Word Embedding Based Event Detection on Social Media

Ali Mert Ertugrul[1], Burak Velioglu[2], and Pinar Karagoz[2](✉)

[1] Informatics Institute, METU, 06800 Ankara, Turkey
alimert@metu.edu.tr
[2] Computer Engineering Department, METU, 06800 Ankara, Turkey
{velioglu,karagoz}@ceng.metu.edu.tr

Abstract. Event detection from social media messages is conventionally based on clustering the message contents. The most basic approach is representing messages in terms of term vectors that are constructed through traditional natural language processing (NLP) methods and then assigning weights to terms generally based on frequency. In this study, we use neural feature extraction approach and explore the performance of event detection under the use of word embeddings. Using a corpus of a set of tweets, message terms are embedded to continuous space. Message contents that are represented as vectors of word embedding are grouped by using hierarchical clustering. The technique is applied on a set of Twitter messages posted in Turkish. Experimental results show that automatically extracted features detect the contextual similarities between tweets better than traditional feature extraction with term frequency - inverse document frequency (TF-IDF) based term vectors.

Keywords: Event detection · Neural feature extraction · Word embedding · Neural probabilistic language models

1 Introduction

Social media has become a basic tool for communication among the Internet users. Micro-blogging platforms enable users to broadcast digital contents including texts, images and videos. Twitter is currently the most popular micro-blogging platform in which people can share their experiences and ideas. Although the twitter messages, tweets, are limited to 140 characters, they constitute an important source of information. Additionally, the propagation of the short information is easier and faster.

Tweets are the time-stamped information and Twitter can be considered as an up-to-date source of event related messages [1]. Individuals who are taking part in or watching an event tend to share a number of event-relevant messages and communicate with each other to exchange opinions about that event [2]. Therefore, it is expected that when an important event, such as a disaster, political election and football game, occurs, the number of the tweets related to that event considerably increases. Consequently, analysis and extraction of event related information from social media resources enable individuals to obtain important knowledge faster and easier.

Event detection can be described as a clustering process of the similar indicators. In our case, tweets are used as indicators. Yet, clustering the semantically similar tweets is not a straightforward task for the researchers. The conventional approach is to construct term vectors representing tweets and calculating the similarity between term vectors. There are several ways to assign weights to terms, yet the mostly used weight assignment schema is using TF-IDF of the tweet terms.

In this study, our purpose is to detect events from Twitter using word-embedding based representation of tweets. To this aim, we represent each word of each tweet as a continuous vector utilizing the word2vec model [3]. Each tweet is represented in terms of vector representations of its words. Next, we cluster the tweet representations with hierarchical clustering methods.

The technique is applied on a set of collected tweets in Turkish. In addition to difficulties due to informal language, working on a morphologically complex and agglutinative language poses challenges in traditional NLP tasks. Events detected under word-embedding based representation is compared against TF-IDF based term vector representation. Experimental analysis show that word embedding based clustering improves event detection performance.

The contributions of this work can be summarized as follows: Word embeddings are used for event detection on micro-blogging platform, which uses short message length and generally involves informal use of language. The technique is applied on Turkish, which is an agglutinative and morphologically complex language. The technique is experimentally analyzed on a collected set of tweets including four different events, where two of them are unexpected events such as a terrorist attack, and two other events are scheduled events such as a celebration. Hence, the performance can be analyzed according to the nature of the event.

This paper is organized as follows. In Sect. 2, related work is summarized. In Sect. 3, proposed method is described. In Sect. 4, experiments on the performance of the proposed method are presented. The paper is concluded with an overview in Sect. 5.

2 Related Work

In the literature, there exists a number of studies related to event detection and retrieval from social media data, especially Twitter. These studies can be classified into three categories according to their detection approaches, namely supervised, unsupervised and hybrid [4]. Among the studies employing supervised approach, Popescu et al. extract the features including part-of-speech tags, regular expressions and relative positional information, then uses gradient boosted decision trees to identify controversial and noncontroversial events [5]. Similarly, Sakaki et al. detect domain specific events like earthquake, which are manually labeled, using SVM classifier [6]. There also exists studies for event extraction from social media platforms using other types of supervised approaches like random forest and logistic regression [7,8]. Supervised approaches are mostly used for specified event detection. However, labeling tweet messages by annotators is time consuming and requires intensive work load.

