Tracing Topic Transitions with Temporal Graph Clusters

Xiaonan Jing, Qingyuan Hu, Yi Zhang, Julia Taylor Rayz
Department of Computer and Information Technology
Purdue University
West Lafayette, IN, USA
{jing, hu528, zhan3050, jtaylor1}@purdue.edu

Abstract

Twitter serves as a data source for many Natural Language Processing (NLP) tasks. It can be challenging to identify topics on Twitter due to continuous updating data stream. In this paper, we present an unsupervised graph based framework to identify the evolution of sub-topics within two weeks of real-world Twitter data. We first employ a Markov Clustering Algorithm (MCL) with a node removal method to identify optimal graph clusters from temporal Graph-of-Words (GoW). Subsequently, we model the clustering transitions between the temporal graphs to identify the topic evolution. Finally, the transition flows generated from both computational approach and human annotations are compared to ensure the validity of our framework.

I. Introduction

Continuously updating data streams make it challenging to identify real-time topics from platforms like Twitter. Previously, topic identification has mainly been studied on static dataset (Stoyanov and Cardie 2008; Lo, Chiong, and Cornforth 2017; Pappagari, Villalba, and Dehak 2018). However, oftentimes, real-world events are dynamic. During a continuously evolving event, the center of topic can shift as new information being updated throughout the event duration. We are interested in learning the underlying structure of how an event unfolds in the online community, especially when the data are limited for studying user behaviors.

Stream based event detection aims to identify a sequence of temporal states of the event(s). Given a continuous event across a set of timepoints $\{t_1, t_2, ..., t_n\}$, we define a temporal (sub)-event as the state s_i of the event at any timepoint $t_i (i < n)$, with t_n being the final timepoint whereas the event has no further updates. One of the biggest challenges in stream based detection lays in locating the temporal state boundary across the timepoints. When a dynamic event evolves over time, the temporal state s_1 may remain unchanged across several timepoints, however, suddenly converting to state s_2 then subsequently emerging to s_3. In other words, the temporal state at each timepoint is not independent of each other. There is a transition between the states when a change is to occur. Thus, the detection of a significant change in state becomes crucial in stream based tasks. Previously, burst based detection (McMinn and Jose 2015; Kaneko and Yanai 2016) and anomaly based detection (Guille and Favre 2015; Fedoryszak et al. 2019) have been explored. Burst based detection exploits a frequency based approach, which does not emphasize the semantic content of the events. On the other hand, anomaly based detection focuses on the change of semantic topic in textual content. In this paper, we are interested in the latter type, which traces the semantic topic change in a continuous time space.

We propose to employ graph structure to model temporal states due to its flexibility in relationship assignment and scalability in computational cost. Graphs have been adopted for similar tasks in event identification (Meladianos et al. 2015; Jing and Rayz 2020). A graph $G = (V, E)$ generally consists of a set of vertices V and a set of edges E which connects the vertices. The vertices can be words, sentences, or documents, and the edges can be used to model the statistical or semantic relationships between the textual elements. Our approach utilizes Graph-of-Words (GoW) to construct a temporal graph for tweets content at each timepoint, allowing the graph clustering to group temporal topics. Consequently, we develop topic transition flow by modeling cluster transitions at global level across all timepoints. Our main contributions are: 1) development of a clustering with nodes removal algorithm to find the optimal graph clusters over topical dataset; 2) improvement on cluster transition modeling to simulate the transition across all timepoints as well as taking re-emerging clusters into consideration; 3) visualization of topic transition flows in the time space.

II. Related Work

In this section, we review previous work on event identification on Twitter and graph based event modeling.

We start with event clustering graphs, for which Jin and Bai (2016) proposed a long document clustering approach utilizing a directed GoW for representing the word features contained in each document. The document clusters were generated based on the maximum common subgraphs between each document graph. In a similar attempt, Edouard et al. (2017) proposed an event clustering model which

Table 1: Statistics of the dataset split by day

Timepoint	8/19	8/20	8/21	8/22	8/23	8/24	8/25	8/26	8/27	8/28	8/29	8/30	8/31	9/1	9/2
Tweets	38	89	87	65	27	68	53	29	19	53	23	18	40	35	16
Nodes	139	218	167	177	69	189	204	101	65	137	72	56	122	128	113

leveraged named entities (NE) based directed GoW structure. The GoW was improved by using surrounding context of the graph nodes, NE, to enrich node level information. The approach is capable to automatically detect different events with the same keywords without any prior knowledge. Jinarat et al. (2018) employed a GoW combined with pretrained Word2Vec embedding (Mikolov et al. 2013) for tweet clustering. The Word2Vec similarity served as a metric for edge removal in generating tweet clusters. However, since abbreviations and hashtags occur regularly in tweets, pre-trained embeddings can be vulnerable to these irregularities which may not present in the training data.

