i) the summaries generated by all base algorithms, and (ii) some performance measure (e.g., Rouge scores) of the summaries generated by the base algorithms.

2. Test set: We use the training set to develop ensemble summarization algorithms for the test datasets.

As in the earlier discussion on unsupervised ensemble algorithms, let the base summarization algorithms be $A_1, A_2, \ldots, A_N$, and A(t) denote the set of base algorithms which have selected the tweet t to include in their summary. Let $P(A_i)$ be some measure of the performance of A_i, $1 \leq i \leq N$ over the training set.

Baseline: Weighted Voting Approach

Similar to the unsupervised voting approach, each base algorithm that selected a particular tweet t (while summarizing the test set) is considered to vote for t. Here, the vote of the base algorithm A_i is weighted by the performance of A_i over the training set. Thus, each tweet t Є S (the union of sets of tweets selected by all the base algorithms) is assigned a 'goodness score' $score(t) = \sum_{A_i \in A(t)} P(A_i)$ where $P(A_i)$is the performance measure (e.g., Rouge score) of A_i. The

tweets in S are ranked in decreasing order of this score, and the top K tweets are chosen to be included in the ensemble summary (ties, if any, broken by random selection).

Learn2Summ: Proposed Ensemble Algorithm

We now describe Learn2Summ that is based on the popular Learning-to-Rank (L2R) paradigm.[6] For a particular train dataset, we compute score(t) for each tweet t (as described above), and rank the tweets based on this score. Next we consider a feature-vector for each tweet; the features in the vector are detailed below. We learn a ranking model based on the feature-vectors and the ranked list of tweets according to score(t), using standard Learning-to-Rank algorithms.[6] The learned ranking model is used to rank tweets for the test dataset, and the top-ranked K tweets are selected for inclusion in the ensemble summary.

Feature vectors: The features are meant to capture the type of tweets that get high scores, i.e., are selected by the base algorithms that perform well over the training datasets. We experiment with two types of vectors:

1. Base algorithms as features: We represent a tweet as a binary vector of size N (the number of base summarization algorithms), where the i^{th} term of the vector is 1 if the tweet was included in the summary output by algorithm A_i, 0 otherwise. These vectors are similar to what were used for the VecSim unsupervised approach.

2. Text-based features: The text-based features essentially capture the informativeness of the tweets, so that the correlation of informativeness of the tweets with their ranking (if any) can be learned. For a tweet, we compute the following features. Note that, apart from the first feature, all features are computed after pre-processing the text by case-folding and removal of English stop words: (1) total number of words; (2) number of words excluding English stop words; (3) sum of TF (term frequency) of all words; (4) sum of IDF (inverse document frequency) of all words, where IDF of a word is computed based on all the tweets in a particular dataset; (5) sum of TFIDF of all words; (6) number of hashtags; (7) whether the tweet is a retweet (binary feature); (8) number of user-mentions; (9) number of numerals; (10) number of nouns; (11) number of verbs; and (12) entropy, computed over all words w in the tweet t as, $H(t) = -\sum_{w \in t} p_w \cdot \log p_w$, where p_w is the probability of occurrence of the word w in the particular dataset.

For the L2R algorithms, we used the RankLib library (https://sourceforge.net/p/lemur/wiki/RankLib/) that contains implementations of several popular L2R algorithms. Specifically, we experimented with the L2R algorithms RankBoost, Random- Forest, and MART.

Note that, if multiple training datasets are available, we obtain the ranked list and feature-vectors of tweets for each of the training datasets, and learn a common ranking model from all the training datasets. The results are reported in the next Chapter.

EXPERIMENTS AND RESULTS

We now describe our experiments and results. We start by describing the datasets and the evaluation measures used, and then we compare the performance of various summarization algorithms.

Experimental Setup

Datasets: We re-use the datasets from our prior work,[3] consisting of tweets posted during four recent disaster events – (i) Bomb blasts in Hyderabad, India, (ii) Typhoon Hagupit in Philippines, (iii) Floods in Uttaranchal state of India, and (iv) Sandy Hook elementary school shooting in USA. The English tweets posted during each event were collected through the Twitter API using keyword search.

