Twitter Event Summarization by Exploiting Semantic Terms and Graph Network

Quanzhi Li, Qiong Zhang
Alibaba Group, US
Bellevue, WA 98004, USA
{quanzhi.li@alibaba-inc.com, qz.zhang@alibaba-inc.com}

Abstract

Twitter is a fast communication channel for gathering and spreading breaking news, and it generates a large volume of tweets for most events. Automatically creating a summary for an event is necessary and important. In this study, we explored two extractive approaches for summarizing events on Twitter. The first one exploits the semantic types of event related terms, and ranks the tweets based on the score computed from these semantic terms. The second one utilizes a graph convolutional network built from a tweet relation graph to generate tweet hidden features for tweet salience estimation. And the most salient tweets are selected as the summary of the event. Our experiments show that these two approaches outperform the compared methods.

Introduction

Social media services, such as Twitter, generate a large volume of content (tweets) for most events. This study aims at exploring new summarization methods for real-time events detected from Twitter. Text summarization methods can be classified into two categories. Extractive methods select a subset of words, phrases, or sentences from the original document to generate a summary. In contrast, abstractive approaches generate a summary by using words and phrases that may not appear in the source text. Our approaches belong to the extractive method category. We choose one or more tweets from a event cluster to represent the event. Usually, only one or two tweets are selected. One reason of selecting only a couple of tweets is that many applications prefer a very short summary, due to their limited UI space or other reasons. The proposed approaches have been adapted in three production systems, which need to monitor real-time events on different topics. They prefer selecting only one to three tweets per event as summary, in order to have more events displayed on one screen. In this study, our experiment also shows that human annotators usually also choose only one to three tweets as event summary. Other applications may want to select more tweets as summary.

To find the best tweets to represent an event, we first need to understand the event characteristics. An event is usually defined by the *4Ws* questions: *who, what, where* and *when* (Mohd 2007). An event tweet usually contains terms corresponding to these aspects, and these terms can be classified into different semantic classes/types, such as entity names (who), location (where), hashtag (topic & what), temporal expression (when), verb & noun (what), and mention (who). Different semantic types have different degrees of importance for describing an event, and this will have implications on both event detection and event summarization algorithms. Previous studies on event summarization from social media have not explicitly exploited this type of information. They do not classify the terms into different semantic classes; they just treat them equally important in their algorithms, e.g. in Hybrid TF-IDF (Inouye and Kalita 2016) and sumBasic (Vanderwende et al. 2007). In the first approach, we take the semantic type of a term into consideration when determining its weight. We hypothesize that classifying tweet terms into their corresponding semantic classes, assigning different weights to them, and then integrating them together will improve the event summarization performance. Therefore, in the first approach, we split the term space into groups of terms, or semantic classes.

In the second approach, we consider the tweet event summarization task as a special multi-document summarization problem, and utilize graph-based neural networks. The tweets selected as the summary should be both *representative* and *informative.* To be representative in an event cluster, a tweet needs to convey similar information with other tweets in this cluster. We use a tweet relation graph to measure how representative a tweet is. In this graph, each node is a tweet, and the edge weight measures how similar two nodes are. To reflect the *informative* aspect, we calculate a score for each node in the relation graph, using the semantic terms the tweet contains. A tweet having more semantic terms usually will be more informative. To better utilize the tweet relation information and the representational power of neural network, we apply graph convolutional network (GCN) (Kipf and Welling 2017; Yasunaga et al. 2017) on the tweet relation graph. A salience score is estimated for each tweet based on the hidden features generated by GCN, and the most salient ones are chosen as the summary. We evaluated the two approaches using 1,000 events detected from Twitter.

Related Studies

There are many studies on text summarization, but only a few of them focus on social media. One of the most popular extractive summarization methods is the centroid-based approach (Becker et al. 2011; Radev et al. 2004). Becker et al. (2011) studied three centroid-based methods, and their experiment showed that the Centroid method outperformed the other two, Degree and LexRank. LexRank (Erkan and Radev 2004) is a graph-based approach, which creates an adjacency matrix for text units and computes the stationary distribution considering it as a Markov chain. TextRank (Mihalcea and Tarau 2004) is another graph-based one, using PageRank to find the highly ranked sentences.

sumBasic is a simple but effective method, which mainly depends on word frequency (Vanderwende et al. 2007). It chooses the tweet that has the highest sum of word probability, which is computed from word frequency. To handle the special situation of tweet event, Inouye and Kalita (2016) redefine TF-IDF in terms of a hybrid document. In their Hybrid TF-IDF approach, the TF component of the TF-IDF formula uses the entire collection of tweets, while the IDF component treats each tweet as a separate document. Experiments from (Inouye and Kalita 2016; Alsaedi et al. 2016) show that Hybrid TF-IDF and sumBasic outperform other approaches. Alsaedi et al. (2016) augmented the TF-IDF and Centroid methods by considering the time window of an event. This method is mainly used for retrospective events. Mishra and Berberich (2017) propose a novel divergence-based framework that selects excerpts from an initial set of pseudo-relevant documents. Their algorithm requires an explicit user input, and it mainly works for long text. This is different from our case, which is to summarize tweets without using any user query.

