IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 5, 2020, accepted June 30, 2020, date of publication July 3, 2020, date of current version July 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3007004

An Approximate Model for Event Detection From Twitter Data

AARZOO DHIMAN AND DURGA TOSHNIWAL, (Member, IEEE)
Department of Computer Science and Engineering, IIT Roorkee, Roorkee 247667, India
Corresponding author: Aarzoo Dhiman (aarzoo@cs.iitr.ac.in)

This work was supported by the University Grant Commission (UGC), Government of India.

ABSTRACT The abundance and real-time availability of Twitter data have proved beneficial in detecting events in various domains such as emergency situations, crime detection, public health, place recommendations, etc. Nevertheless, two critical challenges occur while detecting events using social media data. First, the *uncertainty in capturing the contextual relationship* among tweets, which is the result of the limited availability of the contextual information due to the small length of tweets. Second, the *high computation cost* required in event detection due to massive data processing. Earlier research works, addressing these challenges, have tried to capture the contextual information by using the dense vector representations of texts leveraging deep neural word embedding generation models such as Word2Vec and GloVe. However, these models are trained on the Euclidean vector space which fails to amalgamate the directional information of the vectors with the semantic information in text, incurring high computational costs. To target both the problems simultaneously, we propose modeling Twitter data as a graph-of-sentences which retains the contextual relationships while maintaining lower computational cost. The proposed model captures contextual information using JoSE, a spherical vector representation leveraging the word-word and word-paragraph semantic co-occurrence statistics in a spherical generative model. Furthermore, the framework uses the weighted-graph model to capture all the relationships among the Twitter data efficiently. The graph is further pruned with the help of the graph component filtering approach. The graph clustering model, employed to detect the events, leverages the edge weights and the partial-k clustering approach maintaining low computation costs. The experimentation on the annotated benchmark Twitter data set and the real-world datasets show improved run-time performance up to 30% while maintaining the qualitative performance (F1-score) comparable to the state-of-the-art models.

INDEX TERMS Graph based event detection, social media data, uncertain clustering, word2vec, doc2vec, Jose twitter graph.

I. INTRODUCTION

Online sources such as Twitter provide a lot of meta-data, which can help in providing information such as location, time-stamps, and number of followers, etc., which has been utilized over the years to provide application-centric results. With the increase in availability of real-time online data streams, *event detection* methods have become the primary and fast mode of finding anomalous patterns or trends occurring in different communities earlier than the traditional media. Dou *et al.* [1] states that "*Collectively, events serve as a succinct summary of social media streams. Individually, an event and its sub-events reveal the evolution of certain social phenomena over time*". There are many real-time event detection based applications that use Twitter data such as disaster control [2], controversy tracking [3], health event tracking [4], detecting misleading events [5], detecting hate speech [6], predicting election [7], business events [8] and crime tracking [9] etc. The different data sources used for event detection are social media data [10], news streams [8], audio-video surveillance [11], [12], image classification [13], [14] and other multimedia data [15], [16].

The basic steps of a global event detection process are: a) pre-process the data, b) find the candidate events by grouping the tweets that are related, c) rank the candidate events to generate the actual events, and d) summarize the events to

The associate editor coordinating the review of this manuscript and approving it for publication was Vlad Diaconita.

present the tweets in a meaningful manner. In the current literature, multiple classifications, clustering, and hybrid models have been employed to perform these steps. The classification models have the limitation of the requirement of annotated data. This limitation reduces the applicability of these models on the real-time problems such as disaster situations, disease outbreak, and other unforeseen events. Clustering models, on the other hand, are the most commonly used data mining tasks for event detection. Clustering has been applied on either *tweets as data points* i.e. through the probabilistic models [17] or as the *graphical models* [18] imitating the underlying Twitter data. The two primary challenges that require attention while detecting events from social media data have been summarized below.

- *Maintaing the contextual relationship among tweets*: The real-time Twitter data collection may result in a huge size. However, the maximum allowed length of a tweet is 280 characters, where the most common length of any tweet is 33 characters. Due to the limited length, it becomes a challenge to generate the most related tweets. In this research work, this is referred to as *uncertainty* in capturing the contextual relationship among the tweets.
- *High computation cost*: The Twitter data are available in massive amounts, and the number of tweets belonging to a particular event is generally massive in number. Thus, processing all the data to its extremes may lead to redundant computations and incur high computational cost.

The event detection deals with recognizing the data points that are associated/interlinked, representing a common topic. Thus, modeling the data as a graph helps to capture the relationships among data points, making event detection efficient. Many earlier research works have employed the graph-of-words (GoW) [19] based and the graph-of-sentences (GoS) [20] based event detection models for the Twitter data. The GoW representation only uses uni-grams (words) as nodes and word co-occurrence as the relationships. However, multi-grams help to provide more contextual information while maintaining the relationships among the data points. This limitation is addressed in GoS, which uses tweets as the nodes and the cosine similarity among word vectors as edges.

The earlier research works have employed deep neural model-based dense vector representations such as Word2Vec [21], GloVe [22], and Doc2Vec [23] etc. to capture the prominent contextual information and relationships among the tweets. Though these models have shown efficient performance in capturing the contextual information in the form of position, co-occurrence, and semantics of the words and documents, the directional information is not well captured in these representations. The directional information is crucial because most of the underlying text mining tasks such as clustering, classification and association mining, etc. depend highly on the word similarity measures. The most common word similarity measure used is the cosine similarity measure, which depends on the directions of the vectors (i.e., spherical space), whereas the training of the above-stated word embedding models is performed on the Euclidean space. This introduces a gap in the training and the usage of these vector representation models.

In this research work, we propose to handle the above challenges to introduce an improved graph based unsupervised global event detection model for Twitter data. We represent the Twitter data as a GoS leveraging the semantic context generated through dense word vector representations using the Joint Spherical Embedding(JoSE) model [24]. JoSE efficiently captures both the contextual and the direction information of the tweets in word embeddings.

The pre-trained JoSE model is fine-tuned on the underlying data to generate contextual word vector representation of tweets. The semantic similarity and the time difference among these tweets are used to create edges in the graph. Furthermore, with the incoming tweets, the size of the graph may increase drastically. Thus, this research work proposes to prune the graph based on the component/subgraph sizes. This helps in reducing the size of the generated graph.

The contextual relationship in the GoS may also not produce highly accurate results. So, the edges may be considered to represent the probability of similarity among the data points. This probability induces uncertainty in the relationship among the data points. The partition-based and shortest-path based clustering such as k-means and spectral clustering techniques does not generate the expected outcomes on such probability-based edge weights because these are not capable of capturing the possible world semantics [25]. Thus, to capture the most substantial relationships, the proposed model employs uncertain graph-based clustering MCP Clustering. This clustering is based on partial-k clustering [26], which tries to cover the maximal subset of nodes that are connected with the most weighted edges. This model gives high weightage to the connection density as well as the connection weights in the graph while clustering. The proposed model intuits to reduce the computational cost with the help of graph pruning and the approximation based clustering. Furthermore, the improvement in F1-score (i.e. accuracy) occurs due to the maintenance of contextual information in the tweet-pairs/relationship by using the graph-of-sentences.

A. RESEARCH OBJECTIVES AND CONTRIBUTIONS

The primary objective of this research work is to propose an approximate graph-based global event detection model that captures the uncertainty in the Twitter data. This model will intuitively improve the run-time performance while maintaining the qualitative performance (F1-score) of the event detection process. The details of the research objectives and the contributions of the work are given as follows:

- A spherical embedding based Graph of Tweets representation has been proposed to model the Twitter data. This model captures most of the contextual and the directional information from the tweets.

- The run-time and the F1-score performance of the different feature representation models (Word2Vec, JoSE and Doc2Vec) has been compared on different graph clustering tasks (Modularity Clustering [27], MCL Clustering [28] and MCP Clustering [25]).
- As the size of the graph may become huge due to pair-wise tweet relationship stored as edges reaching up to $O(n^2)$, a subgraph/component-based filtering approach is proposed to prune the graph. This helps in removing the insignificant subgraphs/components from the graph, depending on the size of the subgraphs. The performance of the filtered and unfiltered graph has been compared on a clustering task.
- An edge weight and connection-based graph clustering approach has been employed to detect the events more efficiently. Due to the partial-k nature of clustering, it has been termed as approximation based clustering in the current work. This clustering helps in improving the run-time performance of the model as compared to the state-of-the-art.
- An unsupervised event detection framework based on modeling Twitter data as the graph has been proposed. The proposed framework provides improved run-time and F1-score performance than other event detection models.

The organization of the rest of the paper is given as follows. Section 2 presents the literature review of the event detection models categorized based on techniques. Section 3 presents the methodology of the proposed work, which includes the dataset description, feature extraction, and the proposed approach. Section 4 provides the performance evaluation of the proposed model with state-of-the-art research works. Section 5 concludes the research work.

II. RELATED WORK

There are different survey papers in the literature focusing on summarizing different types of event detection models in terms of the orientation of content and methodologies. In 2013, Aiello *et al.* [29] categorized the event detection models into *document pivot* methods, where the primary features are tweets and *feature pivot* methods, where the primary features are the essential keywords, hashtags and other user-level features such as language-specific or user-specific information. In 2017, Hasan *et al.* [30] summarized the event detection models for Twitter in terms of different techniques used, i.e., term interestingness-based approaches, topic modeling-based approaches, incremental clustering-based approaches, and the other miscellaneous approaches. The authors also identified the primary challenge of event detection as high computational cost, and later in 2019 introduced a framework TwitterNews++ [31] to overcome this challenge. Furthermore, in 2019, Saeed *et al.* [32] categorized the different event detection models in terms of event detection types, i.e., Specified and Unspecified event detection, which use supervised, unsupervised, and semi-supervised approaches. They also categorized the feature extraction methods into keyword-based, Twitter-based, location-based, and language-based approaches. The authors also discussed the common framework of event detection and the challenges and limitations in the current literature.

In this section, some of the critical research works related to the event detection process related to the current proposed research work have been summarized. The categorization used is mostly similar to that provided in [30]. A brief literature review of the same is given further in the section.

A. SUPERVISED APPROACHES

The supervised approaches for event detection generally require ample labeled data to learn the models with high accuracy. These event detection models utilize machine learning and deep learning models for classification.

In 2017, Badjatiya *et al.* [6] proposed an event detection model for hate speech detection on Twitter. The authors compared three deep learning models, i.e., CNN, LSTM, and FastText using Random Embeddings and GloVe Embeddings. The performance was compared with baseline machine learning models, i.e., Logistic Regression, SVM, and GBDT using Char n-gram, TF-IDF, and BOW. Their analysis found that the combination of LSTM, Random Embedding and GBDT provides the best performance. In 2018, Nguyen and Grishman [33] used Graph Convolutional networks (GCN) for event detection while maintaining the document's syntactical information. First, a syntax parse tree was generated using BOI annotations for entity mentions. Then, a multi-class classification was performed using convolutional operation on the dependency graph for each token in the sentence. The vectors were then pooled to perform event detection. In 2019, Bekoulis *et al.* [34] proposed to exploit the sequential/chronological nature of the social media streams for a supervised event detection model. This work is an extension to the work described in [19]. The data stream was divided into bins consisting of tweets, which further consisted of words. They used Random Average pooled word embeddings to generate the word level representations of each bin. Further, Multilayer Perceptron (MLP) was used for binary classification for the presence of an event. They later phrased the sub-event detection as a sequence labeling problem by using LSTM to propagate the chronological information with time.