Unsupervised approaches for event detection and retrieval are generally based on cluster analysis. Some studies employ incremental clustering algorithm based on a similarity threshold to form clusters [9,10]. These studies consider the features containing number of tweets, users and term vectors while clustering. Likewise, Ozdikis et al. apply agglomerative clustering technique to detect events in Turkish, using words with and without semantic expansions as tweet vectors [11,12]. Parikh et al. propose an event detection method by exploring textual and temporal components of the tweets so that events are detected using hierarchical clustering technique [13]. Additionally, several studies employ graph-based clustering for the detection of new events including detection techniques; hierarchical divisive clustering [14], community detection [15], and wavelet analysis and graph partitioning [16].

In addition to these approaches, there are also studies combining both supervised and unsupervised approaches for event detection. Becker et al. use both online clustering and SVM classifier to distinguish the messages belonging to real world events and non-event messages [17]. Moreover, Hua et al. [18] employ semi-supervised approach to detect targeted events like crimes and civil unrests.

The studies in the literature related to event detection and retrieval can also be analyzed in terms of the features they employ. Apart from utilization of hand crafted features, word embedding techniques are used as a feature extraction process for many problems in NLP literature. Tan et al. [19] make a lexical comparison between Twitter and Wikipedia corpora by pursuing a linear relation between obtained word embeddings. To extract Adverse Drug Reactions (ADR), Lin et al. [20] feed the vectors obtained by word2vec algorithm into conditional random fields. To learn a sentiment-specific word embeddings, Tang et al. [21] proposed their own word embedding algorithm and compare their results with classical word2vec algorithm. Besides these studies, Fang et al. [22] utilize word embedding technique to evaluate the coherence of topics from Twitter data.

Number of the studies related to event detection learn features from the sentences instead of using hand crafted features. Among them, Nguyen et al. use Convolutional Neural Network (CNN) to detect events [23]. In [24], Nguyen et al. use deep neural networks to both identify informative tweets and classify them into classes in an online fashion to detect crisis from tweets. As far as we know, neural feature extraction of the words for event detection on Turkish has not been applied before. However, there are studies in which learning continuous vector representations of words is used to develop a Named Entity Recognition (NER) system for Turkish [25,26].

3 Proposed Method

In this work, we propose an event detection approach based on clustering continuous vector representations obtained by word embedding. We use an agglomerative clustering technique with time constraint.

3.1 Data Collection

In order to access Twitter data, Twitter provides two general APIs namely REST API[1] and Streaming API[2]. We use REST API to create a corpus. Using this dataset, we aimed to learn continuous representation of the tokens (words) in tweets by using word embedding technique. We gathered nearly 2.1M tweets for the corpus during three days. On the other hand, we use Streaming API to collect messages posted between April 23rd, 2016 and May 10th, 2016, which is used for event detection. For event detection, more than 1 M tweets have been collected within 18 days. During the collection of data for both datasets, only Turkish tweets are selected.

3.2 Data Preprocessing

We start with removing the tweets including *"I'm at"* text since they are the Foursquare[3] related ones showing the check-in activities of users. We remove the words whose length is less than 2 characters (except numbers), hashtags, mentions, links and *"RT"* keyword. We also remove a number of emoticons. In order to detect and remove various types of emoticons, we use the java library *com.vdurmont*[4]. Furthermore, we omit a number of punctuation marks including (*- + . , ? ! % * / : ; [ ] { } =*). We also filter out the stop words from the tweets using the list of Turkish stop words provided by *Lucene*[5].

After basic removal and parse operations, we fix misspelled words and obtain word stems. To this aim, we use *Zemberek API*[6], which is a general purpose natural language processing library and toolset designed for Turkish language. Turkish is an agglutinative language. We perform stemming on all words of all tweets we collect (both corpus dataset and event detection dataset) and then separate them into form of word stem, derivational affixes and inflectional suffixes using Zemberek API. Inflectional suffixes do not change the meaning of the words, yet they diversify the words. Due to this reason, we extract stems of the words by omitting inflectional suffixes. As a result of preprocessing step, the test dataset has 881 K tweets.

