# TEAGS: Time-aware Text Embedding Approach to Generate Subgraphs

Saeid Hosseini, Saeed Najafipour, Ngai-Man Cheung, Hongzhi Yin, Mohammad Reza Kangavari, and Xiaofang Zhou

**Abstract**—Given a graph over which the contagions (e.g. virus, gossip) propagate, leveraging subgraphs with highly correlated nodes is beneficial to many applications. Yet, challenges abound. First, the propagation pattern between a pair of nodes may change in various temporal dimensions. Second, not always the same contagion is propagated. Hence, state-of-the-art text mining approaches ranging from similarity measures to topic-modeling cannot use the textual contents to compute the weights between the nodes. Third, the word-word co-occurrence patterns may differ in various temporal dimensions, which increases the difficulty to employ current word embedding approaches. We argue that inseparable multi-aspect temporal collaborations are inevitably needed to better calculate the correlation metrics in dynamical processes. In this work, we showcase a sophisticated framework that on the one hand, integrates a neural network based time-aware word embedding component that can collectively construct the word vectors through an assembly of infinite latent temporal facets, and on the other hand, uses an elaborate generative model to compute the edge weights through heterogeneous temporal attributes. After computing the intra-nodes weights, we utilize our Max-Heap Graph cutting algorithm to exploit subgraphs. We then validate our model through comprehensive experiments on real-world propagation data. The results show that the knowledge gained from the versatile temporal dynamics is not only indispensable for word embedding approaches but also plays a significant role in the understanding of the propagation behaviors. Finally, we demonstrate that compared with other rivals, our model can dominantly exploit the subgraphs with highly coordinated nodes.

**Index Terms**—time-aware word embedding, neural network, subgraph mining, generative latent models, propagation networks

✦

## 1 INTRODUCTION

PROPAGATION process over a graph of vertices is multi-disciplinary and involves with epidemiology [1], viral vigilante [2] [3], social media [4], and cybersecurity [5]. Given a graph $G$ as a propagation network over which the contagions (e.g. gossip, virus, and alarm) spread, a subgraph $g$ of $G$ is tightly connected if all the nodes in $g$ are highly correlated with each other. Identifying tightly connected subgraphs finds important applications in numerous domains. For example, highly correlated subgraphs are used (i) in transport systems to reorder the repairs and reduce the number of future defects [5] [6], (ii) in epidemiology to vaccinate influential nodes and prevent the spread of the disease [7], and (iii) in social media to manipulate (add or delete) the network edges and control outspreading of the gossips [8] [9]. Moreover, tightly connected subgraphs can be beneficial for pre-emptive schemes such as preventive medicine and preventative maintenance.

Numerous techniques [4] [10] [11] [12] have been proposed to compute the edge weights among each pair of the nodes in propagation networks that, roughly speaking, is the main step in exploiting of the subgraphs. This reduces the NP-hard problem of subgraph mining to *edge weight calculation* that is tailored by a *Graph-cut algorithm*. Although, the computing of the edge weights is conceptually simple, significant obstacles incurred by the specifications of the propagation process. Therefore, new approaches must be introduced to handle them. In the following, we present the *challenges* to illustrate such a need.

*Challenge 1 (Varied Temporal Pattern)*

The propagation pattern between a pair of nodes $(n_i, n_j)$ may differ in *various latent temporal facets* (e.g. hour, day) that are revealed as parts of the same whole. To exemplify the hour dimension, while the node $n_i$ may spread a contagion to $n_j$ during the day, the propagation may decay to some extent or cease throughout the night hours. The polymorphous *depth* feature even makes hour factor dependent on parent facets (i.e. day and week). However, the traditional remarkable models in inferring of the propagation processes [4] [12] [10] [11] only consider the *sequential* interlude attribute. The rule seems to be that the farther apart in time the two nodes spread a contagion in between, the less likely they are to collaborate tightly.

*Challenge 2 (Diverse Propagation Direction)*

Not only the type of contagion and the amount of the functional load can differ in various temporal facets, but owing to the temporal dimension specifications, the *propagation direction* may also alter. e.g., While the node $n_i$ influences node $n_j$ from Monday to Friday, the influence can either be eliminated or divert inversely during the weekends.