We initially considered the chronologically earliest 5,000tweets for each event. It is known that tweet streams often contain duplicates due to re-tweeting/re-posting of tweets. We removed such duplicates which are not useful for the purpose of summarization. After de-duplication the numbers of distinct tweets in the four datasets are respectively 1,413; 1,461; 2,069; and 2,080.

Evaluation of summarization algorithms: We follow the standard procedure of generating gold standard summaries by human annotators and then comparing the algorithm-generated summary with the gold standard ones. Three human annotators were asked to independently summarize each of the datasets and prepare summaries of K = 30 tweets each. Each annotator is well-versed in English and use of social media, and none is an author of this paper.

We used the standard ROUGE Recall scores of the(i) ROUGE-2 and (ii) ROUGE-L variants for evaluating the quality of the summaries.[19] These scores respectively measure what fraction of the (i) bigrams and (ii) longest matching sequence of words in the gold standard summaries are covered by the summaries produced by the algorithms.

It can be noted that summarization is inherently a subjective process, and the same dataset can be summarized differently by different human annotators. We actually observed signification variations in the summaries written by the three annotators for the same dataset – we quantify these variations in the supplementary information. For computing the ROUGE scores reported in this paper, all the three summaries written by the annotators (for a given dataset) were collectively used as the gold standard.

Table 2. Performance of the Base Summarization Algorithms Averaged Over All Datasets

Base algorithm	Rouge-2 Recall	Rouge-L Recall
ClusterRank (CR)	0.08598	0.26838
COWTS (CW)	**0.17896**	**0.44539**
FreqSum (FS)	0.14732	0.36018
Lex-Rank (LR)	0.0489	0.15254
LSA (LS)	0.15994	0.42336
LUHN (LH)	0.16504	0.40145
Mead (MD)	0.11719	0.37086
SumBasic (SB)	0.1012	0.32899
SumDSDR (SM)	0.09848	0.26016

The best performance is by the COWTS algorithm, and is highlighted in boldface.

Table 3. Performance of the Unsupervised Ensemble Algorithms in Terms of Rouge-2 and Rouge-L Recall Scores Averaged Over All Datasets

Ensemble algorithm	Rouge-2	Rouge-L
Voting (Baseline)	0.13976	0.36768
TextSim–ConComp–MaxDeg	0.15788	0.37436
TextSim–ConComp–MaxLen	0.16505	0.38597
TextSim–ConComp–maxSumTFIDF	0.15961	0.3804
TextSim–ConComp–MaxSumBM25	0.15867	0.37854
TextSim–Community–MaxDeg	0.14932	0.36934

TextSim–Community–MaxLen	0.14625	0.37206
TextSim–Community–maxSumTFIDF	0.14836	0.37281
TextSim–Community–MaxSumBM25	0.14193	0.36266
VecSim–ConComp–MaxDeg	**0.19196**	**0.4457**
VecSim–ConComp–MaxLen	**0.19397**	**0.45057**
VecSim–ConComp–maxSumTFIDF	**0.18863**	**0.45995**
VecSim–ConComp–MaxSumBM25	0.15601	0.37934
VecSim–Community–MaxDeg	0.14377	0.39565
VecSim–Community–MaxLen	0.14515	0.39102
VecSim–Community–maxSumTFIDF	**0.18984**	**0.45906**
VecSim–Community–MaxSumBM25	0.17147	0.39576
EmbedSim–ConComp–MaxDeg	0.14615	0.36093
EmbedSim–ConComp–MaxLen	0.17898	0.40099
EmbedSim–ConComp–maxSumTFIDF	0.17283	0.39372
EmbedSim–ConComp–MaxSumBM25	0.16094	0.37756
EmbedSim–Community–MaxDeg	0.16573	0.37886
EmbedSim–Community–MaxLen	0.178	0.39589
EmbedSim–Community–maxSumTFIDF	0.17356	0.39384
EmbedSim–Community–MaxSumBM25	0.17356	0.39384

The values better than COWTS (the best base algorithm) are in boldface.

Performance of Base Algorithms

We first run all the base summarization algorithms on each dataset. Table 2 reports the performance of the base summarization algorithms, averaged over all the datasets. The COWTS algorithm performs the best for all the measures. This result is not surprising, since COWTS is especially developed for summarization of tweets posted during disaster events.