[1] https://dev.twitter.com/rest/public.
[2] https://dev.twitter.com/streaming/public.
[3] https://foursquare.com/.
[4] https://github.com/vdurmont/emoji-java.
[5] http://lucene.apache.org/.
[6] https://github.com/ahmetaa/zemberek-nlp.

3.3 Word Embedding

In this study, continuous vector representation of each word is obtained by word2vec algorithm [3], which is one of the Neural Probabilistic Language Model (NPLM)-based models. This representation can be obtained by either predicting the word itself using its neighbor words or predicting neighbor words using only corresponding word. NPLM models can be trained using maximum likelihood (ML) principle. Note that, in this study representations are obtained by estimating the neighbor words using the word itself, since it is much faster than the other approach [3].

Note that optimizing the model using maximum likelihood would take $O(|V|)$ time, since maximizing the cost function with ML principle iterates over all words included in the vocabulary. So, rather than maximizing the log-likelihood over all words, lately developed word2vec model is trained utilizing negative sampling. In this approach, objective is optimized by maximizing the probability of words and contexts being in the corpus, and minimizing sampled others which are not in the context. Real words are represented by w and imaginary target words are represented by w^*. Then, the cost function can be defined as,

$$J_{NEG} = logQ_\theta(D = 1|w, h) + k\mathbf{E}[logQ_\theta(D = 0|w^*, h)], \tag{1}$$

where $Q_\theta(D = 1|w, h)$ is the logistic regression calculated in terms of the learned embedding vectors θ. h represents the matrix of embedded representations.

Each preprocessed tweet is represented as a vector using the skip-gram model [3]. Due to character limitation of Twitter, window size is used as 2. After eliminating the tweets with less than 5 words, word vectors are created by training word2vec algorithm with 1.5 million tweets. The length of the continuous vectors are selected as 150 and the number of negative words is selected as 10, empirically. That means, neighbor words are estimated with the center word projected into 150 dimensions and neighbor words are mixed randomly ten times. The five nearest neighbor words of five sample words are given in the Table 1.

Table 1. Sample words and their five nearest words. Skip-gram is trained with 1.5 M tweets

Word	Nearest 5 words
maç match	{hakem, Galatasaray, gol, seyircisiz, Beşiktaş} {referee, Galatasaray, goal, without spectators, Besiktas}
terör terror	{azdıran, bölücü, örgüt, menfez, terörist} {arouser, separatist, organization, culvert, terrorist}
bomba bomb	{roketatar, araçlı, patlat, bombala, patlama} {bazooka, with vehicle, explode, bombing, explosion}
Beşiktaş Besiktas	{şampiyon, Galatasaray, namağlup, Fenerbahçe, taraftar} {champion, Galatasaray, unbeaten, Fenerbahce, supporter}
bayram festival	{gün, işçi, kutlu, nisan, emekçi} {day, worker, blessed, April, laborer}

In order to obtain vector representation of a tweet, vector representations of its all words are summed. Note that, summation vector is divided by the length of tweet to handle tweets with varying lengths.

3.4 Clustering Algorithm

We follow a special type of agglomerative clustering technique based on time constraint for event detection. Basically, vector representation of each tweet is individually evaluated considering its time-stamp and similarity to active clusters. Then, it is included into the matching cluster or a new cluster is created for the corresponding tweet.

In order to add a tweet into an existing cluster, the following two conditions should be satisfied. Firstly, the difference between time-stamp of the latest tweet in the cluster and the tweet to be clustered should be less than or equal to the parameter $\boldsymbol{T_{max}}$. If this condition is satisfied for the given cluster, then it is called active cluster. In our experiments, this parameter is set to 3 and 6 h. Secondly, the similarity between the vector of an active cluster and the vector representation of the tweet to be clustered should be greater than or equal to $\boldsymbol{S_{min}}$. A cluster vector corresponds to cluster centroid which is the arithmetic mean of the vector representations of the tweets in that cluster. In the experiments, cosine similarity measure is used and the parameter S_{min} is set to 0.60, 0.65, 0.70 and 0.75. If there exists clusters satisfying these two conditions for a given tweet, the one with the maximum similarity value is chosen and the tweet is put into that cluster. On the other hand, a new cluster is created for the corresponding tweet unless there exists a cluster satisfying the conditions. The pseudo code of the clustering algorithm we employ is given in Algorithm 1.