*Challenge 3 (Mismatched Node Contents)*

Unlike prior work [4] [10] [11] [12], not always the same contagion is propagated [13] to the subsequent influenced node. For instance, an *oil leakage* initially causes an *extra*

- S. Hosseini and N. Cheung are with the ST Electronics - SUTD Cyber Security Lab., Singapore University of Technology and Design, Singapore.
  Email: ngaiman_cheung@sutd.edu.sg, saeid.hosseini@uq.net.au
- Xiaofang Zhou and Hongzhi Yin are with School of Information Technology and Electrical Engineering, University of Queensland, Brisbane, Australia.
  Email: {zxf,h.yin1}@itee.uq.edu.au
- S. Najafi Pour, M. R. Kangavari, and S.Hosseini(Affiliated) are with computational cognitive model research lab., School of computer engineering, Iran University of Science and Technology, Iran.
  Email: saeed_najafi@comp.iust.ac.ir, kanagvari@iust.ac.ir

*vibration*, continues with *overheating*, and finishes with *engine failure*. As a result, *node-specific textual contents* in a propagation sequence may end up *mismatched*. Consequently, the textual contents associated with the pair of nodes will not gain sufficient statistical signals to utilize current text mining approaches (e.g., topic modeling [14] and other heuristics [15]). As a result, the correlation edge weight between the node pair will not be computed appropriately.

*Challenge 4 (Temporally Skewed Contents)*

Given the corpus $\mathcal{C}$ comprising the textual contents of the nodes, the co-occurrence matrix $X$ reports how each pair of words co-occurs in $\mathcal{C}$. The co-occurrence matrix is the essential element for vector representation. However, current state-of-the-art word vector representation models [16] [17] neglect the fact that the *word proximities* can be affected by the various latent temporal facets.

*Observation:* In order to study how the word co-occurrence patterns change in *various temporal dimensions*, we set up an observation on a dataset [15] of one million tweets in Australia. Tables 1 and 2 show the distribution probability of the word co-occurrences in the hour and season latent factors. As Table 1 demonstrates, almost one-third of the times the word *go* co-occurs with *work* during 6 to 11 am. Compared to those who start the work early in the morning, the night shift people tend more to *drive* to *work* in the evening. Moreover, from the co-occurrence of *Have+Tea*, we understand that more people have the tea in the morning. Finally, people *have+fun* more during 6-11 am that is affected by the weekend tweets.

TABLE 1
Co-occurrence probability - Hour dimension

| Word 1 | Word 2 | 0 - 5 | 6 - 11 | 12 - 17 | 18 - 23 |
|--------|--------|-------|--------|---------|---------|
| Go | Work | 0.295 | **0.324** | 0.119 | **0.260** |
| Drive | Work | 0.144 | **0.289** | 0.108 | **0.457** |
| Have | Tea | **0.303** | **0.383** | 0.147 | 0.165 |
| Have | Fun | **0.306** | **0.388** | 0.143 | 0.161 |

Also as Table 2 reports, the word *cold* mainly comes with *drink* during the summer and the people dispute about the *hot+day* and *hot+night* mostly during the spring and summer. Also, the distribution of *go+surf* exhibits that summer and fall are respectively the most popular seasons for surfing. Thus, the word pair co-occurrence patterns are temporally skewed and conceivably differ in various dimensions.

TABLE 2
Co-occurrence probability - Season dimension

| Word 1 | Word 2 | Spring | Summer | Fall | Winter |
|--------|--------|--------|--------|------|--------|
| Cold | Drink | 0.228 | **0.485** | 0.142 | 0.142 |
| Hot | Day | **0.265** | **0.371** | 0.220 | 0.142 |
| Hot | Night | **0.269** | **0.387** | 0.203 | 0.139 |
| Go | Surf | 0.215 | **0.268** | **0.364** | 0.151 |

*Contributions.* In this work, revealing the *utopian spirit of the time*, we state that the *multi-aspect temporal knowledge* [18] of the propagation network is indispensable for subgraph mining. This in turn benefits many real-world applications that deal with the spread of the contagions in various domains. According to the above discussion, three types of requirements are proposed to cope with the challenges in exploiting of the tightly connected subgraphs: 1) providing a better insight into the word co-occurrence patterns in multiple dimensions; 2) comprising various attributes to better understand discrete temporal dynamics [19]; 3)

appraising propagations in both directions; Such needs are met in Sections 4.1.4, 4.1.5, and 4.2.1.