Most of the above-described models are based on machine learning and deep learning models. However, the deep learning models are highly complex and require expensive hardware and more run-time to achieve the best F1-score. Also, most of these approaches are supervised and require an ample amount of annotated data for processing. However, these requirements make these models undesirable and inapplicable for real-world data and limited resources.

B. PROBABILISTIC TOPIC MODEL BASED APPROACHES

Probabilistic Topic model-based approaches represent tweets or clusters as the probability distribution over various latent

topics. This helps in finding the hidden semantic structure of the tweets, which further helps in detecting the events.

In 2013, Wang *et al.* [17] p erformed bursty word extraction using the gaussian mixture model and applied a time-dependent Hierarchical Dirichlet process (HDP) to detect the new events. In 2014, Xie *et al.* [35], proposed a novel data sketch model to detect acceleration in three quantities: Twitter stream, words, and a pair of words. This approach triggered the topic inference when an acceleration on these stream quantities was detected. In 2015, Wei *et al.* [36] used a Bayesian graphical model approach to find local events while considering the time and space information. The proposed model had three major components, i.e. (1) Event Model containing information about the specific events (2) Document Model containing information about the document and (3) Language Model containing information about the topic of the document. The information here refers to the tweet textual, spatial, and temporal distribution. The model parameters were then inferred using the Gibbs EM algorithm. In 2017, Gou and Gong [37] proposed an incremental document clustering approach leveraging Dirichlet Process. The authors used the summation of Gaussian Kernels to estimate the density function of each incoming document. Further, the events are extracted by applying the Dirichlet process parameter estimations. Later, in 2017 [38], the authors used Bayesian probabilistic clustering to find clusters in the tweet vectors created by using the CBOW model of Word2Vec. This model gave an advantage of higher precision and accuracy over the state-of-the-art.

C. CLUSTERING BASED APPROACHES

Due to the huge amount of data generated through the continuous Twitter streams and the inapplicability of the partition-based methods such as K-means on such data, many researchers have used incremental threshold-based clustering of Twitter streams for event detection. In 2011, Becker *et al.* [39] used an incremental clustering algorithm to detect events from the Twitter stream. In this work, authors clustered the new incoming tweets based on similarity against the existing clusters. Once the clusters are created, SVM based classifier was used to extract the non-trivial events from the set. The confidence score assigned by the classifier was later used for ranking the non-trivial events.

Some event detection models exploit the frequent pattern mining to detect events. In 2013, Aiello *et al.* [29] introduced a Soft frequent pattern mining approach while considering the n-gram similarity in tweets. Further, they used a group average hierarchical clustering algorithm to iteratively merge the closest pairs of clusters, starting with each term as individual clusters. This work was later improved by Choi and Park [40] in 2019, which considered the utility of each tweet and term for mining patterns. This work represented a set of tweets in a window as transactions, where the words of tweets represent the items of transactions. The words occurring with high frequency in the transaction were supposed to lead to the important topics in the window. The utility of words was decided based on the growth of their frequency in the subsequent windows.

Some other research work also used different types of clustering to detect events such as hierarchical clustering [29] and SCAN Clustering [41]. Later, some hybrid approaches were introduced in the literature which used incremental clustering along with other clustering models. For example, in 2017, [38] authors used GMM for the primary clustering and then used a sliding window-based incremental clustering approach to detect events. In 2019 [31], authors used the specialized inverted indices based TF-IDF and incremental clustering to reduce the computational cost of event detection. However, this model only used the textual information and not the localization and temporal information available from the tweets. Similarly, in 2019 Ghaemi and Farnaghi [42] proposed, VCDT (Varied Density-based spatial Clustering for Twitter data), an improvement on DBSCAN by incorporating spatial heterogeneity by identifying clusters of varying spatial densities. They considered the textual as well as spatial features of the Twitter data to detect the spatially clustered events. VCDT determined different search radius for clusters by using exponential features.

D. GRAPH BASED APPROACHES

The matrix representation of the data makes the use of graph clustering approaches very efficient for the underlying problems. In 2011, Weng and Lee [18] proposed EDCoW (Event Detection with Clustering of wavelet-based Signals) to detect events from the Twitter data. EDCoW first measures the burst in words (uni-grams) using the Wavelet Transformation. The similarity between each pair of words having high bursty energy is calculated using cross-correlation. The words are then clustered using Graph Modularity based partitioning [27] for event detection. This work suffered from high computational cost due to wavelet transformation and failing to capture the contextual information due to the usage of unigrams.

In 2014, Zhang *et al.* [43] proposed an event detection model by creating a relations graph from high-frequency words (burst words) and then clustering that graph by finding the strongly connected components. Later, they also predicted the popularity of the detected events in the near future using the related posts and the user relations in the social network. In 2015, Doulamis *et al.* [44] proposed capturing the temporal dynamicity of the contents of the tweets by generating feature trajectory of words distribution for a time-frame. They redefined the TF-IDF score as time-varying metric and also generated a fuzzy tweet based representation of time signals. A graph-of-words (GoW) is generated using the wavelet time-series signals of words. The edge weights are calculated using the correlation coefficient and Riemannian Metric because of their efficiency in capturing temporal dynamic nature. As the words can belong to many different events due to polysemy, they apply multi-assignment graph partitioning on the graph to detect events. In 2015, Meladianos *et al.* [45] used a graph of words of the Twitter data and degeneracy

in the graph for identifying the subevents in the form of important moments. The graph degeneracy is based on the k-core of the graph, i.e., identifying the maximal connected subgraph with at least k degree vertices. The sub-events were then detected using the tweeting weights, which was related to the frequency of the used terms. In 2016, Nguyena and Jung [46] detected the events to analyze online users' social behavior. They extracted the important terms and generated the keyword signals as a discrete-time sequence of its meaningful degree. The data was then converted into a graph with meta information of tweets as nodes and the cross-correlation of time-series as the edges. The graph was then clustered using density-based spatial clustering. In 2016, Manaskasemsak *et al.* [20] proposed to build graph-of-tweets using the TF-IDF term vectors. They applied Markov Clustering [28] on the graph to identify the events in the form of clusters. However, Word2Vec is considered to be more context-sensitive than TF-IDF in the literature because Word2Vec has better efficiency in finding semantic analogies. In 2018, Meladianos *et al.* [19] proposed a graph-based event detection approach by using convex optimization. They represented the set of tweets posted in a given time interval as a graph of words where nodes were the frequent words, and the edges represented the co-occurrence in tweets in the given interval. They identified an event trigger by analyzing the change in edge weights for the change corresponding to the previous time interval.

Most of the graph-based event detection models in the literature focus on generating the graph of words; however, using uni-grams may not be considered sufficient to generate highly accurate results. Thus there is a need to use the multi-grams or segments of tweets to retain maximum contextual information [29]. In this research work, we propose to use vector representation of sentences in place of tokens to represent the nodes and the uncertain relationships among the tweets to represent the edges in the graph. In place of deep learning models, the proposed approach provides a more straightforward graph-based approach to deal with the challenges while requiring a lower computational cost.

III. ELEMENTS OF PROPOSED MODEL

In this section, a detailed description of the steps involved in the proposed approach has been provided. A block diagram representation is given in Figure 1. Broadly, the tweets are first pre-processed by performing tweet parsing and cleaning. Next, the feature representation of the tweets is generated using JoSE, and the Twitter graph is generated where the relationship among the tweets is considered to be uncertain and represented in the form of similarity scores. The graph is further pruned using the proposed component filtration to remove irrelevant tweet-pairs to reduce the size of the Twitter graphs. However, the semantic similarities used to represent the edges are considered uncertain. Thus, we further apply uncertain graph clustering *i.e.* MCP Clustering [25] to extract/capture the most certain clusters. This uncertain graph clustering is also approximate, which helps to improve the

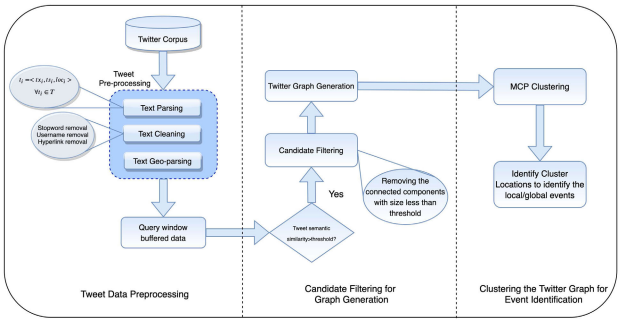


FIGURE 1. The block diagram representing the steps involved in the proposed approach.

run-time performance of the proposed model. The clusters generated will consist of frequently correlated tweets, which represent the events. Algorithm 1 provides the pseudo-code for the underlying event detection model. The details on the steps described are given further in the section.

A. DATASET PRE-PROCESSING

The Twitter data in raw format may contain many impurities that can introduce much interference during the processing. So, to improve the quality of the data for further data processing, the dataset is first pre-processed. Below are some methods used to pre-process the data.

- *Tweet Parsing*: In this step, the most relevant information such as *Text* and *Creation-time* are extracted from the tweets. The tweets thus can be represented as a tuple of these features *i.e.* $t_i =< tx_i, ts_i > \forall t_i \in T$, where T is the Twitter corpus, t_i is i^{th} tweet, tx is the tweet text, ts is the time-stamp.
- *Tweet Cleaning*: In this step, all the stop words, recurring characters, hyperlinks, and other non-decodable information is deleted from the texts of the tweets. Next, some important keywords are generated from the tweet text using the Part-of-Speech (POS) tagging and Named Entity Recognition (NER) detection. The cleaned tweets are further processed to generate the word embeddings.

The Twitter data consists of two broad feature sets, i.e., tweet-level features, which include retweets, replies, etc. and user-level features include the number of followers, number of friends, etc. However, in many previous research works [38], semantic, temporal, and spatial features of the tweets have been considered enough for event detection. Also, the tweet level features such as retweets and replies are all contained in the processed data required during the Twitter graph generation. Thus, the proposed work only focuses on the tweet-level features, i.e., tweet text and time interval, and have not considered the user-level features. The spatial feature has not been considered in this work because this model aims to introduce a global event detection model that is not specific to any region. The location feature can be added to the process to review the events generated. However, this is out of the scope of the current research work and can be taken up for future work.

Algorithm 1 Proposed Model(Twitter Corpus T)

Result: Events $E = \{E_1, E_2, \ldots, E_k\}$ with most certain connection probabilities.