The task B of TREC 2018 real-time summarization track (Sequiera et al, 2018) asks the participating systems to output a list of tweets ordered by relevance to users' interests every day during the evaluation period. This task is different from our case, which asks for the general summary of an event, not specific to a user's interest. Four teams participated in this task, and the team having the best performance uses a method adapted from (Chellal et al. 2017), which ranks the tweets based on their cosine similarity with user's interest. Except the similarity calculation for user interest relevance, this method is basically the same as the "the earliest tweet" approach tested in our experiments.

Recent text summarization studies on single document have employed neural networks (Cao et al. 2016; Wang and Ling 2016; Cheng and Lapata 2016; Nallapati et al. 2017; See et al. 2017; Cao et al. 2017). Chang et al. (2013; 2016) regard Twitter summarization as a supervised classification task through mining rich social features, such as user influence. Wang and Ling (2016) employ encoder-decoder RNNs to generate short abstractive summaries for opinions. Cheng and Lapata (2016) train an extractive summarization approach using attention-based encoder-decoder RNNs to sequentially label summary-worth sentences. See et al. (2017) augment the standard attention-based encoder-decoder RNNs using pointer generator network. SUMMARUNNER (Nallapati et al. 2017) is an abstractive summarization model using an RNN-based encoder. Narayan et al. (2018) proposed a reinforcement learning-based system trained by globally optimizing the ROUGE score. The model from Zhou et al. (2018) extracts sentences from a document by jointly learning to score and select sentences. LATENT (Zhang et al., 2018) uses a latent model to directly maximize the likelihood of human summaries given selected sentences. Liu et al. (2019) use structured attention to induce a multi-root dependency tree representation of the document while predicting the output summary. Most of these models focus on either abstractive method or single document. Applying these sequence-to-sequence approaches to the multi-document summarization task has not been successful. He and Duan (2018) utilize the Twitter network structure to explore whether social relations can help Twitter summarization. Wang and Zhang (2017) build a joint model to filter, cluster, and summarize the tweets for new events. Their summarization model uses a multi-layer perceptron network to estimate a probability score for each tweet and then rank them. The two approaches we presented in this paper are different from previous studies.

Semantic Class Based Approach

In the semantic class-based approach, we have two methods to rank the tweets. We first describe each semantic class, and then present the two methods, Semantic-A and Semantic-B, for selecting the best tweets as the summary of an event. Given a tweet, the first method (i.e., Semantic-A) integrates different semantic terms together to compute a tweet score, and the second method (i.e., Semantic-B) uses learning to rank algorithm to decide if a tweet is the top candidate. We call a term belonging to one of the semantic classes described below as "semantic term". Previous study (Li et al, 2017) has used semantic terms to detect event clusters from Tweet data streaming.

Semantic Classes

Named Entity. In an event, "*who*" is usually defined by proper nouns (named entities), such as people or organization names. Occasionally, "*what*" is also represented by named entities. Location is also represented by named entities, but in this study, we have a separate class for location.

The entity extraction model uses three vectors to form the word representation: the word embedding, a character representation vector generated from Bi-LSTM model, and another character representation generated from CNN model. And a word level Bi-LSTM model is used to detect the entity label, followed by a CRF layer to exploit the correlation between labels in local context and the sentence, by jointly learning the best label chain for a given sentence. Experiment on the English data from CoNLL 2003 (Sang and Meulder 2003) shows that the performance of this approach is comparable to that of the state-of-the-art algorithms (Ma and Hovy 2016; Yadav and Bethard 2018; Yadav et al. 2018). If an entity is identified as location, then it will be classified as location. Therefore, in this paper, when we talk about named entities, it only refers to people or organization.

Mention. Mention (e.g. @Hillary) has been ignored by previous studies, but we found that in many events "*who*" is represented by a mention. Mentions have the same semantic meaning as named entities, but we put them in a separate class because of its special usage pattern.

Location. Locations are identified by our entity extraction algorithm described above.

Verb. Verbs usually describe the "*what*" aspect of an event, such as *killed* and *sued*. A stop word list is used to filter out the very common words, such as *take, do*, etc.