To the best of our knowledge, we propose the first study on multi-aspect time-aware word embedding which aims to leverage the subgraphs from bidirectional propagation networks. Our contributions in this work are threefold:

- We develop a neural network based time-aware word embedding approach that can collectively construct the word vectors based on an infinite number of temporal dimensions.
- We devise a generative model that can compute the bilateral correlation weight between each pair of nodes through multi-aspect latent temporal facets.
- We design a Heap-based graph-cutting algorithm to maximize the correlation inside each of exploited subgraphs and simultaneously minimize the number of utilized edges in subgraph mining procedure.
- We propose a temporal-textual framework that can achieve better effectiveness in leveraging of the tightly connected subgraphs from bidirectional propagation networks.

### 1.1 Paper Organization

We organize the rest of this paper as follows: in Sec. 2, we briefly study the literature; in Sec. 3, we clarify the problem for subgraph mining in bilateral propagation networks and provide a framework overview; our approaches and experiments are described in Sec. 4 and Sec. 5 respectively. Finally, we conclude this paper and discuss future work in Sec. 6.

## 2 RELATED WORK

As we propose a time-aware word-embedding approach to infer propagation processes, the literature includes *Word embedding*, *Propagation inference*, and *Immunization policies*.

### 2.1 Word Embedding

*Word embedding* exploits a real-valued vector for each given word in the corpus. Such vectors can better discover the correlation between the words and tackle relevant challenges in various domains such as Natural Language Processing (NLP) [20], Information Retrieval (IR) [21], and Recommendation Systems (RS) [22] [23]. Traditional embedding methods include *Bag-of-Words*(BOW) [24] and *Latent Semantic Indexing* (LSI) [25]. BOW represents each document with its comprising words and while neglecting the order of vocabularies, relies on word frequencies. Conversely, LSI applies *Singular Value Decomposition*(SVD) to identify the patterns between terms and concepts. Moreover, the BOW model [26] can be extended using Expectation-Maximization(EM) [27] to eliminate the word ambiguity, which particularly benefits machine translation. Furthermore, the state-of-the-art Neural word embedding models (e.g. *Word2vec* [17] and *GloVe* [28]) extract meaningful syntactical and semantical regularities from word pairs. Similar to BOW, the word embedding approaches can further be equipped with other algorithms [14], [29], and [30]. For instance, Nguyen et al. [14] fuses *topic models* with embedding approaches to collectively generate a word either from Dirichlet multinominal component or from the continuous embedding module. Likewise, aiming to perform entity disambiguation, Zhu et al. [29] enrich the word embedding modules with Knowledge Graphs (KG) to make the best utilization of both

knowledge and corpus-based semantic similarities. Nevertheless, vector representation models are not only proposed for words [17], but also involve other data structures, like Concept-Vector [22]. In general, the co-occurrence matrix is the essential element in *count-based representation* methods. However, the word-word co-occurrence patterns are temporally skewed. Yet, many word embedding approaches [17] [16] [28] neglect the temporal dynamics. On the contrary, the time-oriented models [31] [32] utilize terminology in a single dimension to address various use-cases, including temporal word analogies, word-to-word relatedness [33], and semantical relations [33]. Nonetheless, the temporal word pair co-occurrences may alter in *diverse latent temporal facets* (e.g. hours, days, weeks, and month). Hence, in this paper, we devise a novel multifaceted time-aware word embedding approach which can conjointly infer word vectors through multiple temporal dimensions.

### TABLE 3
Literature

| Category | Approach | References |
|---|---|---|
| Word Embedding | Traditional Models | [24] [25] [26] [27] |
| | Neural Network | [17] [28] [21] |
| | Hybrid | [14] [22] [29] |
| | Time-oriented | [31] [32] [33] |
| Propagation Inference | Cascade | [34] [35] |
| | Threshold | [9] [36] [37] |
| | Time-oriented | [10] [11] [18] |
| Immunization Policy | Single Node | [8] [9] [38] [39] |
| | Group Based | [5] [40] [1] [7] [4] |
| | Preventative | [40] |
| | Time-oriented | [10] [11] |