$G_T \leftarrow \phi$

$t_i =< vec_i, ts_i >$ ▷ vec_i be the tweet vector and ts_i be the time-stamp for tweet t_i

for (t_i, t_j) *in* T **do**

  $simi_{temporal}(t_i,t_j) = \frac{ts[t_i]-ts[t_j]}{3600}$

  if $simi_{temporal}(t_i,t_j) < t_{ts}$ **then**

    $simi_{semantic}\left(t_i, t_j\right) = 1 - Cos\left(vec\left[t_i\right], vec\left[t_j\right]\right)$

    if $simi_{semantic}(t_i, t_j) > t_{sem}$ **then**

      $G_T.V \leftarrow G_T.V \bigcup \left\{t_i, t_j\right\}$

      $G_T.E(t_i, t_j) \leftarrow simi_{semantic}(t_i, t_j)$

    end

  end

end

$CC_{G_T} \leftarrow connected\_components(G_T)$ ▷ returns the list of connected components in G_T

$C_{big} \leftarrow CC_{G_T}.sort\_descending[0]$ ▷ saving the biggest component in C_{big}

$C_{freq} \leftarrow most\_frequent\_component(CC_{G_T})$ ▷ saving the most frequent component in C_{freq}

$k = 0.05 - 0.01 * log(\frac{|C_{big}|}{f(C_{freq})})$

$t_{size} = \left|C_{big}\right| * k$

$FG_T \leftarrow FG_T \bigcup \left\{cc : |cc| > t_{size} \text{ for } cc \text{ in } CC_{G_T}\right\}$

$E \leftarrow MCP\_Clustering(FG_T)$

return E

B. FEATURE EXTRACTION

The quality of detected events, while using the Twitter data, is highly dependent on the co-occurred textual information gained through the data. The feature set vectors of the textual data are generally huge. The performance of the event detection process can be enhanced very much by using the essential contextual information in the form of feature sets [38]. Earlier event detection models have used TF-IDF [20] or some variant of TF-IDF [44] to generate the word vector representations. However, the TF-IDF vector representation fails to capture all the features, i.e., the position, semantics, and the co-occurrences of the words in a sentence. Also, as discussed in [29], different closely interconnected topics may have many commonly frequent terms and the pairwise similarity between words tend to generate generic or merged topics. Furthermore, using multi-grams provides more domain-specific information and reduces the noise in the event detection process [47]. The deep neural model-based word embedding generation models such as Word2Vec [21] and Doc2Vec [23] have shown efficient performance in overcoming the limitation of the earlier models such as TF-IDF.

The text mining tasks such as word similarity measurements, word analogy extraction, and text/document clustering works better with normalized embeddings and with spherical clustering models. However, word embedding generation models such as Word2Vec, GloVe, and Doc2Vec perform such tasks by first learning the embeddings on the Euclidean space and then normalizing it for the spherical space. This generates a gap between the training and usage of the textual embeddings. For example, in the current literature, Word2Vec can be considered highly efficient in performing tweet pooling; however, due to the lack of directional information while training, Word2Vec may not capture the full contextual information sometimes. For example, the cosine similarity of the word vectors for very unrelated sentences "NHPINDIA: Food choices exhibit a major role in your oral health" and "RGMVP: A learning session in a village federation of #SHGs RGMVP #Amethi #WomenInHealth #MaternalHealth" is found to be 0.691. This high similarity score introduces the uncertainty in capturing the relationship among tweets. Joint Spherical Embeddings (JoSE) [24] bridges this gap by learning the unsupervised paragraph embeddings jointly by modeling a generative relationship. It uses the Riemannian distance optimization procedure with guaranteed convergence to learn the generative model. JoSE exploits the word-word (local) and word-paragraph (global) co-occurrence statistics in a spherical generative model.

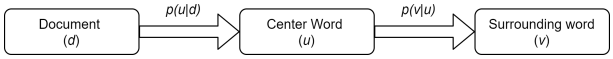


FIGURE 2. The block diagram representing the steps involved in the JoSE.

JoSE is based on the assumption that the directions of the vector can capture textual semantics more effectively and high directional similarity implies a higher co-occurrence probability. JoSE based text embedding generation is a two-step process depicted in the block diagram given in Figure 2. The spherical generative model learns the center word (u) semantics using the global context (document d) expressed as Equation 1 and the local context (i.e., surrounding words v) using the center word semantics (u) expressed as Equation 2.

$$p(u|d) \propto e^{cos(u,d)} \tag{1}$$

$$p(v|u) \propto e^{cos(v,u)} \tag{2}$$

where, $\|u\| = \|v\| = \|d\| = 1$ and $cos(\cdot, \cdot)$ denotes the cosine of the angle between two vectors in a unit sphere.

$$\begin{aligned} p(v, u|d) &\propto p(v|u) \cdot p(u|d) \\ &\propto vMF_p(v; u, 1) \cdot vMF_p(v; d, 1) \end{aligned} \tag{3}$$

Equation 1 and 2 are then derived for the spherical space by using the von Mises-Fisher (vMF) distribution as given in Equation 3. The text embedding training is done as a constrained optimization problem for spherical space using the loss function given in Equation 4.

$$\min_{\Theta} L(\Theta) \quad \text{s.t.} \quad \forall \theta \in \Theta : \|\theta\| = 1,$$

where, $L(\Theta)$ is the objective loss function,

$$\begin{aligned} \Theta &= \{u_i\}|_{i=1}^{|V|} \cup \{v_i\}|_{i=1}^{|V|} \cup \{d_i\}|_{i=1}^{|D|}, \\ \text{and } L(u, v, d) &= max(0, m - cos(u, v) - cos(u, d) \\ &\quad + cos(v, u') + cos(u', d)) \end{aligned} \tag{4}$$

where, u and u' denotes the set of positive and the negative samples respectively.

The Euclidean space-based Stochastic Gradient Descent (SGD) cannot efficiently optimize the proposed objective. Thus, an optimization approach is used that updates the parameter as per a surface with constant positive curvature, i.e., Riemannian Optimization. A sphere can be defined as a Riemannian manifold with constant positive curvature. RGD is expressed as Equation 5.

$$\begin{aligned} x_{t+1} &= exp_{x_t}(-\eta_t grad(f(x_t))), \\ grad(f(x)) &= (I - xx^T)\nabla f(x) \end{aligned} \tag{5}$$

The proposed framework employs Joint Spherical Embeddings in the underlying data to extract features from the text. These embeddings ensure that we get better word similarity and text clustering results in the final steps. The results shown in [24] shows the better accuracy of text clustering and word similarity tasks on the document-based dataset. However, the current work is based on detecting events from Twitter data, which includes the short-text sentences, where capturing semantics may become more stringent than the document-based datasets. Thus, the performance of JoSE is compared on two labeled Twitter datasets with other state-of-the-art word embeddings models, i.e., Word2Vec and Doc2Vec. The performance is compared in terms of run-time and the inter-cluster and intra-cluster similarity scores for clustering in Section IV.B.

C. TWITTER GRAPH GENERATION

The most prominent property of the graph-based data models is to capture the relationships in the data. That is, the graphs tend to give priority to the connections over the data points. The strong relationships in the graph represent the strong correlations/associations among data points/nodes. In this research work, the Twitter data is modeled as an uncertain graph where nodes represent the tweets and edge weights represent the magnitude of semantic and temporal similarity between the tweet pairs.

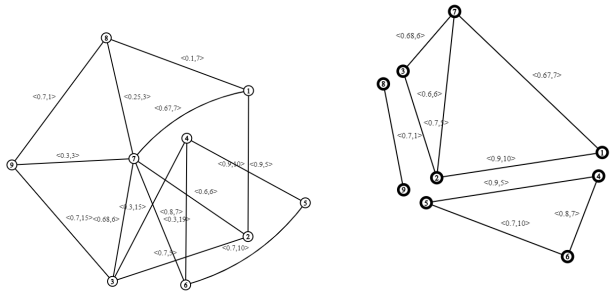
Definition 1: Given a Tweet corpus T such that $t_i =< vec_i, ts_i >$ be any tweet in T. The Twitter Graph $G_T(V, E)$, where $V \subseteq T$ and $E \subseteq \{\{t_i, t_j\} | (t_i, t_j) \in V^2 \wedge t_i \neq t_j\}$. Let $w(t_i, t_j)$ represent the edge weight where $\{t_i, t_j\}$ be an edge in E. Then the edge weights between the tweet-pairs represent the magnitude of pair-wise semantic similarity and time difference between them, given as $w(t_i, t_j) = \left\{< simi_{semantic}(t_i, t_j), simi_{temporal}(t_i, t_j) > \forall i, j\right\}$.

$$\begin{aligned} simi_{semantic}\left(t_i, t_j\right) &= 1 - Cos\left(vec\left[t_i\right], vec\left[t_j\right]\right) \\ simi_{temporal}(t_i, t_j) &= \frac{ts[t_i] - ts[t_j]}{3600} \end{aligned} \tag{6}$$

where, $Cos\left(vec\left[t_i\right], vec\left[t_j\right]\right)$ is the cosine distance between the tweet vectors. Here, we use Cosine distance to calculate the similarity between the tweet vectors. Cosine distance is a simple and efficient metric to calculate the distance between the vectors when the magnitudes of the vectors do not matter. This distance will only represent how the sentences are related to each other [48].

Depending on the scale, the event may show burst in different time-frames. For generating the Twitter graph, the proposed method does not use the query window framework, which needs to assign the *window shift width* along with the *window size* to generate a Twitter edge combination. In the given process, a window size of t_{ts} time-units is used, and the tweet-pairs having time-difference less than t_{ts} time-units are connected by an edge. Furthermore, with every new incoming tweet in the data, an edge is created in the existing graph if the time difference is found to be less than t_{ts} time-units for any two nodes. Later, the tweet pairs having a similarity score, calculated using equation 6, less than the threshold, i.e., t_{sem} are removed from the Twitter graph G_T.

The value of the similarity threshold (t_{sem}), is calculated by fitting the logistic regression model on the sample data. First, the raw Twitter data is binned into five equal-width bins based on $simi_{semantic}$ values. These bins are used as strata in stratified sampling by taking an equal proportion of samples. A binary relevance score is assigned manually to each sample for the quality of $simi_{semantic}$ of the tweets. Later, a logistic regression model is fitted on the sample where the relevance score is the independent variable and the similarity score is the dependent variable. The similarity score value, where the prediction probability of logistic regression switches from less than 0.5 to greater than 0.5, is chosen as the similarity threshold. This results in a t_{sem} to be 0.6. Further, the tweet pairs having $simi_{semantic}$ greater than t_{sem} are used to generate the Twitter graph.



(a) (b)

FIGURE 3. An example of application of Twitter graph generation step. (a) Graph Generated from the raw data (b) Graph generated after applying graph generation step.