Noun. Same as verbs, nouns, such as *hurricane* and *bombing*, are also used to describe the "*what*" aspect of an event.

Hashtag. Hashtags usually describe the topic of an event, but they may also reflect the "*what*" aspect, e.g. *#bombing*.

Temporal Information. The "*when*" aspect of an event is usually described by temporal expressions, such as *8am today* or *this morning*. We use an in-house algorithm to identify temporal expressions.

Semantic-A: Semantic Class Integration

Terms in different semantic classes play different roles in an event, so we give them different weights in the summarization algorithm. The terms in the same semantic class will have the same weight. In addition to the class weight, we also consider term frequency in the event tweet collection, which is reflected on the term probability distribution described below. Retweet count and tweet length are also considered in our algorithm.

Term Probability Distribution. A probability distribution is calculated for each semantic term in the tweet collection of an event. For term *t*, its probability is:

$$p(t) = \frac{c}{N} \tag{1}$$

where N is the total number of tweets in this event, and c is the total number of tweets that contain this term.

Tweet Score Integration. Given a tweet T, we integrate the scores of the semantic terms in this tweet to get its score:

$$Y = \sum_{t \in T} \sum_{u \in S} w_u \cdot p(t) \tag{2}$$

where u is a semantic type belonging to semantic type collection S, t is a semantic term in tweet T, w_u is the weight for semantic type u, and $p(t)$ is the probability of t. The parameter w_u for each semantic class is learned from training data.

Tweet Score Normalization. In Equation 2, we do not consider the length of a tweet. This means it will always bias toward longer tweets, which is not desired, although usually a longer text segment conveys more information. We define a normalization factor, which is the logarithm of the tweet length (number of words, excluding stop words) plus 1, to normalize the tweet score by dividing it by this factor.

Another factor we considered for adjusting the tweet score is the retweet information. Retweeting reinforces a message, and the number of retweets is an indication of popularity. The following equation shows how the tweet score is normalized:

$$Z = \frac{Y * log\ (1{+}r)}{log\ (1{+}n)} \tag{3}$$

where Y is the tweet's score from Equation 2, r is the retweet value of this tweet, and n is the number of words in this tweet. Here we use $log(1{+}r)$ and $log(1{+}n)$, to avoid a value of 0 when r or n is 1. The tweet with the highest score is selected as the summary of the event. We also tested other equations to deal with the retweet and tweet length problems, and Equation 3 gave us the best performance.

The features described above convey different types of information: syntactic, semantic, co-occurrence, popularity, time, and hashtag composition. We hypothesize that integrating them together will improve the performance, since they complement each other to certain degree.

Semantic-B: Learning to Rank

Because of the nature of this task, which is to give an ordered list of recommended tweets, in the Semantic-B method, we design the integration of different semantic types as a learning to rank problem. Learning to rank algorithm can rank the candidates according to their relevance or other factors to the target, which is an event summary in our case. It has been used in many general recommendation tasks, and also in some tweet related tasks. The features we consider in this approach included all the seven types of semantic terms described before, and the two normalization factors for handling retweet and tweet length, e.g. $log(1{+}r)$ and $log(1{+}n)$ described in Equation 3. For the semantic terms in a tweet, we calculate their feature values by probability distributions, using the same equation, Equation 1, described in Semantic-A.

Putting all the features and factors together, we will have multiple candidate lists with different candidates or different orders. A learning to rank algorithm is used to build a ranking model based on the training data, and this model is used on the test data set. The learning to rank algorithm used in this study is extended from LibLinear (Lee and Lin 2014).

Graph Neural Network Based Approach

We describe the graph-based neural model in this section. Figure 1 shows the high-level structure for estimating tweet salience. For a given event cluster, a tweet relation graph is first built, then GCN networks are applied on the relation graph, and finally a salience score is calculated for each tweet by considering both the tweet hidden state and also the cluster embedding that represents the whole event cluster. We describe each component below.

Tweet Representation. We fine-tune the pre-trained BERT language model (Devlin et al. 2019) to generate tweet representation. BERT, which is based on a multi-layer bidirectional Transformer, is a neural network language model trained on large corpuses of text data. It has generally been applied to specific natural language understanding tasks by fine-tuning a final output layer for each task, and has improved the state-of-the art performance across the board for natural language understanding tasks. In our graph-based model, given a tweet, after removing URLs and retweet mentions, its text is tokenized using WordPiece and fed into the BERT model. The [CLS] token vector of the final output layer is taken as the tweet representation. This vector is then used by both the tweet relation graph component and the GCN component, shown in Figure 1.