Extracting event streams of an ongoing event from Twitter has a goal of detecting how an event unfolds as people post updates. Meladianos et al. (2015) improved the GoW approach by integrating the tweet length with the global co-occurrence frequency to identify the sub-events of a World Cup match on Twitter. Tweets containing the top k degenerated subgraph were used to describe each sub-event. Fedoryszak et al. (2019) proposed an interpretation of Twitter event streams - a chain of clustered trending entities arranged in chronological order. Additionally, Fedoryszak et al. overcame the limitation of lack of coverage in event detection with the aid of Twitter's internal knowledge graph (KG). Jing and Rayz (2020) introduced Graph of Tweets (GoT) for modeling popular events with both word and document level structures. GoT treat a tweet as a collection of conceptualized token nodes, whereas tokens of contextual similarities were merged prior to the GoT construction. Popular sub-events were extracted by detecting cliques among the similar tweet nodes.

Last but not least, we review previously proposed event representation on Twitter as there has not been a formal definition due to the various nature of the tasks. Hashtag based event identifications (Feng et al. 2015; Yang and Rayz 2018) treat an event as "a group of hashtags that focus on the same topic". Another approach utilizes NE to define event, whereas each NE is treated as an individual event and a set of NE as a merged event (McMinn and Jose 2015). Text triplet (subject, predicate, object) has also been adopted to describe a Twitter event (Tonon et al. 2017). In such approaches, OpenIE (Angeli, Johnson Premkumar, and Manning 2015) has been one of the top candidates for triplet extraction. Finally, embedding based approach which treats each tweet embedding as an event has also been explored (Dhingra et al. 2016). All of the above representations have their pros and cons – hashtags are excellent carries of topical information, but they may not be present in every tweet; NE can support details of the event, but they may also cause crucial information to be excluded (i.e. pronouns which are often subjects of an event); triplets can provide relational information, but entities that are not involved in a triplet cannot be captured; Embeddings allow efficient mathematical computations but also requires an adequate amount of data to train. We combine hashtags, NE and nouns in this paper to represent Twitter topics due to the limited amount of data we could collect for our experiment.

III. Proposed Method

We are interested in identifying topic transitions in specified events. We break the task into the following steps: 1) constructing a temporal graph for each timepoint; 2) applying clustering with node removal on each temporal graph; 3) modeling cluster transition flow across timepoints.

Dataset

Opportunistically, we chose to model local responses to the on-going event "COVID-19" for a short duration. Thus, tweets were collected from a local community from Aug 19th to Sep 2nd. "COVID-19" related tweets were identified by matching a set of manually selected hashtags for the corpus. The tweets of interests are pre-processed with Stanford CoreNLP [1] to annotate the part-of-speech and named entities. The statistics of the dataset is shown in Table 1.

Graph Construction

We treat each day as a timepoint and split the collected dataset based on the timestamp of the tweets. Temporal graphs are constructed using GoW with the nodes being the unique nouns or named entities extracted from tweets of the day. Furthermore, we employ normalized Point-wise Mutual Information (PMI) value as the edge weights between two nodes (1). In Equation (1), the marginal probabilities $p(x)$ and $p(y)$ and the joint probability $p(x,y)$ are computed as the proportions of the occurrence of tokens x and y in a total of N tweets, where n_x, n_y, and n_{xy} denote the (joint) frequency of tokens x and y. For consistency and computational efficiency, we further normalize the PMI with self-information $h(x,y)$ which sets the boundary of the PMI value to $[-1,1]$ (2).

$$pmi = log\frac{p(x,y)}{p(x)p(y)} = log\frac{n_{xy}}{n_x n_y}N \tag{1}$$

$$npmi = \frac{pmi}{h(x,y)} = \frac{pmi}{-logp(x,y)} \tag{2}$$

Clustering with Node Removal

For the dataset used in this paper, nodes that are closely related to the fetching keywords of the tweets tend to co-occur with every other node, while their neighboring nodes might observe no connections between each other. We refer to this type of nodes as the bridging nodes in this paper. Many common graph clustering methods, such as spectral