Performance of Unsupervised Ensemble Algorithms

We apply the two unsupervised ensemble algorithms on the summaries produced by the base algorithms. Table 3 reports the performance of the ensemble algorithms, averaged over all the datasets. The voting method (baseline) performs worse than several of the base algorithms, which shows that ensemble summarization is not a trivial problem.

The proposed EnGraphSumm scheme performs better than all the base algorithms in several cases; the performances that are better than that of COWTS (the best base algorithm) are highlighted in boldface. In general, VecSim gives the best performance, followed by EmbedSim and then TextSim. Specifically, the best Rouge-2 Recall score is obtained with the MaxLen function (8.4 percent higher than that of COWTS, the best performing base algorithm), while the best Rouge-L Recall score is obtained with the maxSumTFIDF function (3.3 percent higher than that of COWTS).

Table 4. Performance of the Supervised Ensemble Algorithms in Terms of Rouge-2 and Rouge-L Recall Scores Averaged Over All Datasets

Ensemble algorithm	Rouge-2	Rouge-L
Weighted Voting (Baseline)	0.13884	0.38941
RankBoost with base algos as features	0.16716	0.43417
RandomForest with base algos as features	0.1617	0.42942
RankBoost with text-based features	**0.18326**	0.43846
RandomForest with text-based features	**0.1894**	0.44172
The values better than COWTS (the best base algorithm) are in boldface.		

Performance of Supervised Ensemble Algorithms

Now we describe the performance of the supervised ensemble algorithms. Since we have four datasets, we follow a training-testing approach analogous to *n-fold cross validation* with $n = 4$, as follows. In each iteration, we use three of the datasets for training, and the other as test dataset. We perform four iterations, where each dataset is considered as the test dataset in one iteration, and we report results averaged over the four iterations.

Table 4 shows the Rouge scores for the summaries generated by different ensemble algorithms. Similar to the unsupervised case, we find that the weighted voting scheme performs worse than some of the base algorithms. Among the proposed ensemble algorithms, the RankBoost L2R algorithm performs slightly better when the base algorithms are used as features, while the RandomForest L2R algorithm performs better with the text-based features. The best performance is achieved using RandomForest L2R algorithm with text-based features (Rouge-2 Recall score: 0.189), which is a 5.8 percent improvement over the best base algorithm (COWTS). The Rouge-L score of this ensemble algorithm (0.442) is also very close to that of COWTS (0.445).

Overall, we see that among the proposed ensemble algorithms, the graph-based unsupervised algorithm performs better than the supervised algorithm, both in terms of Rouge-2and Rouge-L Recall scores. The supplementary information accompanying the paper demonstrates the summaries generated by the base algorithms and some of the unsupervised algorithms developed in this work.

CONCLUSION

Though there have been prior works on ensemble methods for classification and clustering,[7] to our knowledge, this is the first attempt to develop ensemble methods for text summarization. To summarize our contributions: (1) We show that different extractive summarization algorithms produce very different summaries for the same input data. (2) We propose two ensemble algorithms – first, an unsupervised graph-based scheme, and second, a supervised scheme based on Learning-to-Rank – for combining the outputs of multiple extractive summarization algorithms. (3) We demonstrate that it is possible to combine off-the-shelf summarization algorithms to achieve better summarization for microblogs. As one class of summarization methods represent a dimension of processing texts, this research result also verifies the advantages of the multi-dimensional summarization method proposed in Multi-dimensional summarization in cyber-physical society.[1]

In future, we plan to extend the ensemble schemes to abstractive summarization algorithms by considering different text fragments (instead of whole tweets) selected by abstractive algorithms for generating the ensemble summaries.

ACKNOWLEDGEMENTS

The authors thank the anonymous reviewers and the Editor for their constructive comments, and Dr. Kripabandhu Ghosh, IIT Kanpur, India for useful discussions. The authors also acknowledge the annotators who helped in evaluating the algorithms. Additionally, T. Chakraborty would like to acknowledge the support of the Infosys Center for AI, IIIT Delhi, India, and the Ramanujan Fellowship, SERB, DST, India.