Tweet Relation Graph. The underlying idea behand utilizing a relation graph is that a tweet is important in an event cluster if it has strong connections with other important tweets in this event. In this graph, a node is a tweet, and an edge is the relation between two tweets, measured by their similarity. Following previous studies (Erkan and Radev 2004; Cao et al. 2017), we also use cosine similarity to measure the connection strength between node, using tweet embeddings. An edge is added between two nodes if the cosine value is greater than the threshold, obtained empirically.

Another factor that affects the importance of a tweet on representing an event is how informative the tweet is. As described earlier, an event is usually defined by the *4Ws* questions. Therefore, we calculate a score for each node by

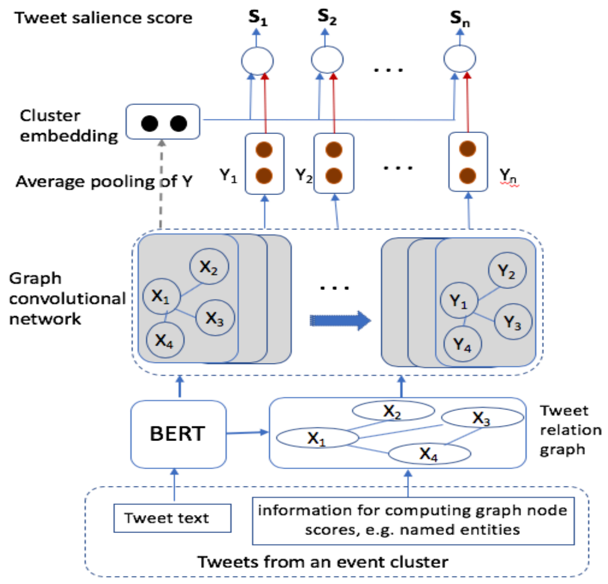


Figure 1: The tweet salience estimation workflow.

considering the semantic terms it contains, and then use these scores to update the graph edges. Using these terms as input, we apply linear regression, which has been used in (Christensen et al. 2013), to each tweet *x* to obtain its score, *s(x)*. Then for a directed edge of (x_i, y_j), its weight is transformed by the score of node x_i, and then normalized by the total outgoing edge scores, using this equation:

$$w_{new}(x_i, x_j) = \frac{w(x_i, x_j)\, s(x_i)}{\sum_{j=1,\ldots,N} w(x_i, x_j)\, s(x_i)} \qquad (4)$$

where $w(x_i, x_j)$ is the original edge score from node x_i to x_j, $s(x_i)$ is the score of node x_i, and $w_{new}(x_i, x_j)$ is the normalized score. All edges are then transformed to undirected, the edge weights are updated to the average of the weights of both directions, and finally they all are rescaled to 0 to 1.

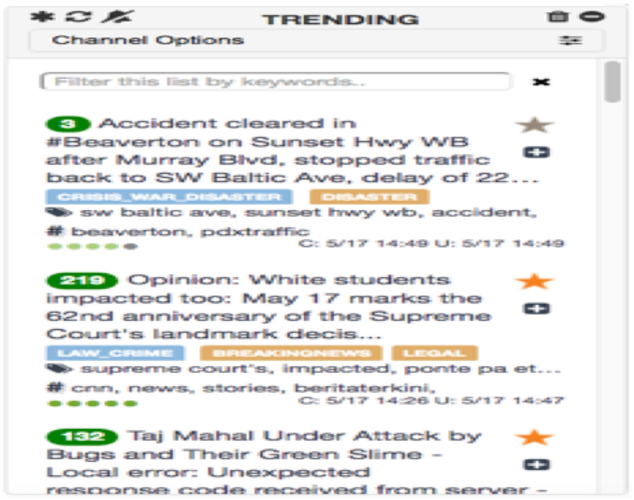
Graph Convolutional Network. GCN networks are applied over the tweet relation graph. What GCN does is to learn a transformation function *f(X,E)* to produce hidden features for each node, taking the tweet relation graph nodes and edges as input. This function uses the following layer-wise propagation rule:

$$H^{(l+1)} = \sigma\left(\frac{1}{\sqrt{\tilde{D}}}\, \tilde{A}\, \frac{1}{\sqrt{\tilde{D}}}\, H^{(l)} W^{(l)}\right) \qquad (5)$$

Tweet Salience Estimation. The GCN component learns new tweet embeddings, e.g. $(Y_1, Y_2, \ldots, Y_n)$ in Figure 1. In order to get a global context of an event, a cluster-level embedding, called *C*, is generated by averaging the embeddings of all the tweets in this cluster. Then the salience score of tweet x_i is computed by using both the cluster embedding *C* and the tweet embedding Y_i:

$$S_i = f(x_i) = v^T \tanh\,(W_s\, Y_i + W_c\, C) \qquad (6)$$

where *Ws, Wc* and *v* are parameters. S_i is then normalized over all the tweets of this cluster, via a softmax operation. Previous study (Yasunaga et al. 2017) has used similar approach for multi-document summarization.