[1]Stanford CoreNLP ver. 3.9.2.

clustering (Ng et al. 2002) and highly connected subgraph clustering (Hartuv and Shamir 2000), achieve the grouping from graph cutting. One drawback of applying cutting based algorithm on graphs with bridging nodes is that no obvious local structures can be observed due to the inter-connectivity introduced by these nodes. More precisely, the cluster assignments for the neighboring nodes of a bridging node tend to fail as the graph cannot be cut in an appropriate way. As a result, the graph cutting mechanism tends to cluster each node into individual cluster. Thus, we propose to exclude the bridging nodes from the GoW during the clustering process, and treat them as belonging to each resulting clusters which have at least one edge in between. We locate a bridging node by its clustering coefficient $C_i \in [0, 1]$ (Watts and Strogatz 1998). As defined in Equation 3, clustering coefficient measures the embeddedness of a single nodes among its neighbors. A larger C_i indicates the neighbors of i tend to be more connected to form a community. In our case, the smaller the C_i is, the more likely the node is to serves as bridging node.

$$C_i = \frac{2e_i}{k_i(k_i - 1)} \tag{3}$$

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) \delta(c_i, c_j) \tag{4}$$

- e_i: number of edges between the neighbors of node i
- k_i and k_j: degree of node i and j
- A_{ij}: the edge weight between nodes i and j
- $2m$: sum of all edge weights
- c_i and c_j: communities of nodes i and j
- δ: an indicator function, $\delta = 1$ if $c_i = c_j$, $\delta = 0$ otherwise

To achieve an optimal clustering, we determine the number of bridging nodes to exclude by incrementally removing the denser node from the graph based on a clustering quality metric. Modularity (Newman 2006) is a common metric used for measuring community quality in graph theory. Given a partitioning of graph G, modularity $Q \in [-1, 1]$ computes the difference between actual and expected number of edges within groups (Equation 4). A larger modularity value Q indicates more significant community structure. Algorithm 1 outlines our method for finding optimal clustering. During each iteration, a node with the lowest C_i is removed (with its edges) from the graph G and the rest of the subgraph is clustered. The modularity value Q is computed on the subgraph to determine the current clustering quality. The best clustering is achieved at the (first) max Q value.

We adopt the random walk based Markov Clustering (MCL) (Van Dongen 2008) as our choice of graph clustering algorithms over other common graph cutting based clustering algorithms due to the drawbacks mentioned above. MCL simulates the stochastic flow in a graph which makes it more scalable. Furthermore, unlike other clustering algorithms, the number of clusters does not need to be pre-determined in MCL.

Modeling Cluster Transitions

In many stream based data analysis tasks, modeling cluster transitions is the key to identify the evolution of the target of interests. Given a timepoint t_i, a cluster transition can be defined as "the change experienced by a cluster that has been discovered at an earlier timepoint" (Spiliopoulou et al. 2006). Previously proposed frameworks such as MONIC (Spiliopoulou et al. 2006) and MEC (Oliveira and Gama 2012) define a set of transition types between clusters across consecutive timepoints and use a matching function with a threshold to identify these types. We adopt the basic transition types defined in MONIC and MEC, and further extend them to make the framework more robust for our task. From timepoint t_i to t_{i+1}, we define pairwise transition types for consecutive timepoints in Table 2 (first 7 types), where X and Y are clusters at timepoints t_i and t_{i+1} respectively, α is the threshold for the $match$ function to measure the overlaps between two clusters. In addition to transition types defined between pairs of consecutive timepoints, we introduce another transition type, namely, "a cluster has re-emerged", which measures the transition across non-consecutive timepoints. To visualize the transitions of a cluster in a global view, we model the set of transitions starting at a newly emerged cluster as a flow. In other words, we treat the pairwise transitions as the edges between the cluster nodes across different timepoints. When a transition exists between two cluster nodes, an edge with the transition type as value is assigned to connect them. It should be noted that to be computationally consistent with the basic transition types, the re-emergence transition is matched in respect of the last node in the sequence of the consecutive transitions.