Let $T = \{t_1, t_2, \ldots, t_n\}$ be a Twitter corpus with n tweets, where $t_i =< vec[t_i], ts[t_i] >$, $vec[t_i]$ is the vector representation of tweet text and $ts[t_i]$ is the timestamp of creation of tweet. The graph-of-sentence G_T generated from this data will be a set of nodes i.e. $G.V = \{1, 2, \ldots, n\}$ representing the tweet ids, and edges i.e. $G.E = \left\{< simi_{semantic}(t_i, t_j), simi_{temporal}(t_i, t_j) > \forall i, j\right\}$ representing the pair-wise cosine similarity and the time difference between tweets calculated by using Equation 6. Figure 3a

shows an example of the raw graph generated on the sample of the tweets. As explained above the irrelevant edges can be removed from this graph by using threshold on semantic similarity and time difference. The edges having cosine similarity less than t_{sem} and time-difference more than t_{ts} are deleted from the graph. In the Figure 3 the exemplary thresholds are taken to be $t_{sem} = 0.6$ and $t_{ts} = 12hrs$. After the deletion the transformed graph G_T is shown in Figure 3b. In Figure 3, both the semantic similarity and temporal similarity among the tweet-pairs have been provided as the edge weights. However, in Algorithm 1, only semantic similarity is retained in the edges. It is because the temporal difference is used initially to filter the graph, as shown in the initial portion of For loop in Algorithm 1. The final input for the graph filtration step only keeps the semantic similarity scores as the edge weight, and the temporal similarity scores are removed after the initial processing.

D. COMPONENT BASED GRAPH FILTRATION

The resultant graph consists of multiple numbers of disjoint components/subgraphs, as shown in Figure 3b. Thus the Twitter graph represents the transactional graph setting [49], where different components may represent different sub-events. The *size* of the connected component is defined as the number of vertices in the graph. The analysis of the component size and the respective frequencies shows that the frequency of the smaller components is very high, and this frequency becomes minimal as the size of the component increases, as depicted in Figure 4.

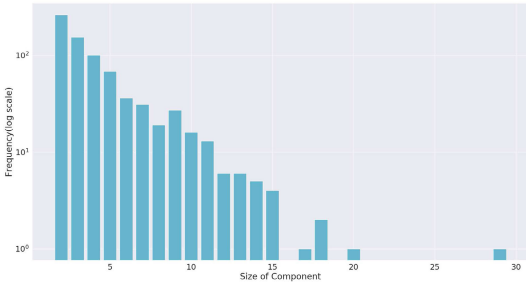
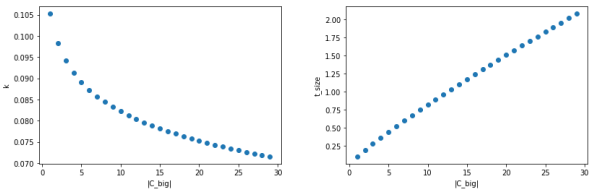


FIGURE 4. Number of components for a given component size in log scale.

These small-sized high-frequency subgraphs do not impart any new knowledge into the event detection process, and the removal of such subgraphs makes the Twitter graph more event-oriented. Thus, a component filtering approach is proposed to remove such irrelevant components. The components having size less than a threshold are removed from the graph G_T, and the resultant graph is named FG_T as given in Algorithm 1. The value of threshold is taken to be k^{th} of the size of the *biggest* component in the graph. The value of k is evaluated by using the Equation 7. The equation is chosen such that the variance and distribution of the data remain captured. A large value of the size of the biggest component represents that most of the tweets are linked to each other with high certainty. This process ensures that the different components/subgraphs in G_T have maximum intra-component and minimum inter-component topic similarity. The subgraphs/components having size i.e., number of nodes less than t_{size} are removed from the graph.

$$k = 0.05 - 0.01 * log(\frac{|C_{big}|}{f\left(C_{freq}\right)})$$
$$t_{size} = |C_{big}| * k \tag{7}$$

where, $|C_{big}|$ denotes the size of the biggest component (C_{big}) and $f\left(C_{freq}\right)$ denotes the frequency of the most frequent component C_{freq} in the graph G_T. $f\left(C_{freq}\right)$ is taken to normalize the size of the component and the frequency. It is divided by frequency because the smallest components are the most frequent and the largest components are the least frequent ones. As the value of the threshold (t_{size}) is dependent on the size of the biggest component, the reduction in the dataset decreases with the increase in the size of the biggest component. Figure 5 depicts the visualization of this change in the value of threshold k (Figure 5a) and t_{size} (Figure 5b) for different values of C_{big} keeping $f\left(C_{freq}\right) = 250$ in correspondence to the example shown in Figure 4, where 250 is the raw count and Figure 4 shows frequency on the log scale.



(a) (b)

FIGURE 5. The value of threshold k and t_{size} for the given component size $|C_{big}|$ and frequency of the most frequent component $f\left(C_{freq}\right)$.

E. GRAPH CLUSTERING FOR EVENT DETECTION

The Twitter graph FG_T generated as the result of candidate filtering may consist of variable-sized components. The importance of any component is directly proportional to the size of the component. Thus, a general idea to detect any event from such a graph would be to extract the top-k components. However, the semantic similarities calculated using the cosine similarity function can be misleading, as described in Section III.C. Thus, these semantic similarities can be treated as the probability of similarity among the tweets. This probability generates uncertainty in the graphical structure. The application of shortest-path based clustering (using Euclidean distance) and partition-based clustering (such as k-means) by considering edge probabilities as edge weights do not yield significant outcomes because it disregards the possible world semantics. Also, the partition-based clustering models such as k-means and spectral clustering and the probabilistic topic modeling based clustering such as LDA do not address the interconnection similarities i.e. the edge weights among the tweets. Thus, we propose to employ *MCP Clustering* [25] model, an uncertain graph clustering model to extract the

graph-clusters as events that maximize the certainty in the relationship among the tweets inside a cluster. The resultant clusters will be a set of nodes i.e., tweets having the strongest relationships among each other. These tweets are intuited to belong to the events oriented towards common topics.

Definition 2 (MCP Problem): "Given an uncertain graph G with n nodes and an integer k, with $1 \leq k < n$, the *Minimum Connection Probability (MCP)* problem requires to determine a k-clustering C (cluster centers c) of G with maximum *min-prob(C)*."

The *MCP Clustering* [25] model, is used for creating the clusters from the graph. This model is based on the fact that when the connection probabilities *i.e.* p_s are minimal, the exact computations become computationally expensive. So, a clustering model should be introduced, which is robust to estimations and tries to avoid the computations of minimal probability values. The MCP problem is defined as given in Definition 2.

where,

$$min - prob(C) = \min_{1 \leq i \leq k} (\min_{v \in C_i} (p_s(c_i, v))) \tag{8}$$

The MCP algorithm is modified to optimize the running time [25]. The probability threshold q of *MCP-Clustering* is set to $q_i = max\{1 - \gamma \cdot 2^i, p_L\}$ in iteration i, with $p_L = 10^{-4}$. When q_i becomes equal to p_L or all the nodes are covered in clustering, a binary search between q_i and q_{i-1} is performed to find the final probability guess. In each iteration the value of q is updated and the search is stopped when the ratio between the lower and upper bound becomes greater than $1 - \gamma$.

The core of the MCP clustering lies in Min-Partial clustering which is an approximation based clustering. *Min-Partial* [26] is the primitive model for a given threshold q on the connection probability which computes the partial k-clustering (Definition 3) of the uncertain graph, where the clusters cover a maximal subset of nodes, each connected to its cluster center with probability at least q *i.e. min-prob(C)* $\geq q$, where *min-prob(C)* is given in Equation 8, while all other nodes, deemed outliers, remain uncovered.

Definition 3 (Partial k Clustering): "Given a graph $G(V, E)$ and number of clusters k, a partial k clustering $C = (C_1, \ldots, C_k; c_i, \ldots, c_k)$ is a partition of a subset of V into k clusters $C_1 \ldots C_k$ (i.e. $\bigcup_{i=1,k} C_i \subseteq V$), where cluster c_i has the cluster centers $c_i \in C_i$ for $1 \leq i \leq k$."

Min-Partial randomly selects a random vertex v from the given vertex set V and saves it as the cluster center c_i. A set M_v is populated with the vertices $u \in V$ that have the $p_s(u, v)$ greater than the specified threshold q. All the $u \in V$ such that $p_s(u, c_i) \geq q$ are removed from the search set. This process is repeated for a user-given number of clusters k. At the end of k iterations, if the number of cluster centers in S is less than k, remaining vertices are added randomly. The resultant ensures that the best set of covered nodes are assigned to the k selected centers.

Definition 4: Let $p_{opt-min}(k)$ be the maximum value of $min - prob(C)$ over all full k-clusterings C of G. "For any $q \leq p^2_{opt-min}(k)$ and $\bar{q} \in [q, 1]$, the k-clustering C returned by *Min-Partial*$(G, k, q, \bar{q})$ covers all nodes." The proof of the same can be found in [25].

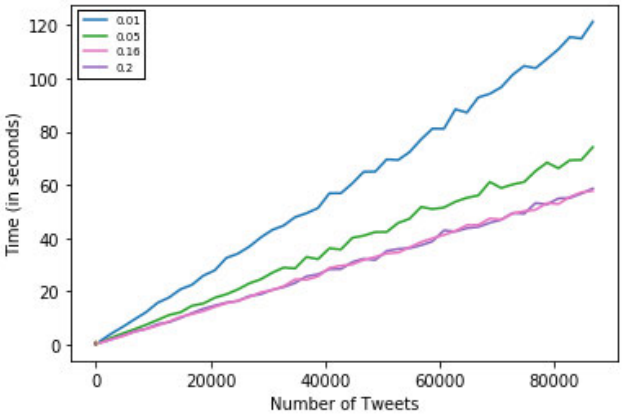


FIGURE 6. Variation in run-time performance of the MCP for different values of γ.

According to Definition 4, for a suitable guess for the minimum connection probability threshold q, *Min-Partial* returns clustering that covers all the nodes and provides a good solution to the *MCP* problem. To solve the *MCP* problem, the *Min-Partial* is run repeatedly, starting with $q = 1$, for progressively smaller guesses of q, decreasing q by $(1 + \gamma)$ for suitable $\gamma > 0$. This is done until the clustering that covers all the nodes is obtained. Figure 6 shows a change in run-time of MCP clustering for different values of $\gamma = \{0.01, 0.05, 0.16, 0.2\}$. As depicted from the Figure that it shows similar behavior on $\gamma = 0.16$ and $\gamma = 0.2$, so, the experimentation have been performed on $\gamma = 0.16$. The MCP Clustering algorithm is suitable for the approximate event detection model because the partial k clustering developed by the Min-Partial algorithm generates the clusters whose nodes are connected with the highest probability as given in Definition 4. Also, any event tends to trigger a high amount of tweets in a short duration, thus processing all these tweets can be considered redundant, and using such approximation models helps in improving the time complexity.

IV. EXPERIMENTATION AND RESULTS

This section provides experimental results of the proposed model on a set of benchmark and real-time Twitter datasets. First, the details of the benchmark and real-time dataset are provided. Then, the performance of the proposed model is compared with different state-of-the-art graph-based and probabilistic event detection models. All the experiments are performed on a computer system with an Intel® Xeon processor running at 2.20 GHz using 128 GB of RAM, running Ubuntu version 18.04.

A. DATASET DESCRIPTION

The performance of the proposed model is investigated on different types of Twitter datasets. Table 1 summarizes the

TABLE 1. Statistical description of the datasets used for performance analysis of the proposed model.