REFERENCES

1. H Zhuge, *Multi-dimensional summarization in cyber-physical society*, Morgan Kaufmann, 2016.
2. V. Gupta and G.S. Lehal, "A Survey of Text Summarization Extractive Techniques," *IEEE Journal of Emerging Technologies in Web Intelligence*, vol. 2, no. 3, 2010, pp. 258–268.
3. K. Rudra et al., "Extracting situational information from microblogs during disaster events: A classification-summarization approach," *Proc. ACM CIKM*, 2015.
4. S. Mackie et al., "Comparing Algorithms for Microblog Summarisation," *Proc. CLEF*, 2014.
5. D.I. Inouye and J.K. Kalita, "Comparing twitter summarization algorithms for multiple post summaries," *Proc. IEEE SocialCom / PASSAT*, 2011.
6. T.-Y. Liu, "Learning to rank for information retrieval," *Foundations and Trends in Information Retrieval*, vol. 3, no. 3, 2009, pp. 225–331.
7. G. Seni and J. Elder, *Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions*, Morgan and Claypool Publishers, 2010.
8. N. Garg et al., "Clusterrank: a graph based method for meeting summarization," *INTERSPEECH. ISCA*, 2009, pp. 1499–1502.
9. *Text summarization with NLTK*, 2014; https://tinyurl.com/frequencysummarizer.
10. G. Erkan and D.R. Radev, "Lexrank: Graph-based lexical centrality as salience in text summarization," *Journal of Artificial Intelligence Research*, vol. 22, no. 1, 2004, pp. 457–479.
11. Y. Gong and X. Liu, "Generic text summarization using relevance measure and latent semantic analysis," *Proc. SIGIR*, 2001, pp. 19–25.
12. H.P. Luhn, "The automatic creation of literature abstracts," *IBM Journal of Research & Development*, vol. 2, no. 2, 1958, pp. 159–165.
13. D.R. Radev, E. Hovy, and K. McKeown, "Introduction to the special issue on summarization," *Comput. Linguist.*, vol. 28, no. 4, 2002, pp. 399–408.
14. A. Nenkova and L. Vanderwende, "The impact of frequency on summarization," *Microsoft Research, Tech. Rep.*, 2005.
15. Z. He et al., "Document summarization based on data reconstruction," *Proc. AAAI Conference on Artificial Intelligence*, 2012, pp. 620–626.
16. T. Mikolov, S.W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013.
17. V.D. Blondel et al., "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics*, vol. 2008, 2008, p. P10008.
18. S.E. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Foundations and Trends in Information Retrieval*, vol. 3, no. 4, 2009, pp. 333–389.
19. C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," *Proc. ACL Workshop on Text Summarization Branches Out*, 2004.

ABOUT THE AUTHORS

Soumi Dutta is a PhD candidate at the department of computer science and technology (CST), Indian Institute of Engineering Science and Technology (IIEST), Shibpur. She received her Bachelor of Technology in information technology and her Master of Technology in computer science, both from Techno India Group. Her research interests include textand data mining, and social media analysis.

Vibhash Chandra is an under graduate student at the department of CST, IIEST, Shibpur, India. His research interests are in information retrieval and natural language processing.

Kanav Mehra is an undergraduate student at the department of information technology, IIEST, Shibpur, India. His research interests are in data mining and natural language processing.

Asit Kumar Das is an associate professor at the department of CST, IIEST, Shibpur, India. His research areas are data mining and machine learning, text and image mining, crime data analysis, and audio and video data analysis.

Tanmoy Chakraborty is an assistant professor and a Ramanujan Fellow at the department of CSE, Indraprastha Institute of Information Technology, Delhi, India. He was earlier a postdoctoral researcher at University of Maryland, College Park, USA. He finished his Ph.D. as a Google India PhD fellow from IIT Kharagpur, India. His research interests include data mining, social media and data-driven cyber security.

Saptarshi Ghosh is an assistant professor at the department of CSE, Indian Institute of Technology, Kharagpur, India. He completed his PhD from the same institute and was a Humboldt post-doctoral fellow at MPI-SWS, Germany. His research interests include social network analysis, data mining, and information retrieval.