Algorithm 1 Finding Optimal Graph Clustering

```
for node $v_i \in V : \{v_1, v_2, ..., v_m\}$ do
    $C_i = \textbf{Equation\_3}(v_i)$
end for
$V = \textbf{sort\_asc}(V, key = C_i)$
$best\_clustering = $ None
$Q\_max = -1$
for $v_i \in V$ do
    $G.\textbf{remove}(v_i)$
    $clusters = \textbf{Clustering}(G)$
    $Q = \textbf{Equation\_4}(clusters)$
    if $Q > Q\_max$ then
        $Q\_max = Q$
        $best\_clustering = clusters$
    end if
end for
return $best\_clustering$
```

Evaluation

We validate the topic transition results from our computer generated approach by applying the same transition framework to human annotated clusters. Each tweet is annotated with at most three noun (-phrase) labels which summarize the tweet the most by a human expert. The majority of the labels are directly selected from the tweets. Out-of-content labels are only generated when the tokens in a tweet cannot meaningfully summarize it.

A preliminary inspection over the manually and computationally generated clusters showed a mismatch in quantity

Table 2: Cluster transition types for cluster X at timepoint t_i

Transition Type	**Mathematical Definition**
the cluster stays unchanged	$X \to X'$, where $X' = match_\alpha(X)$
the cluster is absorbed	$X \to Y$, where $match_\alpha(X) \subset Y$ and $X - match_\alpha(X) \not\subset Y$
the cluster is dissolved	$X \to Y$, where $match_\alpha(Y) \subset X$ and $Y - match_\alpha(Y) \not\subset X$
the cluster splits into multiple clusters	$X \to \{Y_1, Y_2, ..., Y_m\}$, where $\bigcup_1^m Y_j = match_\alpha(X)$
the cluster is merged from multiple clusters	$\{X_1, X_2, ..., X_n\} \to Y$, where $\bigcup_1^n X_i = match_\alpha(Y)$
the cluster disappears	$X \to \emptyset$
a new cluster has emerged	$\emptyset \to Y$
a cluster has re-emerged	$X \to \emptyset \to X'$, where $X' = match_\alpha(X)$

Table 3: Statistics of the GoW During Clustering

Timepoint	8/19	8/20	8/21	8/22	8/23	8/24	8/25	8/26	8/27	8/28	8/29	8/30	8/31	9/1	9/2
$\bar{C}_{rm}$	0.41	0.47	0.47	0.48	0.46	0.48	0.39	0.74	0.33	0.41	0.43	0.4	0.44	0.42	0.28
$\bar{C}_{all}$	0.95	0.89	0.93	0.91	0.93	0.92	0.93	0.96	0.95	0.92	0.94	0.93	0.94	0.96	0.96
$\bar{C}_{best}$	0.99	0.99	0.99	0.99	0.99	0.99	0.99	1.0	0.99	0.99	0.98	0.98	0.98	0.996	0.98
$\%_{rm}$	7.19	18.81	11.98	16.95	11.59	14.81	10.78	16.83	6.15	12.41	6.94	8.93	8.2	5.47	3.54

of labels. In addition, we observed several co-occurring topics in the annotated clusters due to retweets. To merge the co-occurring topics in human annotated clusters, we employ frequent itemsets (Hornik, Grün, and Hahsler 2005) to discover strongly associated topics. Support *supp* is calculated as an indication of how frequently an itemset appears in the data (Equation 5). If two itemsets share the same *supp* value and one itemset is the subset of another, we merge the subset into its parent.

$$supp(X) = \frac{|t \in T; X \subseteq t|}{|T|} \quad (5)$$

- T: a set of transactions of a given dataset.

It should be noted that the methods of clustering between computer generated and human-annotated data are very different. However, the trends should be visible regardless of the methods, provided that the clusters are done well.

IV. Experimental Results

The number of nodes distributed in each temporal GoW separated by timepoint is shown in Table 1. As previously mentioned, we are interested in learning the responses and interests of a local community, and the real-world data we collected is limited in both quantity and quality. Thus, the number of nodes in each GoW varies largely depending on the community's activity of that day. Reporting global average based results would not guarantee an accurate evaluation to our framework. We instead will report results by timepoint and evaluate our framework by trend based comparison between computer and human generated clusters to ensure the validity of this work.