Sr. No.	Dataset	Number of Tweets	Number of Nodes	Number of Components	Size of Biggest Component
1	RepLab 2013 Twitter data	2481	2389	785	52
2	#Auspol Twitter Data	21065	12874	1601	200
3	Common Disease Dataset	1.085 million	195332	19434	74
4	Customer Support on Twitter Dataset	3.196 million	601271	172742	40

statistics of all the datasets taken. A brief description of the datasets is provided below:

- **Public Annotated Data 1: RepLab 2013 Twitter data:** RepLab2013 Twitter data belongs to the RepLab 2013 Competition [50]. However, the dataset used in the current form has been taken from [51]. The dataset has been filtered to contain only the relevant topics. That is, the dataset contains only the topics which have at least 100 tweets belonging to it. This dataset consisted of 2657 tweets IDs, out of which only 2481 could be retrieved using the Twitter Search API on 30 March 2020. The dataset is labeled with 15 different topics such as, *For sale*, *Suzuki cup*, *User comments* and *Money laundering / terrorism finance* etc.
- **Public Annotated Data 2: #Auspol Twitter Data:** This data consists of tweets containing keywords/hashtag #AusPol [51]. This hashtag was used to discuss the various topics regarding Australian Politics from 13 Jun 2017 to 2 Sept 2017 on Twitter. This dataset consists of 29,283 labeled tweets with 14 target labels such as #qdpol, #insiders and #trump etc. Out of the 29,283 tweet Ids, only 21065 tweets could be retrieved using the Twitter Search API on 30 Mar 2020.
- **Real-World Data: Common Diseases Dataset:** The Common Disease dataset contains tweets for two months period of 1 March to 30 April 2019, pertaining to the keywords related to different diseases. This data was collected using the Twitter Streaming API. The keywords used for the data collection are as follows: *Fever, Headache, Measles, Cough, Chicken Pox, Sneeze, Food Poisoning, Mumps, Heatstroke, Typhoid, Jaundice* and *Loss of appetite*. These diseases (*i.e.* keywords) were chosen because these are the commonly occurring diseases during the data collection period of 1 March to 30 April 2019. The data collection is global and resulted in a collection of around 1.09 million tweets.
- **Public Unannotated Data: Customer Support on Twitter Dataset:** The Customer Support on Twitter dataset is publicly available on Kaggle.[1] This dataset contains approximately 3 million tweets on customer support tweets and replies for the study of modern customer support practices and impact. It consists of tweets of complaints regarding different products and their corresponding solutions/help provided by the service providers. This dataset consists of a variety of product complaints. The detection of event in such a dataset can help in identifying a recurring problem in similar/same products/brands.

The datasets above described cover different categories in terms of size and the topic. The datasets can be arranged in increasing order of size as Data 1, Data 2, Data 3 and Data 4 and increasing order of topic-orientation, i.e., how much a dataset is specific to a given topic as Data 4, Data 1, Data 2=Data 3. Overall the datasets can be categorized as Data 1 having low tweet count and low topic orientation, Data 2 having low tweet count and high topic orientation, Data 3 having high tweet count and high topic orientation and Data 4 having high tweet count and low topic orientation. These datasets will help in generalizing the applicability of the proposed model on different qualitative and quantitative varieties.

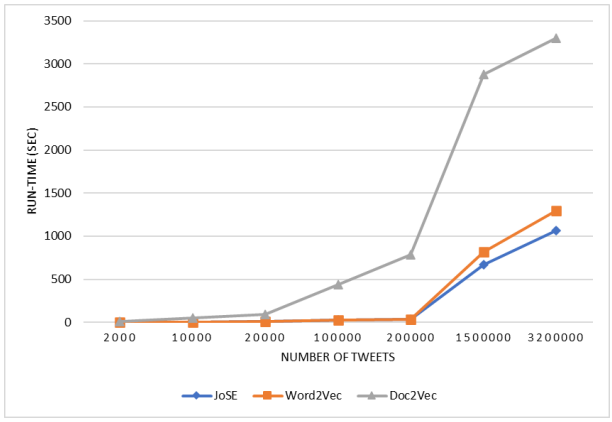


FIGURE 7. Variation in run-time performance of different unsupervised word embedding learning models.

B. FEATURE REPRESENTATION ANALYSIS

In this section, the run-time and qualitative performance (F1-score) of various feature extraction techniques have been compared. The comparison will help in identifying the feature extraction model that requires minimum run-time to provide the best qualitative performance. Thus, we train three feature extraction models Word2Vec [21], Doc2Vec [23] and JoSE [24] on the underlying dataset for an equal number of epochs i.e. 100. The values of all other model-specific parameters have been set to the default according to the python module used i.e., Gensim. Figure 7 provides the line-plot representation of the time taken by all the feature representation methods to complete 100 epochs. The figure depicts that with the increase in the data size, the time taken by Doc2Vec increases drastically as compared to its counterparts. JoSE and Word2Vec generally require a similar amount of time; however, for high data size JoSE requires lesser time than Word2Vec and Doc2Vec.

Next, the performance of these feature representation models has been compared on the clustering task on the Public Annotated Data set 1 and 2. We choose to compare the model on clustering tasks because we want to propose an unsupervised model for event detection. The graph clustering

[1] https://www.kaggle.com/thoughtvector/customer-support-on-twitter

(a) JoSE (b) Word2Vec (c) Doc2Vec

FIGURE 8. The t-SNE plot representation of different feature extraction models on the RepLab 2013 Twitter dataset. The different colors depict the desired target cluster labels.

models used for this task are Modularity Clustering (Mod-C) [27], MCP Clustering (MCP-C) [25] and MCL Clustering (MCL-C) [28]. The labels provided in Data 1 and Data 2 will help in visualizing the performance of the proposed model. The cluster quality performance metrics used for the comparison are Mutual Information (MI), Normalized Mutual Information (NMI), Adjusted Rand Index (ARI) as given in Equation 9, 10 and 11 respectively, having standard definitions.

$$MI(X,Y) = \sum_{y\in Y}\sum_{x\in X} log\left(\frac{p(x,y)}{p(x)p(y)}\right) \tag{9}$$

$$NMI(X,Y) = \frac{MI(X,Y)}{\sqrt{H(X)H(Y)}}$$

$$\text{where, } H(X) = -\sum_{i=1}^{n} p(x_i)log(p(x_i)) \tag{10}$$

$$ARI(X,Y) = \frac{RI(X,Y) - E\{RI(X,Y)\}}{max\{RI(X,Y)\} - E\{RI(X,Y)\}}$$

$$\text{where, } RI(X,Y) = \frac{a+b}{\binom{n}{2}} \tag{11}$$

As the proposed framework models the Twitter data as a graph for further processing, the clustering models used for comparison are graph clustering models. Table 2 and 3 summarizes the mean score for each clustering approach on each feature extraction technique for the given performance metrics on RepLab2013 dataset and AusPol dataset respectively. The top scorers per metric have been shown in bold. Table 2 and 3 depicts that JoSE and Word2Vec have comparable performances in terms of all the metrics for MCL Clustering and MCP Clustering. However, JoSE with MCP clustering shows better performance on the AusPol data, which is approximately ten times bigger in size than RepLab2013 data. Thus, scores in Table 3 can be considered more generalized. This indicates that JoSE has a better performance than other feature representation models in terms of run-time and F1-score. Furthermore, Figure 8 provides a t-SNE plot [52] visualization of the vector representations fine-tuned on the RepLab2013 dataset. The data points are colored as per the target cluster labels given in the data. The figure depicts that JoSE creates more identifiable, significant, and label oriented clusters, where Doc2Vec fails to capture that semantic similarity.

TABLE 2. The performance evaluation of feature representation and graph clustering models on the RepLab 2013 Twitter data.

Feature Representation	Graph Clustering	MI	NMI	ARI
JoSE	MCL-C	0.9313	0.3581	**0.2387**
	MCP-C	0.8488	0.3265	0.2349
	Mod-C	0.5959	0.2815	0.0189
Word2Vec	MCL-C	**0.9857**	**0.4064**	0.2234
	MCP-C	0.7919	0.3995	0.2236
	Mod-C	0.5511	0.25	0
Doc2Vec	MCL-C	0.5089	0.201	0.0905
	MCP-C	0.47542	0.18131	0.1095
	Mod-C	0.328	0.09429	0.00545

TABLE 3. The performance evaluation of feature representation and graph clustering models on the AusPol Twitter data.

Feature Representation	Graph Clustering	MI	NMI	ARI
JoSE	MCL-C	0.9069	0.3723	0.24
	MCP-C	**1.0177**	**0.4178**	**0.2799**
	Mod-C	0.1647	0.092	0.0074
Word2Vec	MCL-C	0.7773	0.3296	0.1926
	MCP-C	0.7783	0.3235	0.192
	Mod-C	-0.58	-2.65	0
Doc2Vec	MCL-C	0.3946	0.1749	0.123
	MCP-C	0.5554	0.2283	0.1545
	Mod-C	0.0996	0.0791	0.0134

C. CANDIDATE FILTRATION ANALYSIS

In this section, the performance of the event detection model before and after the component filtration is compared.

This analysis will help in investigating the effects of the filtration process. The initial Twitter graph is full of subgraphs/components, where small-sized subgraphs are the most frequent, and big sized subgraphs are the least frequent. The proposed filtration is based on the intuition that the small-sized high-frequency subgraphs do not impart any new knowledge into the event detection process, and removal of such subgraphs reduces the number of components and vertices and thus makes the Twitter graph more event-oriented. Here, event orientation means that only relevant information will remain in the graph, which will help the clustering model to focus on the critical subgraphs representing events.

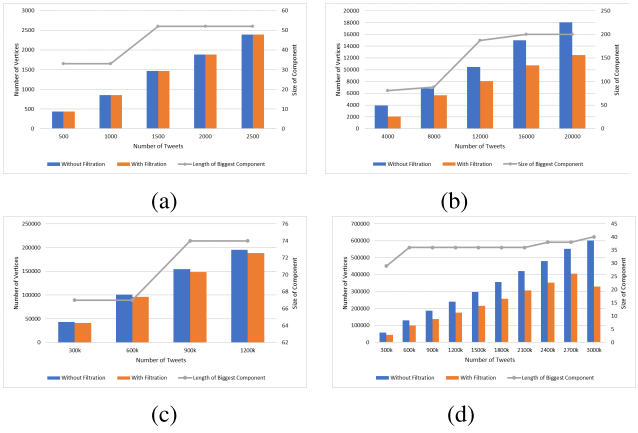


FIGURE 9. Number of vertices in the Twitter graph before and after the component filtration. (a) RepLab 2013 Twitter data (b) AusPol Twitter data (c) Common Diseases Dataset (d) Customer Support on Twitter Dataset.