Algorithm 1. Clustering Algorithm

```
for $t \in Tweets$ do
  $is\_assigned \leftarrow$ false
  $active\_clusters \leftarrow getActiveClusters(t.getTime(), T_{max})$
  if $active\_clusters \neq \emptyset$ then
    $optimum\_cluster \leftarrow findMaxSimilarCluster(t, active\_clusters)$
    if $similarity(t, optimum\_cluster) \geq S_{min}$ then
      $optimum\_cluster.add(t)$
      $optimum\_cluster.calculateNewCentroid()$
      $is\_assigned \leftarrow$ true
    end if
  end if
  if not $is\_assigned$ then
    $new\_cluster \leftarrow createNewCluster(t)$
    $all\_clusters.add(new\_cluster)$
  end if
end for
```

4 Experiments and Results

During the 18-day test data collection period between April 23rd, 2016 and May 10th, 2016, we focused on four events to analyze, where two of them are unexpected events and two other events are scheduled events as celebrations. The unexpected events are the suicide bomb attack in Bursa and gun attack to a Turkish journalist in Istanbul. On the other hand, the scheduled events are anniversaries of national sovereignty and children's day and labor day. The information about the events are presented in Table 2.

In order to identify the event clusters for each event, we firstly specify representative query words. We ask ten Turkish participants to determine the words to represent corresponding events. Based on their answers, we finalize the query sentences. For example, the query sentence for E_1 is "Bursa canlı bomba saldırı (Bursa suicide bomb attack)" as given in Table 2. For each event, we obtain word2vec representation of the query sentence and calculate the similarity between this representation and cluster vectors. Finally, if the similarity between the query sentence representation and the cluster vectors are larger than the threshold, which is specified for the clustering process, corresponding clusters are called event clusters.

In order to compare performance of word2vec representation, we obtain TF-IDF [27] based vectors for each tweet, perform clustering by using these term vectors, as the baseline. IDF weight for each word is calculated using only corpus dataset. Note that, TF-IDF based vectors are obtained after the same preprocessing step. There exist 59214 unique words included in the corpus dataset. Then, normalized TF score for each word of each test tweet is calculated. The TF weight of a word that occurs in a tweet is simply proportional to the word frequency. To represent each tweet as a feature vector, bag-of-words representation is used. In other words, each tweet is represented as a feature vector where features correspond to words in the corpus. For each word in the tweet, TF-IDF

Table 2. Description of the events

	Id	Event type	Event time	Query
Suicide bomb attack in Bursa	E_1	Unexpected	27.04.2016 17:30	Bursa canlı bomba saldırı (Bursa suicide bomb attack)
Armed attack to journalist	E_2	Unexpected	06.05.2016 17:25	[Ad, soyad] saldırı ([Name, surname] attack)
National sovereignty and children's day	E_3	Scheduled	23.04.2016 All day	23 nisan çocuk bayramı (April 23 child festival)
Labor day	E_4	Scheduled	01.05.2016 All day	1 mayıs işçi bayramı (May 1 labor day)

Table 3. Information related to clusters obtained using word2vec

		3 H				6 H			
		0.60	0.65	0.70	0.75	0.60	0.65	0.70	0.75
Number of event clusters	E_1	8	6	5	3	9	6	5	4
	E_2	26	16	11	5	37	19	16	7
	E_3	3	3	1	1	3	3	2	1
	E_4	8	7	6	3	10	9	6	5
Number of Tweets in the best cluster	E_1	323	619	393	407	492	332	343	423
	E_2	943	749	586	462	902	823	615	590
	E_3	1229	889	679	545	1552	784	727	581
	E_4	1169	1075	778	676	1297	949	915	505
Avg. size of event clusters	E_1	1220,8	963.2	444.8	274	1143.89	898.33	509	338.5
	E_2	1573.46	1641.31	324.36	864.2	1235.11	1398.37	320	513.43
	E_3	1339.7	775.67	679	545	1347	850.33	420.5	581
	E_4	768.5	662.25	578.67	624.67	579.3	583.89	585.5	374.8

scores are computed and assigned to corresponding element of feature vector. The remaining entries of the feature vector are 0. Using this representation, each tweet is represented by 59214 dimensional feature vector. In our implementation, each feature vector is stored sparsely as hash map to overcome memory problem.