Clustering with Node Removal

To summarize, the average percentage of bridging nodes removed in each temporal GoW is 10.71% with an average C_i of 0.44. Timepoint specific data of the clusterings is shown in Table 3, where $\bar{C}_{rm}$, $\bar{C}_{all}$, $\bar{C}_{best}$ denote the average C_i for removed nodes, the original GoW, and the best clustered subgraph respectively; and $\%_{rm}$ denotes the percentage of nodes removed. The best subgraphs across all timepoints showed an increase in average C_i after the removal of bridging nodes, which confirms that the global embeddedness of the graph have become stronger. It is noteworthy that many of the best subgraphs achieve a nearly 1.0 (maximum boundry) average C_i, which suggests that the neighbors of each node in the subgraph are inter-connected. This can occur when the resulting subgraph is a complete graph or consists several fully connected components which are not interconnected. Our results are latter. Another metric should be considered for future tasks as the clustering coefficient C_i lacks the ability to differentiate the structural characteristics since it only considers the internal variance of the clusters.

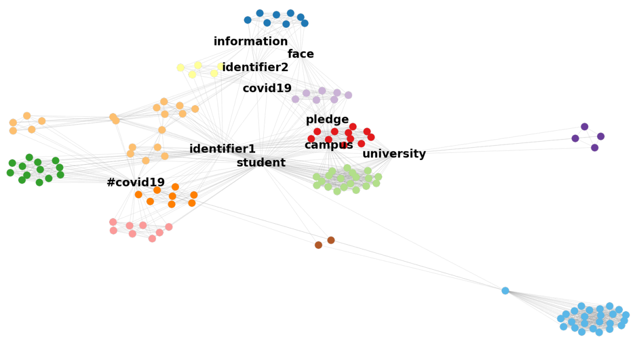
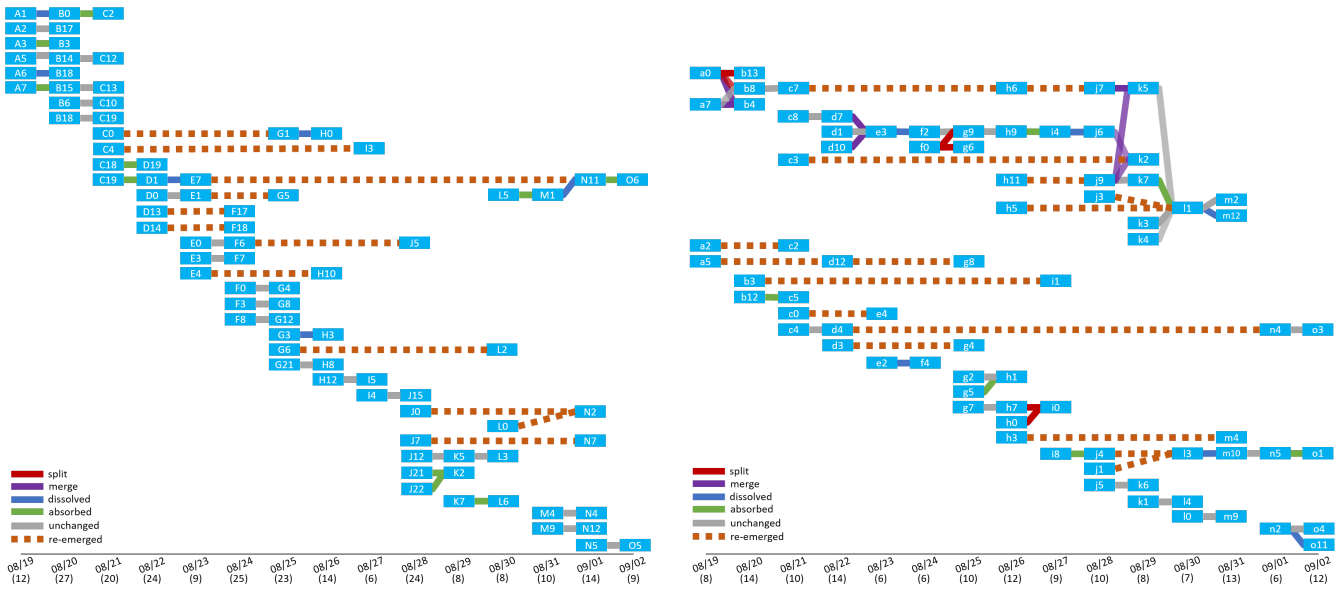


Figure 1: An instance of event graph for Aug 19, 2020 with 12 clusters. Each color represents a cluster and the bold (anonymized) words are the bridging nodes.