Figure 9 shows the number of vertices in the Twitter graph before and after applying the filtration. The size of the *RepLab2013 data* is minimal as compared to the other datasets, which results in no change in the number of vertices in the Twitter graph after filtration. This means that no tweets were removed during the filtration. This is required to maintain a trade-off between the relevant and redundant information. The proposed model tries to provide a general filtration process that does not remove the relevant information while pruning the graph even if that requires keeping some redundant information. That is, no relevant information should be lost in efforts to remove superfluous information. However, the RepLab2013 data is so small that the removal of tweets from this data would not have much effect on the run-time performance. Whereas, *AusPol Twitter data* shows the largest amount of reduction in the number of vertices. This is due to the huge size of the biggest component in the Twitter graph, i.e., 200 (shown as gray line in the graphs shown in Figure 9) in comparison to the overall number of tweets in the dataset. This results in value of $k = 0.0595$ and $t_{size} = 12$; that is all the components with size less than 12 were removed from the graph. Similarly, for *Common Disease Dataset*, $k = 0.116$ and $t_{size} = 9$ and for *Customer Support on Twitter Dataset*, $k = 0.125$ and $t_{size} = 6$ with 3.6% and 45.22% reduction in the data respectively. Thus, it can be said the reduction in the data depends on the type of the dataset chosen, i.e., its orientation towards any given event, which is captured as the size and frequency of the components.

The graph filtration process depends on the size and the frequency of the underlying components in the graph. The skewness in the data may generate high threshold values as visible for AusPol Twitter data. However, this reduction of data may impact the performance of the event detection model. Thus, the intra-cluster similarity, F-measure, and run-time tradeoff of the proposed event detection model are analyzed on the filtered and unfiltered graph.

The qualitative performance measurement is done in terms of Precision, Recall and the F1-score. The Recall is calculated using the Equation 12. However, the proposed model needs to provide the number of clusters to be identified k beforehand. Furthermore, as the value of k can be controlled, the number of *Relevant events* and the *Retrieved events* become the same. Hence, to eliminate this ambiguity, *pseudo-Precision* is calculated as the average of the precision calculated for a range of number of clusters as given in Equation 13, starting from $k = 2$ up to the maximum number of *Relevant events* (i.e., n). The F1-Score is calculated using Precision and Recall values as given in Equation 14.

$$Recall = \frac{\left|\{Relevant\} \bigcap \{Retrieved\}\right|}{\left|\{Relevant\}\right|} \tag{12}$$

$$psudeo\text{-}Precision = \frac{\sum_{i=2}^{n} Precision(i)}{n} \tag{13}$$

where,

$$\begin{aligned} n &= \left|\{Relevant\}\right| \\ Precision(i) &= \frac{\left|\{Relevant\} \bigcap \{Retrieved\}\right|}{i} \\ F_1 &= 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \end{aligned} \tag{14}$$

TABLE 4. The values of *Precision, Recall* and *F1-score* obtained after applying the proposed model on Filtered and Unfiltered Twitter data.

Dataset	Filtered Graph			Unfiltered Graph		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score
Common Disease Dataset	0.714	0.651	0.681	0.754	0.701	0.727
Customer Support on Twitter Dataset	0.51	0.651	0.571	0.520	0.663	0.584
AusPol Twitter data	0.718	0.695	0.706	0.602	0.675	0.636

Table 4 provides a comparison in the F1-score of the proposed model on the filtered and unfiltered graph, i.e., *Common Disease, Customer Support on Twitter, and AusPol Twitter data*. The RepLab2013 data was not used for this process because there was no change in the number of vertices after filtration. There is an average of 2.86%, 2.21% and 0.29% decrease in the Precision, Recall and F1-Score of the proposed model, respectively. Similarly, Figure 10 provides the intra-cluster similarity (Equation 15) score comparison of the proposed model on the filtered and unfiltered graph. The average of absolute difference in the intra-cluster similarity for all the three datasets is 0.207, 0.0486 and 0.0434 as given in Figure 10a, 10b and 10c respectively. The change in the

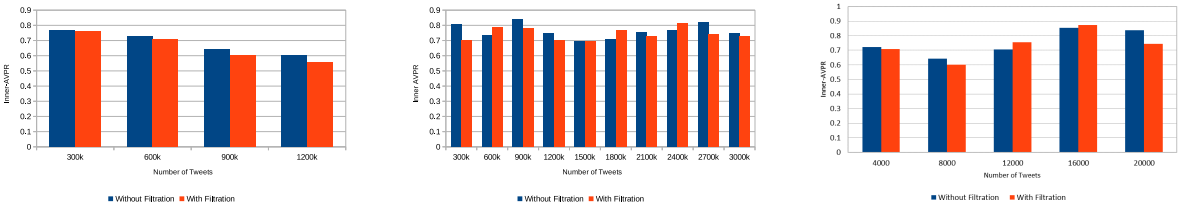


FIGURE 10. Intra-cluster similarity comparison of the proposed model on all the datasets with and without applying component filtration. (a)AusPol Twitter data (b) Common Diseases Dataset (c)Customer Support on Twitter Dataset.

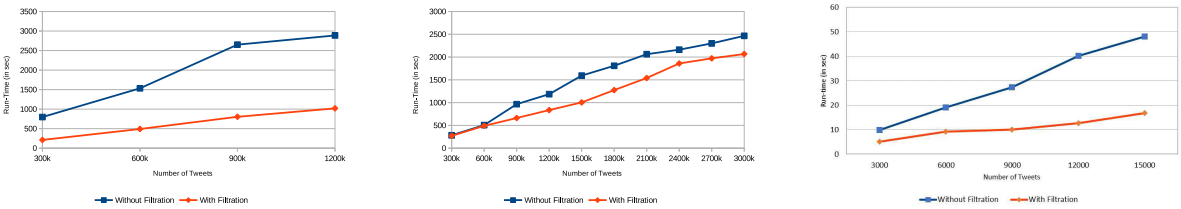


FIGURE 11. Run-time performance of the proposed model on all the datasets with and without applying component filtration. (a) AusPol Twitter data (b) Common Diseases Dataset (c) Customer Support on Twitter Dataset.

F1-score and intra-cluster similarity is very minimal, which indicates that the component filtration reduces the size of the Twitter graph without affecting the significant/important relationships.

Figure 11 provides the run-time performance comparison of the proposed model on filtered and unfiltered datasets. There is a reduction of approximately 69.01%, 20.51% and 59.39% on average in Figure 11a, 11b and 11c respectively. The Customer Support Twitter Dataset is the only dataset that is not oriented towards a topic. This gives rise to a large number of average size subgraphs/components in the Twitter graph and very small size of the biggest component, i.e., 40 in comparison to the data size, i.e., approx. 3,000,000 tweets. Thus, the data reduction after filtration is less, and as a result, the reduction in run-time is less.

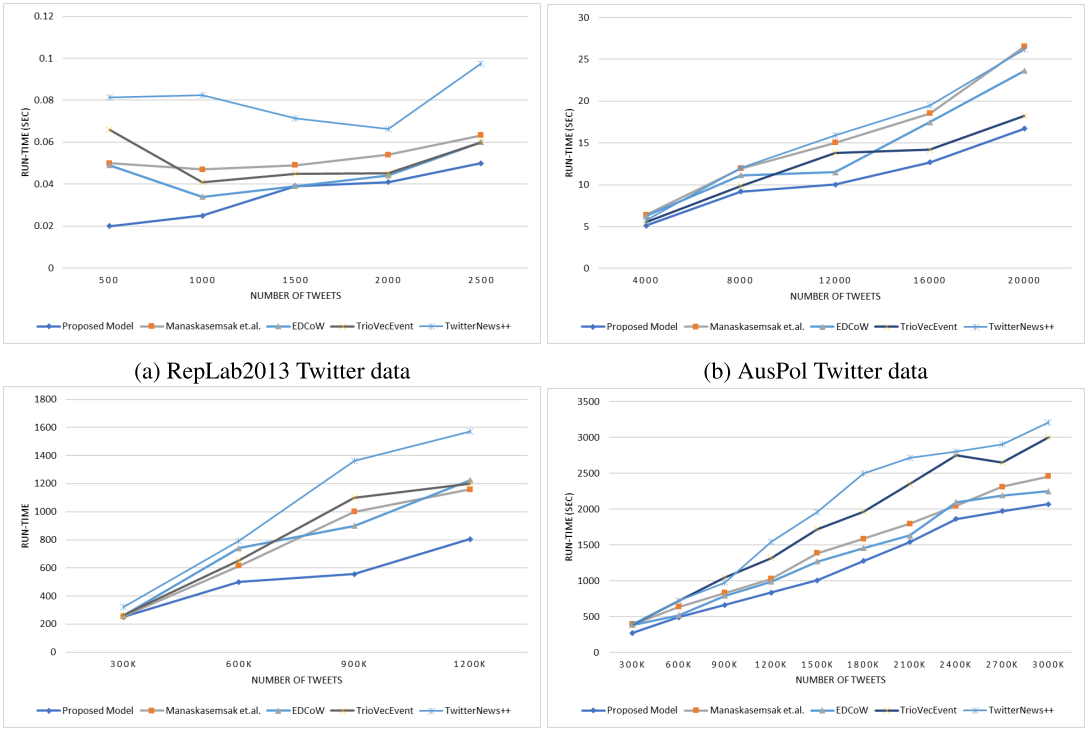
As seen from this analysis, for a maximum reduction of 2.86% in precision and a maximum change of 0.207 in the intra-cluster similarity, there is at least a 20.51% reduction in the run-time of the proposed model with the application of filtration. Thus, we consider this trade-off to be sufficient to use as part of the event detection model without loss of generality.

D. RUN-TIME PERFORMANCE ANALYSIS

In this section, the run-time performance of the proposed model with component filtration is compared to different event detection models for Twitter data. As the proposed model is unsupervised and based on clustering, the clustering-based event detection models are chosen as the opponents. Below is the brief detail of the opponent models.

- TrioVecEvent [38]: This event detection model is based on the probabilistic topic modeling approach. First, a multi-modal word-vector representation of the tweets is generated using Word2Vec CBOW [21], and then Bayesian Mixture Model Clustering is applied to detect the events from the data.
- EDCoW [18]: This event detection model is based on the graph clustering approach. The Twitter data is modeled as the Graph-of-words (GOW) first, and then modularity based clustering [27] is applied on the graph to detect clusters of essential keywords. The modularity clustering is based on the idea of the density of nodes in the network.
- Manaskasemsak et.al. [20]: This event detection model is also a graph clustering-based approach. The Twitter data is first modeled as the Graph-of-Sentences (GOS), where they use TF-IDF based word vector representation for tweets. The events are detected by using MCL Clustering [28] on the Twitter graph. The MCL Clustering is based on the principle of random walks in the graph.
- TwitterNews++ [31]: This event detection model uses the incremental clustering approach by using specialized inverted indices to store the term-tweet relationship and term-event relationships. It first searches for the non-unique incoming tweets and then finds candidate event clusters based on the TF-IDF generated tweet vector similarities using cosine distance.

Figure 12b, 12c and 12d shows that the run-time of the proposed model is least for all the datasets The proposed model does not show improved run-time performance in the case of RepLab2013 data because there was no removal of vertices after filtration. This indicates that the proposed model tends to show a better performance for bigger datasets.