We perform clustering experiments using word2vec and TF-IDF representations of the tweets as term vectors with the following parameters; $S_{min} \in [0.60, 0.65, 0.70, 0.75]$ and $T_{max} \in [3, 6]$ hours. During the experiments, for each event we analyze the number of event clusters, number of tweets in the best event cluster and average size of the event clusters obtained using word2vec (see Table 3) and TF-IDF (see Table 4) representations. *Best* event cluster refers to the cluster whose cluster vector is the most similar to the query sentence representation for a given event. Experiments show that the number of event clusters obtained using word embedding are less than the ones obtained by TF-IDF regardless of the parameters and event. Also, the number of the tweets in the best event cluster and average size of event clusters obtained by word embedding are extremely larger than those obtained by TF-IDF. These results reflect that, employing word embedding leads to better clusters compared to TF-IDF.

In addition to measures given in Tables 3 and 4, we compute precision scores of the best event clusters to evaluate the success and quality of clustering based on word embedding. The tweets in the best event clusters are labeled by two annotators and the precision scores are given in Table 5. The results show that the precision of the scheduled events, E_3 and E_4, increases with an increase in the parameter S_{min}, regardless of the parameter T_{max}. We also observe that the number of tweets in the best event clusters decrease as the parameter S_{min} increases. This can be inferred in such a way that people are more likely to share structured tweets for the scheduled events such as celebrations. On the other hand, it is observed that the unexpected events E_1 and E_2 exhibit a different pattern compared the scheduled events E_3 and E_4. When $T_{max} = 3$ h, the precision values of the best event clusters reach its peak value at $S_{min} = 0.65$. However, the precision values are maximum at $S_{min} = 0.70$ when $T_{max} = 6$ h.

Table 4. Information related to clusters obtained using TF-IDF

		3 H				6 H			
		0.60	0.65	0.70	0.75	0.60	0.65	0.70	0.75
Number of event clusters	E_1	9	8	3	1	9	8	2	1
	E_2	65	37	32	15	56	37	26	15
	E_3	9	8	5	4	8	8	5	3
	E_4	39	26	15	10	34	17	12	6
Number of Tweets in the best cluster	E_1	3	3	6	4	3	7	6	4
	E_2	4	4	3	2	4	4	2	3
	E_3	1	1	1	1	4	1	1	1
	E_4	1	1	1	2	1	1	1	2
Avg. size of event clusters	E_1	5	4	1.83	4	5.22	4	5	4
	E_2	2.28	1.81	1.97	1.93	2.2	3.35	1.65	2.53
	E_3	2.22	2	1.6	1.6	2.5	2	1.6	2.67
	E_4	2.15	1.88	2.8	1.7	2	1.76	3	1.67

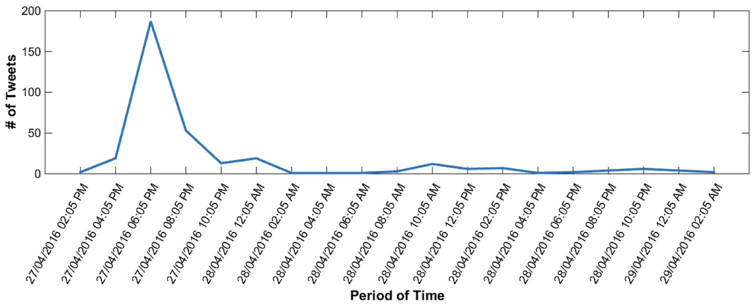
In other words, unlike in scheduled events, we cannot observe a monotonically increasing pattern with an increase in the parameter S_{min}. We observe that, after a similarity threshold (0.65 for 3 h and 0.70 for 6 h), the precision value decreases for unexpected events. We can infer that, tweets related to unexpected events are less likely to be structured as in scheduled events and less similar tweets are shared for these events.