TRENDING

Channel Options

Filter this list by keywords..

3 Accident cleared in #Beaverton on Sunset Hwy WB after Murray Blvd, stopped traffic back to SW Baltic Ave, delay of 22…

CRISIS_WAR_DISASTER DISASTER

sw baltic ave, sunset hwy wb, accident,

beaverton, pdxtraffic

C: 5/17 14:49 U: 5/17 14:49

219 Opinion: White students impacted too: May 17 marks the 62nd anniversary of the Supreme Court's landmark decis…

LAW_CRIME BREAKINGNEWS LEGAL

supreme court's, impacted, ponte pa et…

cnn, news, stories, beritaterkini,

C: 5/17 14:26 U: 5/17 14:47

132 Taj Mahal Under Attack by Bugs and Their Green Slime - Local error: Unexpected response code received from server -

Figure 2: Screenshot of event summary examples

Deployment. Figure 2 shows a preliminary user interface example which displays three events with summary. The number next to the summary is the total number of tweets of that event, and it can change dynamically as the system detects more tweets for that event. A new panel will pop up to show all the tweets of an event if its number is clicked by user.

Experiments and Results

Data Collection

Previous studies (Alsaedi et al. 2016; Inouye and Kalita 2016) have used the data set created by Inouye and Kalita, which has 50 events. Shen et al. (2013) use six NBA games as their event data set. Chua and Asur (2013) use four events, built by searching Twitter using some keywords. All the events in these data sets were not detected at real-time via Twitter's streaming data; they all belong to retrospective and specified events (Atefeh and Khreich 2013). The latter two data sets are not available to the public.

Due to the above reasons, we decided to create our own data set. Reuters Tracer is a real-time event detection system on Twitter (Liu et al. 2016; Liu et al. 2017). It clusters tweets talking about the same story into the same cluster at real-time, by considering tweet link, entity name, hashtag, etc. Their evaluation shows that the system has the state-of-the-art real-time event detection/clustering performance. We used the events generated by this system in our study, and collected 1,000 events from it. They include both big events, such as bombing, and small events, such as an NBA game.

Six human annotators were trained in social media news events, and reviewed these events. They were asked to go through all the tweets of an event to select the most *representative* and *informative* ones as the "true" summary. Because tweets are very short, it was not too time-consuming for the annotators to identify the appropriate tweets. The experts may choose more than one tweet as the summary. If multiple tweets are selected for an event, they should not convey repetitive message, instead they should represent different aspects or development stages of the event. Overall, 72% of the events have one tweet chosen as summary, 27% of them have two or three tweets, and less than 1% of them have more than three tweets.

Algorithm Performance Comparison

Evaluation Metrics. We use both ROUGE (Lin and Hovy 2003) and BLEU-4 (Papineni et al. 2002) as the evaluation metrics. ROUGE works by comparing an automatically produced system summary against the reference summary. It basically measures the overlap of N-grams between the system and reference summaries. However, it does not tell you much as a metric. To get a good quantitative value, in the context of ROUGE, we compute precision and recall using the overlaps, and then report the F1-measure. This metric has been used in previous studies (Alsaedi et al. 2016; He and Duan 2018). In this study, we use ROUGE-1, which is based on unigram overlap, and ROUGE-2, which is based on bigram overlap, to compute the F1 score.

Algorithms Compared. We compare our approaches to these methods:

1. *The earliest tweet*: this is the first tweet in the event. This tweet is the first one describing the event, and so it is a special one.
2. *The most retweeted tweet*: retweeting reinforces a message and the number of retweets shows its popularity.
3. *sumBasic*: previous studies have shown that sumBasic is a simple but effective method (Alsaedi et al. 2016; He and Duan 2018).
4. *Hybrid TF-IDF*: this is the state-of-the-art approach on tweet event summarization (Inouye and Kalita 2016).
5. *Multi-Perceptron*: This method uses a multi-layer perceptron neural network to estimate a probability score for each tweet and rank them (Wang and Zhang 2017).