(a) RepLab2013 Twitter data

(b) AusPol Twitter data

(c) Common Diseases Dataset

(d) Customer Support on Twitter Dataset

FIGURE 12. Run-time performance analysis of the proposed model in comparison to other event detection models.

There is an average 30% decrease in the run-time in the given datasets by using the proposed model. The reduction in the run-time is the result of the component filtration and approximate clustering. The graph filtration reduces the size of the overall dataset by removing the tweets that are not relevant to the study and the approximation based clustering uses partial-k clustering, which generates clusters covering a maximal subset of nodes with at least probability q. The combination of data filtration and approximation based clustering helps to reduce the overall run-time of the proposed event detection model.

E. INTRA-CLUSTER SIMILARITY

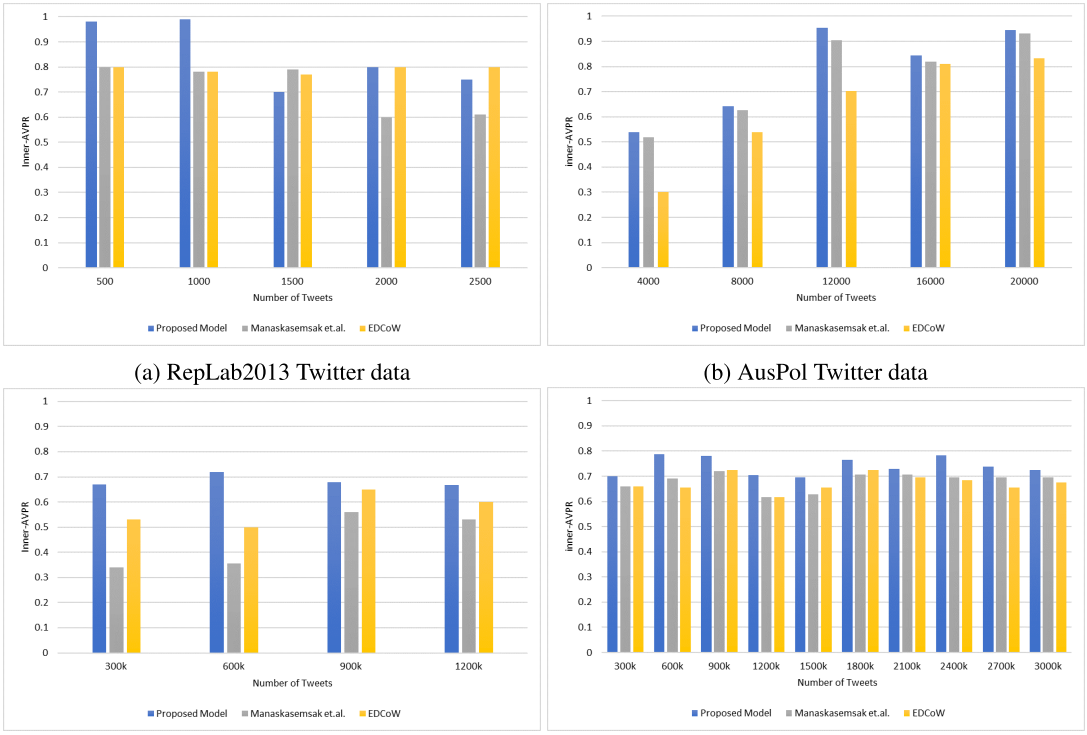
In this section, a performance metric is used to find out the intra-cluster similarity index [25]. The intra-cluster similarity index is called the *inner-Average Vertex Pairwise Reliability* ($AVPR_{inner}$) as given in Equation 15. The higher value of $AVPR_{inner}$ indicates that the cluster consists of nodes/tweets with higher similarity scores among them. The $AVPR_{inner}$ score of the proposed model is compared with the graph-based event detection models described in the previous section, i.e., Manaskasemsak *et al.* and EDCoW as given in Figure 13. The comparison is performed only with the graph-based models because only the graph-based models show this property. Figure 13 indicates that the proposed model shows the maximum score as compared to the other graph-based clustering algorithms.

$$AVPR_{inner} = \frac{\sum_{i=1}^{\tau} \sum_{u,v \in C_i} Pr\,(u, v)}{\sum_{i=1}^{\tau} \binom{|C_i|}{2}} \tag{15}$$

The improvement in the AVPR of the resultant events is because of the threshold-based edge creation (using temporal and semantic similarity threshold) and component-based graph filtration. The filtration is based on the size of the component, where components having size smaller than the given threshold are removed from the graph. This technique helps in retaining the higher number of tweets in the graph having strong/certain relationships among each other. The threshold based Twitter graph creation helps in keeping the strong relationships, and the filtration process helps in removing the redundant ones. Thus, the critical relationships among the graph are retained.

F. F1-SCORE MEASUREMENT

In this section, the qualitative performance of the proposed model is compared with the clustering based event detection models as given above. The qualitative performance is calculated in terms of *pseudo-Precision* (Equation 13), *Recall* (Equation 12) and *F1-score* (Equation 14). As the proposed model is unsupervised and Data 3 and 4 have no predefined labels for tweets, Latent Dirichlet Allocation (LDA) is applied to extract the topics and related keywords from the raw data as well as the clustered data to calculate the performance. This technique was applied on all the datasets to ensure fairness. The tweets belonging to the same events were pooled together manually, and the frequent keywords were extracted from these topic-pooled tweets. Similarly, the output of clustering was labeled by applying LDA on the clusters formed. The keywords extracted from the raw



(a) RepLab2013 Twitter data

(b) AusPol Twitter data

(c) Common Diseases Dataset

(d) Customer Support on Twitter Dataset

FIGURE 13. Intra-cluster similarity performance of the proposed model in comparison to other graph based event detection models.

TABLE 5. The values of *Precision, Recall* and *F1-score* obtained after applying the proposed model on different datasets.

Method	Precision	Recall	F1-Score
RepLab2013 Twitter Data			
Proposed Method	0.7098	0.833	0.766
Manaskasemsak et.al.	0.31	0.803	0.447
EDCoW	0.208	0.793	0.329
TwitterNews++	0.73	0.79	0.709
TrioVecEvent	0.69	0.76	0.723
AusPol Twitter Data			
Proposed Method	0.718	0.695	0.706
Manaskasemsak et.al.	0.25	0.786	0.379
EDCoW	0.361	0.806	0.498
TwitterNews++	0.754	0.636	0.689
TrioVecEvent	0.722	0.664	0.691
Common Disease Dataset			
Proposed Method	0.714	0.651	0.681
Manaskasemsak et.al.	0.25	0.604	0.353
EDCoW	0.194	0.612	0.294
TwitterNews++	0.742	0.572	0.646
TrioVecEvent	0.684	0.65	0.666
Customer Support on Twitter Dataset			
Proposed Method	0.51	0.651	0.571
Manaskasemsak et.al.	0.453	0.512	0.480
EDCoW	0.325	0.542	0.406
TwitterNews++	0.65	0.452	0.533
TrioVecEvent	0.553	0.595	0.573

Twitter corpus and the generated clusters were matched. If the keywords from the topics found in the raw corpus matched the keywords in a given cluster, the event was considered to be detected.

Table 5 gives the qualitative performance of all the discussed models for event detection in terms of *Precision, Recall* and *F1-score*. The TrioVecEvent uses Bayesian Mixture Model to cluster the tweet vectors (i.e., 250 dimensional) to perform event detection. However, the quality of the tweet vectors depends on the underlying language model being used. As described earlier, this generates the uncertainty in capturing the relationship among tweets while using the tweet vectors. Similarly, in the case of TwitterNews++, the lower F1-score could be due to the use of TF-IDF and cosine distance to identify the event clusters. The proposed model applies a more context-sensitive tweet representation model i.e., JoSE, for event detection. This leads to a better qualitative performance of the proposed model. Furthermore, the graphical representation of data, used in this work, models the Twitter data in the form of beneficial relationships. Thus, MCP Clustering helps in improving the F1-score of the event detection process by exploiting the relationships among the tweets.

The proposed event detection model helps in improving the F1-score because of the proposed filtration process. The general clustering model starts with randomly selecting the initial data points, in our case nodes (tweets). The proposed filtration process removes the subgraphs with a smaller size, which can be considered irrelevant because of their inability to represent any event due to its small size. Had these small-sized subgraphs belonged to the same topic, they would have remained connected by an edge. So, by removing these subgraphs, the initial selection becomes more focused on the relevant data points or tweets. That is, the clustering process will tend to select tweets that have higher chances of being a

part of an event. Thus, by reducing the probability of clustering process to get stuck in the outliers, the performance can be improved, and convergence of clustering is made faster.

The poor Precision score of the graph clustering-based event detection approaches i.e., EDCoW and Manaskasemsak *et al.*, is due to a large number of clusters returned. This is due to the unspecified number of clusters to be generated. It can be seen from the Table 5 that the proposed model has better Recall, Precision and F1-Score than all the other models. An anomalous behavior shown by all the models in case of *Customer Support on Twitter Dataset* indicates that the event detection process depends on the type of the dataset. This dataset is the least topic-oriented as compared to the other datasets, and the events generated turn out to be vague. Similarly, the size of RepLab2013 data is least and the F1-score of detected events is found to be maximum for this dataset.

V. CONCLUSION

In this paper, an unsupervised approximate graph clustering-based global event detection model is proposed. The proposed graph-based Twitter data representation helps in incorporating the uncertainty in the relationships among the tweets in the form of a graph. This Twitter graph helps in keeping track of the contextual as well as directional information while keeping the computational costs i.e., run-time requirements low. Furthermore, the component filtration of the Twitter graph removes the irrelevant tweet-pairs and edges from the graph. The removal of insignificant components from the graph helps in focusing the initial data point selection on the critical data points. These data points will have a higher probability of representing an event. Furthermore, the partial-k based uncertain graph clustering employed to cluster the graph improves the run-time performance of the proposed model. The experimental performance analysis affirmed the better performance of the proposed model versus its competitors on different types of datasets. The current work focuses on improving a graph-based event detection model that acts on data without the location consideration. The work can be further extended to make it region-centric and generate location-wise events. Also, for future work, different applications of the proposed event detection models such as health-care, emergency detection and disaster prediction, etc. can be explored.

REFERENCES

[1] W. Dou, X. Wang, W. Ribarsky, and M. Zhou, "Event detection in social media data," in *Proc. IEEE VisWeek Workshop Interact. Vis. Text Anal.-Task Driven Anal. Social Media Content*, Oct. 2012, pp. 971–980.

[2] Z. Chen and S. Lim, "Collecting typhoon disaster information from Twitter based on query expansion," *ISPRS Int. J. Geo-Inf.*, vol. 7, no. 4, p. 139, Apr. 2018.

[3] K. Garimella, G. D. F. Morales, A. Gionis, and M. Mathioudakis, "Quantifying controversy on social media," *ACM Trans. Social Comput.*, vol. 1, no. 1, pp. 1–27, Feb. 2018.

[4] M. J. Paul and M. Dredze, "Discovering health topics in social media using topic models," *PLoS ONE*, vol. 9, no. 8, Aug. 2014, Art. no. e103408.