Table 5. Precision scores of the best event clusters obtained using word2vec, with respect to the parameters T_{max} and S_{min}

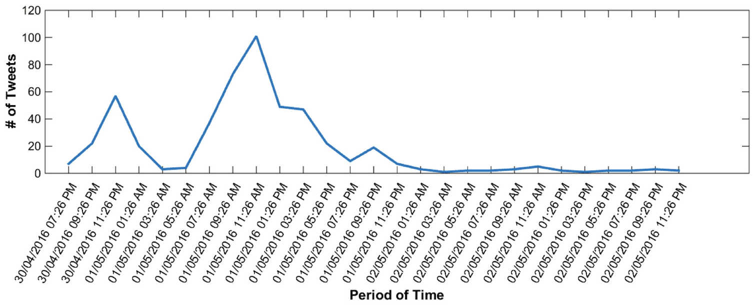
	3 H				6 H			
	0.60	0.65	0.70	0.75	0.60	0.65	0.70	0.75
E_1	62.54	**83.52**	79.39	79.61	76.42	81.33	**90.08**	84.63
E_2	85.04	**86.92**	84.13	84.20	85.03	86.03	**88.78**	83.73
E_3	50.04	76.38	82.03	**91.74**	50.19	88.78	94.49	**94.84**
E_4	89.48	82.83	93.18	**95.71**	80.26	90.94	94.86	**95.84**

Note that, precision values of the best event clusters obtained using TF-IDF are 100% for all events and conditions. The reason for this is that, event clusters contain only a few tweets which are highly similar to query sentences. Due to the large amount of data, it is time-consuming to annotate all event related tweets and calculate recall values. However, it is clear that, since event clusters consist of only a few tweets, they miss a large amount of event related tweets. It results in very low recall values obtained using TF-IDF compared to word embedding.

In order to analyze temporal information of the tweets belonging to the best event clusters, we plot the figures revealing the number of tweets shared within



(a) E_1, ($S_{min} = 0.70$ and $T_{max} = 6$ hours)



(b) E_4, ($S_{min} = 0.75$ and $T_{max} = 6$ hours)

Fig. 1. The number of tweets shared within the periods of 2 h for each of the best event clusters obtained by the corresponding parameters S_{min} and T_{max}

the periods of 2 h. Due to space limitation, we do not present the figures for E_2 and E_3 since their behaviors are similar to E_1 and E_4, respectively These best event clusters are the ones having the highest precision values where $T_{max} = 6$ h. We compare the original event times given in Table 2 with the corresponding times of number of shared tweets in Fig. 1. We observe that the tweets related scheduled events are shared during the all-day and even the beginning of the following day. We also observe that tweets related to scheduled event E_2 are posted starting from the last hours of the previous day. The number of the shared tweets for the same event are minimum between two peaks since people are most likely to sleep in that interval (03:26 AM – 07:26 AM). Moreover, the unexpected events E_3 and E_4 gets their peak values just after the corresponding original event times.

5 Conclusion

In this study, we explore the effectiveness of word-embedding based representation of tweets to detect events from Twitter. In that regard, the words of the tweets are represented as a continuous vector employing the *word2vec* model. We analyze the proposed method on a set of collected tweets in Turkish, which contains four events, in comparison to the TF-IDF based vector representations of tweets. The results reflect that utilizing word2vec features improves the performance on event detection on Twitter although it includes mostly short-length and informal data.

As a future work, we are planning to compare the performance of different continuous vector representations of words techniques such as GloVe [28] in event detection in Turkish. Additionally, we will analyze the effect of employing embedding techniques at different granularity namely *sentence2vec* and *doc2vec* [29], which are extensions of word2vec algorithm, for the event detection on micro-blogging platforms.