The earliest and most retweeted methods are two simple but intuitive approaches. Previous studies (Alsaedi et al. 2016; Inouye and Kalita 2016) already demonstrated that sumBasic and Hybrid TF-IDF outperform other methods, such as LexRank, Mead, TextRank. Therefore, in the experiments, our approaches are compared to these two approaches, and the Multi-Perceptron method that is based on neural network. The approach proposed by (He and Duan 2018) is based on social network structure, and it is not suitable for real-time event summarization, since usually it is impossible for a summarization model to build the social media network structure at real time, due to the real-time constraint and lack of access to the whole Twitter network. Therefore, we did not include this approach in our experiments.

Training Settings

Semantic-A Training. We used 700 events and the logistic regression algorithm to learn the weights of semantic types in Equation 2, using a 5-fold cross-validation, and the other 300 events as evaluation data. The seven semantic types are the features, and their values from Equation 1 are the feature values. For each of the training events annotated with true summary, we randomly selected 5 tweets from this event as the false summary. The coefficient of each semantic type is used as the weight in Equation 2.

Semantic-B Training. The 700 events mentioned in the last section are also used to train the learning to rank model. If an annotated event has more than one tweet as summary, then they were already ranked by our annotators, and so we can get the ranks between them. For each of the training events, we randomly selected 5 tweets from this event as negative summary examples. These negative tweets have the same rank among themselves.

Graph-based Approach Training. The graph-based model is trained end-to-end. The same 700 events are used as the training and validation data. We fine tune the pre-trained BERT-base model for generating tweet representation. The embedding dimension size is 768. We train the model using an Adam optimizer with β_1 = 0.9 and β_2 = 0.999, and use a dynamic learning rate during the training

Method	ROUGE F	BLEU
Earliest	0.511	0.384
Most retweeted	0.540	0.413
Hybrid TF-IDF	0.572	0.437
SumBasic	0.588	0.448
Multi-Perceptron	0.593	0.450
Semantic-A	0.607	0.456
Semantic-B	0.622	0.465
Graph based	0.647	0.490

Table 1. Comparison of different algorithms

process. The number of GCN layers, shuffled mini-batch size, dropout rate, and learning rate are 3, 25, 0.3 and 0.001, respectively. The graph model is trained by minimizing the cross-entropy loss between the predicted tweet salience score and its ROUGE F1 score:

$$l = -\sum_D \sum_{x_i} \log(f(x_i))\, R(x_i) \qquad (7)$$

where D is the whole training data, x_i is a tweet in a cluster, $f(x_i)$ is the salience score of x_i, and $R(x_i)$ is the average F1 score of ROUGE-1 and ROUGE-2, after normalized over all the tweets of the cluster, through a softmax operation. Similar approaches are also used in previous studies (Wang and Zhang 2017; Yasunaga et al. 2017; He and Duan 2018).

Number of Tweets Selected as Summary

Some events may need to have more than one tweet as their summaries. In this case, each tweet represents a different aspect or development stage of the event. In this study, whether multiple tweets should be selected is determined as follow: all the tweets are first ordered in descending order based on their scores produced by the corresponding model. The tweet ranked first is selected as part of the summary first, and then from the second one, each tweet is checked and is also selected as part of the summary if it meets the following two conditions: 1. Its cosine similarity with any of the already selected tweets is less than a threshold *C1*. This condition is to make sure that the next selected tweet talks about a different aspect of the event from the already selected tweets. 2. The percentage of tweets that are similar to this one (measured by cosine similarity, and the threshold is *C2*) should be greater than a threshold *P*. This is to make sure that this tweet is representative enough. The values learned from training data are: *C1* = 0.68, *C2* = 0.81, *P* = 16%. They are independent of individual summarization algorithms. The baseline approaches only select one tweet as summary by their design. To make the comparison fair, we did two comparisons: 1. Each method just selects one tweet as summary. 2. To be able to select multiple tweets as summary, the other methods also used the learned values for *C1, C2* and *P* to determine if more tweets should be selected.

Comparison Result and Discussion

Table 1 presents the result of the first comparison, based on the average F1 scores of ROUGE-1 & ROUGE-2, and BLEU-4, using the 300 evaluation events. It shows that Semantic-B and the GCN based approach outperform other methods, and the differences are statistically significant at *p=0.05* level using *t-test* (Rice 2006). Table 1 also shows that Semantic-B, which is based on the learning to rank algorithm, achieves better performance than Semantic-A, which is based on integrating all the semantic features into one final score. Compared to the ROUGE and BLEU values reported in some previous studies on other summarization data sets, the scores in this study may look high. One reason is that tweets are very short and the ones in the same event are similar, which makes the scores higher.

Table 2 presents the result when we can choose more than one tweet as summary. We can see that the overall performance increases, compared to selecting only one tweet. This shows that our designed parameters, *C1, C2* and *P,* do improve the performance.