[5] C. Boididou, S. Papadopoulos, L. Apostolidis, and Y. Kompatsiaris, "Learning to detect misleading content on Twitter," in *Proc. ACM Int. Conf. Multimedia Retr.*, Jun. 2017, pp. 278–286.

[6] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in tweets," in *Proc. 26th Int. Conf. World Wide Web Companion WWW Companion*, 2017, pp. 759–760.

[7] A. Tsakalidis, S. Papadopoulos, A. I. Cristea, and Y. Kompatsiaris, "Predicting elections for multiple countries using Twitter and polls," *IEEE Intell. Syst.*, vol. 30, no. 2, pp. 10–17, Mar. 2015.

[8] Y. Qian, X. Deng, Q. Ye, B. Ma, and H. Yuan, "On detecting business event from the headlines and leads of massive online news articles," *Inf. Process. Manage.*, vol. 56, no. 6, Nov. 2019, Art. no. 102086.

[9] L. Vomfell, W. K. Härdle, and S. Lessmann, "Improving crime count forecasts using Twitter and taxi data," *Decis. Support Syst.*, vol. 113, pp. 73–85, Sep. 2018.

[10] C. De Boom, S. Van Canneyt, and B. Dhoedt, "Semantics-driven event clustering in Twitter feeds," in *Making Sense Of Microposts*, vol. 1395. Florence, Italy: CEUR, 2015, pp. 2–9.

[11] Q. Kong, Y. Xu, I. Sobieraj, W. Wang, and M. D. Plumbley, "Sound event detection and time–frequency segmentation from weakly labelled data," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 4, pp. 777–787, Apr. 2019.

[12] M. Ravanbakhsh, E. Sangineto, M. Nabi, and N. Sebe, "Training adversarial discriminators for cross-channel abnormal event detection in crowds," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2019, pp. 1896–1904.

[13] Z. Yang, Q. Li, L. Wenyin, and J. Lv, "Shared multi-view data representation for multi-domain event detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 5, pp. 1243–1256, May 2019.

[14] T. Kaneko and K. Yanai, "Event photo mining from Twitter using keyword bursts and image clustering," *Neurocomputing*, vol. 172, pp. 143–158, Jan. 2016.

[15] K. Ntalianis and N. Doulamis, "An automatic event-complementing human life summarization scheme based on a social computing method over social media content," *Multimedia Tools Appl.*, vol. 75, no. 22, pp. 15123–15149, Nov. 2016.

[16] S. Zhao, Y. Gao, G. Ding, and T.-S. Chua, "Real-time multimedia social event detection in microblog," *IEEE Trans. Cybern.*, vol. 48, no. 11, pp. 3218–3231, Nov. 2018.

[17] X. Wang, F. Zhu, J. Jiang, and S. Li, "Real time event detection in Twitter," in *Proc. Int. Conf. Web-Age Inf. Manage.* Berlin, Germany: Springer, 2013, pp. 502–513.

[18] J. Weng and B.-S. Lee, "Event detection in Twitter," in *Proc. 5th Int. AAAI Conf. Weblogs Social Media*, 2011, pp. 1–22.

[19] P. Meladianos, C. Xypolopoulos, G. Nikolentzos, and M. Vazirgiannis, "An optimization approach for sub-event detection and summarization in Twitter," in *Proc. Eur. Conf. Inf. Retr.* Cham, Switzerland: Springer, 2018, pp. 481–493.

[20] B. Manaskasemsak, B. Chinthanet, and A. Rungsawang, "Graph clustering-based emerging event detection from Twitter data stream," in *Proc. 5th Int. Conf. Netw., Commun. Comput. ICNCC*, 2016, pp. 37–41.

[21] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: http://arxiv.org/abs/1301.3781

[22] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.

[23] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[24] Y. Meng, J. Huang, G. Wang, C. Zhang, H. Zhuang, L. Kaplan, and J. Han, "Spherical text embedding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8206–8215.

[25] M. Ceccarello, C. Fantozzi, A. Pietracaprina, G. Pucci, and F. Vandin, "Clustering uncertain graphs," *Proc. VLDB Endowment*, vol. 11, no. 4, pp. 472–484, Dec. 2017.

[26] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan, "Algorithms for facility location problems with outliers," in *Proc. SODA*, 2001, pp. 642–651.

[27] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, "On modularity clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 2, pp. 172–188, Feb. 2008.

[28] S. vanDongen, "A cluster algorithm for graphs," Inf. Syst., CWI, Amsterdam, The Netherlands, Tech. Rep. R 0010, 2000.

[29] L. M. Aiello, G. Petkos, C. Martin, D. Corney, S. Papadopoulos, R. Skraba, A. Goker, I. Kompatsiaris, and A. Jaimes, "Sensing trending topics in Twitter," *IEEE Trans. Multimedia*, vol. 15, no. 6, pp. 1268–1282, Oct. 2013.

[30] M. Hasan, M. A. Orgun, and R. Schwitter, "A survey on real-time event detection from the Twitter data stream," *J. Inf. Sci.*, vol. 44, no. 4, pp. 443–463, Aug. 2018.
[31] M. Hasan, M. A. Orgun, and R. Schwitter, "Real-time event detection from the Twitter data stream using the TwitterNews+ framework," *Inf. Process. Manage.*, vol. 56, no. 3, pp. 1146–1165, May 2019.
[32] Z. Saeed, R. A. Abbasi, O. Maqbool, A. Sadaf, I. Razzak, A. Daud, N. R. Aljohani, and G. Xu, "What's happening around the world? A survey and framework on event detection techniques on Twitter," *J. Grid Comput.*, vol. 17, no. 2, pp. 279–312, 2019.
[33] T. H. Nguyen and R. Grishman, "Graph convolutional networks with argument-aware pooling for event detection," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 5900–5907.
[34] G. Bekoulis, J. Deleu, T. Demeester, and C. Develder, "Sub-event detection from Twitter streams as a sequence labeling problem," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol., (Long Short Papers)*. vol. 1, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 745–750. [Online]. Available: https://www.aclweb.org/anthology/N19-1081
[35] R. Xie, F. Zhu, H. Ma, W. Xie, and C. Lin, "CLEar: A real-time online observatory for bursty and viral events," *Proc. VLDB Endowment*, vol. 7, no. 13, pp. 1637–1640, Aug. 2014.
[36] W. Wei, K. Joseph, W. Lo, and K. M. Carley, "A Bayesian graphical model to discover latent events from Twitter," in *Proc. 9th Int. AAAI Conf. Web Social Media*, 2015, pp. 1–10.
[37] J. Guo and Z. Gong, "A density-based nonparametric model for online event discovery from the social media data," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 1732–1738.
[38] C. Zhang, L. Liu, D. Lei, Q. Yuan, H. Zhuang, T. Hanratty, and J. Han, "TrioVecEvent: Embedding-based online local event detection in geo-tagged tweet streams," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 595–604.
[39] H. Becker, M. Naaman, and L. Gravano, "Beyond trending topics: Real-world event identification on Twitter," in *Proc. 5th Int. AAAI Conf. Weblogs Social Media*, 2011, pp. 1–17.
[40] H.-J. Choi and C. H. Park, "Emerging topic detection in Twitter stream based on high utility pattern mining," *Expert Syst. Appl.*, vol. 115, pp. 27–36, Jan. 2019.
[41] B. O'Connor, M. Krieger, and D. Ahn, "Tweetmotif: Exploratory search and topic summarization for Twitter," in *Proc. 4th Int. AAAI Conf. Weblogs Social Media*, 2010, pp. 1–4.
[42] Z. Ghaemi and M. Farnaghi, "A varied density-based clustering approach for event detection from heterogeneous Twitter data," *ISPRS Int. J. Geo-Inf.*, vol. 8, no. 2, p. 82, Feb. 2019.
[43] X. Zhang, X. Chen, Y. Chen, S. Wang, Z. Li, and J. Xia, "Event detection and popularity prediction in microblogging," *Neurocomputing*, vol. 149, pp. 1469–1480, Feb. 2015.
[44] N. D. Doulamis, A. D. Doulamis, P. Kokkinos, and E. M. Varvarigos, "Event detection in Twitter microblogging," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2810–2824, Dec. 2016.
[45] P. Meladianos, G. Nikolentzos, F. Rousseau, Y. Stavrakas, and M. Vazirgiannis, "Degeneracy-based real-time sub-event detection in Twitter stream," in *Proc. 9th Int. AAAI Conf. Web Social Media*, 2015, pp. 1–10.
[46] D. T. Nguyen and J. E. Jung, "Real-time event detection for online behavioral analysis of big social data," *Future Gener. Comput. Syst.*, vol. 66, pp. 137–145, Jan. 2017.
[47] C. Li, A. Sun, and A. Datta, "Twevent: Segment-based event detection from tweets," in *Proc. 21st ACM Int. Conf. Inf. Knowl. Manage. CIKM*, 2012, pp. 155–164.
[48] K. Orkphol and W. Yang, "Word sense disambiguation using cosine similarity collaborates with Word2vec and WordNet," *Future Internet*, vol. 11, no. 5, p. 114, May 2019.
[49] R. Angles and C. Gutierrez, "Survey of graph database models," *ACM Comput. Surv.*, vol. 40, no. 1, pp. 1–39, Feb. 2008.
[50] E. Amigó, J. C. De Albornoz, I. Chugur, A. Corujo, J. Gonzalo, T. Martín, E. Meij, M. De Rijke, and D. Spina, "Overview of replab 2013: Evaluating online reputation monitoring systems," in *Proc. Int. Conf. Cross-Lang. Eval. Forum Eur. Lang.* Berlin, Germany: Springer, 2013, pp. 333–352.
[51] S. A. Curiskis, B. Drake, T. R. Osborn, and P. J. Kennedy, "An evaluation of document clustering and topic modelling in two online social networks: Twitter and reddit," *Inf. Process. Manage.*, vol. 57, no. 2, Mar. 2020, Art. no. 102034.
[52] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

AARZOO DHIMAN received the bachelor's degree from Kurukshetra University, in 2014, and the M.Tech. degree from the National Institute of Technology at Kurukshetra, Kurukshetra, India, in 2016. She is currently pursuing the Ph.D. degree with IIT Roorkee, Roorkee, India. Her main research interests include datamining, text data analytics, and social media data analytics. She was a recipient of the UGC-NET Fellowship for Doctoral Research. She has presented her work in several reputed conferences, including SIG-KDD.

DURGA TOSHNIWAL (Member, IEEE) received the bachelor's degree in engineering and the M.Tech. degree from the National Institute of Technology at Kurukshetra, Kurukshetra, and the Doctor of Philosophy degree from IIT Roorkee, Roorkee, India. She is currently an Associate Professor with the Department of Computer Science and Engineering, IIT Roorkee. She is also associated with IBM Research India. She has published her research work in several inter-national journals and conferences. Her research interests include data mining and databases, mining time series, data streams and social media data, privacy preserving DM, soft computing in DM, and machine learning. She is a fellow of ACM. She has received various awards and honors. She has attended, chaired sessions, and presented her work in several reputed international conferences in USA, U.K., Australia, and Europe.