In addition, we observed consistent converging trends among the modularity plots over all timepoints. As the nodes with lower C_i get excluded from the subgraph, the resulting clustering quality starts to increase and gradually converges. Once a maximal modularity value is achieved, re-



(a) Computed Clusters (b) Human Annotated Topics

Figure 2: Cluster progression charts for (a) computed clusters vs. (b) human annotated topics from Aug. 19, 2020 to Sep. 02, 2020. Numbers below each date indicate the total number of clusters for the given date.

moving more bridging nodes will cause a decay in quality. Currently, our approach chose the result at the maximal modularity value as the optimal clustering. To increase generalizability and reduce computational cost, Algorithm 1 can be updated to stop when the modularity value starts to converge. Figure 1 illustrates an example clustering on Aug 19.

Comparing Cluster Transition

We applied the metrics defined in Table 2 to construct transition flows for both the computer generated clusters and human annotated label sets. An opportunistic threshold of $\alpha = 2/3$ is used for the $match$ function to determine if two clusters are considered as the same, no other threshold have been attempted. The computational approach generated 34 independent transition flows with an average length of 2.41 timepoints per flow, while 151 clusters only exist across a single timepoint. On the other hand, the human annotation suggests that there are 17 independent transition flows with an average length of 4.53 timepoints per flow, and 68 sets only exist across a single timepoint.

It should be noted that the computer generated clusters are based on nouns and NE from tweets with no order in mind, while human-annotated clusters are based on three summarized terms per tweet. Thus, computer generated clusters group all topic per day together irrespective of which tweet they come from, while human annotated ones have an additional layer of summarization: each tweet to keywords. The mismatch in the number of clusters can thus be explained by this additional layer of complexity, and should not necessarily be treated as a negative result. What is more interesting to see is the patterns that emerge from repeating topics, within both layers, throughout two weeks of data.

Figure 2 demonstrates the cluster progression charts of the computational approach and human annotated topics side by side. It can be seen that many clusters, both computer and human generated, reappear overtime. One clear outlier is a sequence of human-annotated clusters that contains a series of various transitions for over a week. A shared label between these clusters denotes the name of a policy that the local community is enforcing to ensure safety during the pandemic. Coincidentally, the same policy name has been detected as the bridging node in computer generated clusters in nearly all timepoints during the clustering process. Further analysis suggests that the longest sequence of transitions in both computer and human generated clusters can be traced to the same topic not mentioned here due to anonymity requirement. Despite the differences between the cluster generating mechanisms, both transition flows progress similarly in trends of emergence, re-appearance, and disappearance.

V. Conclusion and Future Work

In this paper, we proposed a graph based framework in modeling topic transitions of a local online community throughout two weeks of Tweets. Our proposed clustering with node removal approach attempted to resolve the drawback of the traditional hard clustering method on topical datasets. The cluster transition modeling provided insides on how topic flows can be traced in a continuous time space. Finally, the flow comparison between computer generated clusters and human annotated label sets demonstrated the applicability of our framework on real-world data.

Several improvements can be made in the future: 1) using a convergent schema to locate the best modularity value for the best subgraph to increase the model's robustness and reduce computational costs; 2) the assignments of membership for the bridging nodes to each resulting clusters as currently they are considered to belong to all inter-connected cluster with equal weights; 3) incorporating conceptual level information (i.e. knowledge graph) into the temporal graph to refine the events/topics clusters. However, even without these improvements, our framework is capable of tracing the evolution of the topics and the duration of the progressions as evident by comparison with human annotation.

References

Angeli, G.; Johnson Premkumar, M. J.; and Manning, C. D. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 344–354. Beijing, China: Association for Computational Linguistics.

Dhingra, B.; Zhou, Z.; Fitzpatrick, D.; Muehl, M.; and Cohen, W. W. 2016. Tweet2Vec: Character-Based Distributed Representations for Social Media. *arXiv:1605.03481 [cs]*. arXiv: 1605.03481.