Ablation Test. In the GCN based method, when building the tweet relation graph, we use the semantic terms to calculate a score for each node, and then use it to update the edge weight, in order to include the *informative* aspect of a tweet in the tweet salience estimation. To show how important it is, we did an extra experiment, in which the edge weight is not updated by the node score in the tweet relation graph. When computing the tweet salience score, in order to get a global context of an event, a cluster-level embedding is generated by averaging the embeddings of all the tweets in this cluster, and the salience score is computed by using both the cluster embedding and the tweet embedding. To show how important the cluster level embedding is, we also conducted an experiment without using the cluster level embedding.

Table 3 presents the results of these two experiments, compared to the proposed graph-based model. The results show that the both the ROUGE and the BLEU scores had a 3% decline when the node weight is not updated using semantic terms, but it is still slightly better than the two methods based on semantic classes. This demonstrated that explicitly including the semantic terms in node relation creation do help in tweet salience estimation. When excluding the cluster level embedding in computing tweet salience score, the performance decreases greatly, from 0.658 to 0.620, more than 6%. It shows that the cluster level embedding provides very useful information on estimating whether a tweet is representative or not.

Method	ROUGE F	BLEU
Earliest	0.522	0.391
Most retweeted	0.550	0.423
Hybrid TF-IDF	0.579	0.442
SumBasic	0.596	0.456
Multi-Perceptron	0.603	0.459
Semantic-A	0.618	0.468
Semantic-B	0.630	0.476
Graph based	0.658	0.498

Table 2. Performance when multiple tweets may be chosen

Method	ROUGE F	BLEU
Graph based - without node weight update using semantic terms	0.637	0.477
Graph based - without cluster level embedding for salience score estimation	0.620	0.466
Graph based	0.658	0.498

Table 3. Ablation test result

Conclusion

This paper presents two summarization approaches for events detected from Twitter. The first one utilizes the semantic types of event related terms, and ranks the tweets based on the score computed from analyzing these semantic terms. The second one employs GCN and tweet relation graph to generate tweet hidden features for tweet salience estimations. Experiments on a large data set show that our two approaches outperform other summarization methods.

References

Alsaedi, N.; Burnap, P.; Rana, O. 2016. Automatic summarization of real world events using Twitter. The International AAAI Conference on Web and Social Media (ICWSM 2016).

Atefeh, F.; and W. Khreich. 2013. A survey of techniques for event detection in Twitter, Computational Intelligence.

Becker, H.; Naaman, M.; and Gravano, L. 2011. Selecting quality Twitter content for events. The International AAAI Conference on Web and Social Media (ICWSM 2011).

Cao, Z.; Li, W.; Li, S.; and Wei, F. 2017. Improving multi-document summarization via text classification. The Thirty-first AAAI Conference on Artificial Intelligence (AAAI 2017).

Cao, Z.; Li, W.; Li, S.; Wei, F.; and Li, Y. 2016. Attsum: Joint learning of focusing and summarization with neural attention. International Conference on Computational Linguistics (COLING 2016).

Chellal, A.; Boughanem, M.; and Dousset. B. 2017. Word similarity-based model for tweet stream prospective notification. The annual European Conference on Information Retrieval (ECIR 2017).

Chang, Y.; Wang, X.; Mei, Q.; Liu, Y. 2013. Towards twitter context summarization with user influence models. International Conference on Web Search and Data Mining (WSDM 2013).

Chang，Y.; Tang, J.; Yin, D.; Yamada, M.; and Liu, Y. 2016. Timeline summarization from social media with life cycle models. International Joint Conferences on Artificial Intelligence (IJCAI 2016).

Cheng, J.; and Lapata, M. 2016. Neural summarization by extracting sentences and words. Annual meeting of the Association for Computational Linguistics (ACL 2016).

Christensen, J.; Soderland, S; and Etzioni, O. 2013. Towards coherent multidocument summarization. Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2013).

Chua, F.; and Asur, S. 2013. Automatic summarization of events from social media. The International AAAI Conference on Web and Social Media (ICWSM 2013).

Devlin, J.; Chang, M.W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019).

Erkan, G.; and Radev, D. 2014. Lexrank: graph-based centrality as salience in text summarization, Journal Of Artificial Intelligence Research.

He, R.; and Duan, X. 2018. Twitter summarization based on social network and sparse reconstruction. The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018).

Inouye, D.; and Kalita, J.K. 2016. Comparing summarization algorithms for multiple post summaries. Social Computing 2016.