References

1. Goodchild, M.F.: Citizens as sensors: the world of volunteered geography. GeoJournal **69**(4), 211–221 (2007)
2. Abdelhaq, H., Sengstock, C., Gertz, M.: EvenTweet: online localized event detection from twitter. Proc. VLDB Endowment **6**(12), 1326–1329 (2013)
3. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
4. Atefeh, F., Khreich, W.: A survey of techniques for event detection in Twitter. Comput. Intell. **31**(1), 132–164 (2015)
5. Popescu, A-M., Pennacchiotti, M., Paranjpe, D.: Extracting events and event descriptions from Twitter. In: Proceedings of the 20th International Conference Companion on World Wide Web, pp. 105–106 (2011)
6. Sakaki, T., Okazaki, M., Matsuo, Y.: Earthquake shakes Twitter users: real-time event detection by social sensors. In: Proceedings of the 19th International Conference on World Wide Web, pp. 851–860 (2010)
7. Kallus, N.: Predicting crowd behavior with big public data. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 625–630 (2014)
8. Reschke, K., Jankowiak, M., Surdeanu, M., Manning, C.D., Jurafsky, D.: Event extraction using distant supervision. In: LREC, pp. 4527–4531 (2014)
9. Phuvipadawat, S., Murata, T.: Breaking news detection and tracking in Twitter. In: 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), vol. 3, pp. 120–123 (2010)
10. Petrović, S., Osborne, M., Lavrenko, V.: Streaming first story detection with application to Twitter. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 181–189 (2010)
11. Ozdikis, O., Senkul, P., Oguztuzun, H.: Semantic expansion of Tweet contents for enhanced event detection in Twitter. In: 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 20–24 (2012)
12. Ozdikis, O., Senkul, P., Oguztuzun, H.: Context based semantic relations in Tweets. In: Can, F., Özyer, T., Polat, F. (eds.) State of the Art Applications of Social Network Analysis. Lecture Notes in Social Networks, pp. 35–52. Springer, Switzerland (2014)
13. Parikh, R., Karlapalem, K.: ET: events from Tweets. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 613–620 (2013)

14. Long, R., Wang, H., Chen, Y., Jin, O., Yu, Y.: Towards effective event detection, tracking and summarization on microblog data. In: Wang, H., Li, S., Oyama, S., Hu, X., Qian, T. (eds.) WAIM 2011. LNCS, vol. 6897, pp. 652–663. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23535-1_55
15. Sayyadi, H., Hurst, M., Maykov, A.: Event detection and tracking in social streams. In: ICWSM (2009)
16. Weng, J., Lee, B-S.: Event detection in Twitter. In: ICWSM, vol. 11, pp. 401–408 (2011)
17. Becker, H., Naaman, M., Gravano, L.: Beyond trending topics: real-world event identification on Twitter. In: ICWSM, vol. 11, pp. 438–441 (2011)
18. Hua, T., Chen, F., Zhao, L., Lu, C-T., Ramakrishnan, N.: STED: semi-supervised targeted-interest event detection in Twitter. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1466–1469 (2013)
19. Tan, L., Zhang, H., Clarke, C.L.A., Smucker, M.D.: Lexical comparison between Wikipedia and Twitter corpora by using word embeddings. In: Short Papers, vol. 2, p. 657 (2015)
20. Lin, W-S., Dai, H-J., Jonnagaddala, J., Chang, N-W., Jue, T.R., Iqbal, U., Shao, J.Y-H., Chiang, I-J., Li, Y-C.: Utilizing different word representation methods for twitter data in adverse drug reactions extraction. In: 2015 Conference on Technologies and Applications of Artificial Intelligence (TAAI), pp. 260–265 (2015)
21. Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., Qin, B.: Learning sentiment-specific word embedding for Twitter sentiment classification. In: ACL, no. 1, pp. 1555–1565 (2014)
22. Fang, A., Macdonald, C., Ounis, I., Habel, P.: Using word embedding to evaluate the coherence of topics from Twitter data. In: Proceedings of SIGIR (2016)
23. Nguyen, T.H., Grishman, R.: Event detection and domain adaptation with convolutional neural networks. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, vol. 2, pp. 365–371 (2015)
24. Nguyen, D.T., Joty, S., Imran, M., Sajjad, H., Mitra, P.: Applications of online deep learning for crisis response using social media information. arXiv preprint arXiv:1610.01030 (2016)
25. Demir, H., Özgür, A.: Improving named entity recognition for morphologically rich languages using word embeddings. In: 2014 13th International Conference on Machine Learning and Applications (ICMLA), pp. 117–122 (2014)
26. Onal, K.D., Karagoz, P.: Named entity recognition from scratch on social media. In: ECML-PKDD, MUSE Workshop (2015)
27. Luhn, H.P.: A statistical approach to mechanized encoding and searching of literary information. IBM J. Res. Dev. **1**(4), 309–317 (1957)
28. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
29. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. In: ICML, vol. 14, pp. 1188–1196 (2014)