Edouard, A.; Cabrio, E.; Tonelli, S.; and Le Thanh, N. 2017. Graph-based Event Extraction from Twitter. In *RANLP17 - Recent advances in natural language processing*.

Fedoryszak, M.; Frederick, B.; Rajaram, V.; and Zhong, C. 2019. Real-time event detection on social data streams. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2774–2782.

Feng, W.; Zhang, C.; Zhang, W.; Han, J.; Wang, J.; Aggarwal, C.; and Huang, J. 2015. Streamcube: Hierarchical spatio-temporal hashtag clustering for event exploration over the twitter stream. In *2015 IEEE 31st International Conference on Data Engineering*, 1561–1572.

Guille, A., and Favre, C. 2015. Event detection, tracking, and visualization in twitter: a mention-anomaly-based approach. *Social Network Analysis and Mining* 5(1):18.

Hartuv, E., and Shamir, R. 2000. A clustering algorithm based on graph connectivity. *Information processing letters* 76(4-6):175–181.

Hornik, K.; Grün, B.; and Hahsler, M. 2005. arules-a computational environment for mining association rules and frequent item sets. *Journal of statistical software* 14(15):1–25.

Jin, C., and Bai, Q. 2016. Text clustering algorithm based on the graph structures of semantic word co-occurrence. In *2016 International Conference on Information System and Artificial Intelligence (ISAI)*, 497–502.

Jinarat, S.; Manaskasemsak, B.; and Rungsawang, A. 2018. Short text clustering based on word semantic graph with word embedding model. In *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*, 1427–1432.

Jing, X., and Rayz, J. T. 2020. Graph-of-tweets: A graph merging approach to sub-events identification. In *Proceedings of the IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*.

Kaneko, T., and Yanai, K. 2016. Event photo mining from twitter using keyword bursts and image clustering. *Neurocomputing* 172:143–158.

Lo, S. L.; Chiong, R.; and Cornforth, D. 2017. An unsupervised multilingual approach for online social media topic identification. *Expert Systems with Applications* 81:282–298.

McMinn, A. J., and Jose, J. M. 2015. Real-time entity-based event detection for twitter. In Mothe, J.; Savoy, J.; Kamps, J.; Pinel-Sauvagnat, K.; Jones, G.; San Juan, E.; Capellato, L.; and Ferro, N., eds., *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, 65–77. Cham: Springer International Publishing.

Meladianos, P.; Nikolentzos, G.; Rousseau, F.; Stavrakas, Y.; and Vazirgiannis, M. 2015. Degeneracy-based real-time sub-event detection in twitter stream. *ICWSM* 15:248–257.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In Burges, C. J. C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems*, volume 26, 3111–3119. Curran Associates, Inc.

Newman, M. E. 2006. Modularity and community structure in networks. *Proceedings of the national academy of sciences* 103(23):8577–8582.

Ng, A. Y.; Jordan, M. I.; Weiss, Y.; et al. 2002. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems* 2:849–856.

Oliveira, M., and Gama, J. 2012. A framework to monitor clusters evolution applied to economy and finance problems. *Intelligent Data Analysis* 16(1):93–111.

Pappagari, R.; Villalba, J.; and Dehak, N. 2018. Joint verification-identification in end-to-end multi-scale cnn framework for topic identification. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6199–6203. IEEE.

Spiliopoulou, M.; Ntoutsi, I.; Theodoridis, Y.; and Schult, R. 2006. Monic: modeling and monitoring cluster transitions. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 706–711.

Stoyanov, V., and Cardie, C. 2008. Topic identification for fine-grained opinion analysis. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, 817–824.

Tonon, A.; Cudré-Mauroux, P.; Blarer, A.; Lenders, V.; and Motik, B. 2017. Armatweet: Detecting events by semantic tweet analysis. In Blomqvist, E.; Maynard, D.; Gangemi, A.; Hoekstra, R.; Hitzler, P.; and Hartig, O., eds., *The Semantic Web*, 138–153. Cham: Springer International Publishing.

Van Dongen, S. 2008. Graph clustering via a discrete uncoupling process. *SIAM Journal on Matrix Analysis and Applications* 30(1):121–141.

Watts, D. J., and Strogatz, S. H. 1998. Collective dynamics of 'small-world'networks. *nature* 393(6684):440–442.

Yang, S.-F., and Rayz, J. T. 2018. An event detection approach based on twitter hashtags.