Kipf, T.N.; and Welling, M. 2017. Semisupervised classification with graph convolutional networks. The International Conference on Learning Representations (ICLR 2017).

Lee, H.P; and Lin, C.J. 2014. Large-scale linear rankSVM. Neural Computation.

Li, Q.; Nourbakhsh, A.; Shah, S.; and Liu, X. 2017. Real-Time Novel Event Detection from Social Media, IEEE International Conference on Data Engineering (IEEE ICDE 2017).

Lin, C.-Y.; and Hovy, E. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. 2003 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2003).

Liu, X.; Li, Q.; Nourbakhsh, A.; et. al. 2016.Reuters Tracer: A Large Scale System of Detecting & Verifying Real-Time News Events from Twitter, The Conference on Information and Knowledge Management (CIKM2016).

Liu, X.; Nourbakhsh, A.; Li, Q.; et. al. 2017. Reuters Tracer: Toward Automated News Production Using Large Scale Social Media Data, 2017 IEEE International Conference on Big Data (IEEE BigData 2017).

Liu, Y.; Titov, I.; and Lapata, M. 2019 Single document summarization as tree induction. 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019).

Ma, X.; and Hovy, E. 2016. End-to-end Sequence labeling via bi-directional LSTM-CNN-CRF, The 54th annual meeting of the Association for Computational Linguistics (ACL 2016).

Mihalcea, R.; and Tarau, P. 2004. TextRank: Bringing order into texts, Conference on Empirical Methods in Natural Language Processing (EMNLP 2004).

Mishra, A.; and Berberich, K. 2017. Event digest: A holistic view on past events, International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017).

Mohd, N. 2007. Named entity patterns across news domains. 1st BCS IRSG conference on Future Directions in Information Access.

Nallapati, R.; Zhou, B.; Gulcehre, C.; et. al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. Conference on Computational Natural Language Learning (CoNLL 2016).

Narayan, S.; Cohen, S.B.; and Lapata, M. 2018. Ranking sentences for extractive summarization with reinforcement learning. 2018 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2018).

Owoputi, O.; O'Connor, B.; Dyer, C.; Gimpel, K.; Schneider, N.; et. al. 2013. Improved part-of-speech tagging for online conversational text with word clusters. 2013 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2013).

Papineni, K.; Roukos, S.; Ward, T; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. Annual meeting of the Association for Computational Linguistics (ACL 2002).

Radev, D.; Allison, T.; Blair-Goldensohn, S.; Blitzer, J.; et al. 2004. Mead – a platform for multidocument multilingual text summarization, The International Conference on Language Resources and Evaluation (LREC 2004).

Rice, J.A. 2006. Mathematical Statistics and Data Analysis, Third Edition.

See, A.; Liu, P.J.; and Manning, C.D. 2017. Get to the point: Summarization with pointer generator networks. Annual Meeting of the Association for Computational Linguistics (ACL 2017).

Sequiera, R.; Tan, L.; and Lin, J. 2018. Overview of the TREC 2018 Real-Time Summarization Track, TREC 2018.

Shen, C.; Liu, F; Weng, F.; and Li, T. 2013. A participant-based approach for event summarization using Twitter-Streams. Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2013).

Vanderwende, L.; Suzuki, H.; Brockett, C.; and Nenkova, A. 2007. Beyond SumBasic: Task-focused summarization with sentence simplification and lexical expansion, IPM.

Wang, L.; Ling, W. 2016. Neural network abstract generation for opinions and arguments. 2016 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2016).

Wang, Z.; and Zhang, Y. 2017. A neural model for joint event detection and summarization, International Joint Conferences on Artificial Intelligence (IJCAI 2017).

Yasunaga, M.; Zhang, R.; Meelu, K.; et al. 2017. Graph-based Neural Multi-Document Summarization, Conference on Computational Natural Language Learning (CoNLL 2017).Conference Name:ACM Woodstock conference

Yadav, V.; Bethard, S. 2018. A Survey on Recent Advances in Named Entity Recognition from Deep Learning models. International Conference on Computational Linguistics (COLING 2018).

Yadav, V.; Sharp, R.; Bethard, S. 2018. Deep affix features improve neural named entity recognizers. Conference on Lexical and Computational Semantics 2018.

Zhang, X.; Lapata, M.; Wei, F.; and Zhou, M. 2018. Neural latent extractive document summarization. Conference on Empirical Methods in Natural Language Processing (EMNLP 2018).

Zhou, Q.; Yang, N.; Wei, F.; Huang, S.; et al. 2019. Neural document summarization by jointly learning to score and select sentences. Annual Meeting of the Association for Computational Linguistics (ACL 2019).