## 2.2 Propagation Inference

A *Diffusion* or *propagation* process interprets how each contagion (e.g. alarm or virus) may spread in a graph. The conventional models suppose that the nodes spread the contagions equally [41]. In general, the propagation inference algorithms are of two types of *cascade* [34] and *threshold-based* [42] [6] [36] [9]. The threshold based approaches like *Linear Threshold* (LT) determine under what verge the contagion may outspread. Specifically, [37] claim the epidemic threshold as the first eigenvalue of the *adjacency matrix* that is associated with the diffusion graph. On the contrary, the cascade models examine the condition of each vertex during propagation phase: *Susceptible*, *Infected*, *Removed*, or *Vigilant*. Subsequently, depending on the status of each vertex, one can track how the contagions are transmitted. From immunity perspective the cascade models are divided into three categories [35]: without immunity ($SIS$), provisional immunity ($SIRS$), and perpetual immunity ($SIR$). Though, newer approaches amend either the number of contagions ($SI_1I_2R$) or the subgraph structure, e.g. *clique*, *chain*, and *star*. Furthermore, the contagions may spread in either or both directions [10].

For the time aspect, the retrospective *Sequential* attribute [10] [11] elucidates that the farther apart in-time a pair of vertices spread a contagion, the less correlation weight

will be assigned to the edge between them. However, we argue that the dynamical processes may vary in different temporal dimensions. Hence, we need to collectively infer propagations in various dimensions. Owing to the fact that the vector representation of the words in textual contents can be distorted by the *temporal permutations* [18], we recommend a diligent neural network based *multi-aspect time-aware embedding* module to mutually construct the word vectors through an infinite number of temporal dimensions.

## 2.3 Immunization Policies

Immunization aims to stop or slow down the spread of contagions. *Vaccination* [43] and *social distancing* [44] are the two common immunization tactics. Specifically, immunization can be applied on *individual* or *group* (i.e. subgraphs) of vertices. Single node models include *random vaccination* [38], *network manipulation* [8], [9], *data-driven* [13], and recently *non-spectral models* [39] [9]. Furthermore, even though leveraging of the subgraphs typically result in sub-optimal solutions and the exploited subgraphs may also reshape [40], the group-based inoculation policies [45] can better control the outspread. Nevertheless, our proposed research benefits immunization tactics in three ways: First, where prior works [10] [11] only consider the sequential temporal property, we devise a temporal-textual approach which infers the edge weights via miscellaneous time-oriented features. Second, our group-based approach can better restrain dynamical procedures [45]. Third, unlike limited directed networks [4], we track intra-nodes propagations bilaterally.

## 3 PROBLEM STATEMENT

In this section, we define the concepts and problems of the paper. We then explain our framework overview.

### 3.1 Preliminary Concepts

The preliminary definitions are introduced as follows:

#### 3.1.1 Definition 1 (node)

A Node with identifier $n_i \in \mathbb{N}$ is a vertex over which the contagions propagate.

#### 3.1.2 Definition 2 (alarm)

An alarm $a_i \in \mathbb{A}$ is recorded when a node gets infected by a contagion. Each alarm has an identity ($a_i$), pertinent node $n_m$, **category** $c_l$, infection **time** $a_i.t$, and **string** contents $a_i.s$. $\mathbb{A}$ represents the alarm sets associated with each of the nodes ($\mathbb{A} = \{A_1, A_2, ...., A_n\}$). Hence, $A_i$ denotes the set of times when the node $n_i$ has raised any of the alarms. As far as the type of the contagions is the same, the propagation can be admitted. In a *medical context*, influenza can spread in three forms of $A$, $B$, and $C$, but all are counted as the same virus.

#### 3.1.3 Definition 3 (propagation network)

The undirected graph $G = (\mathbb{N}, \mathbb{L})$ with the set of nodes $\mathbb{N}$ and links $\mathbb{L}$ comprises numerous interleaved *directed acyclic graphs* that reflect how contagions spread bidirectionally.

#### 3.1.4 Definition 4 (latent temporal facets)

The propagation behavior between a given pair of nodes $(n_i, n_j)$ can differ based on the set of *latent temporal facets*: $\mathbb{T} = \{z^1, z^2, ..., z^t\}$. While our proposed solution (Section 4.1.5) can accommodate an infinite number of temporal dimensions, owing to density we limit $\mathbb{T}$ to four dimensions of hour, day, week and month $\mathbb{T} = \{z^h, z^d, z^w, z^m\}$

### 3.1.5 Definition 5 (unifacet temporal slab)

Each latent temporal dimension $z^x$ can comprise $\eta$ splits $z^x = \{s_1^x, s_2^x, ..., s_\eta^x\}$, e.g. 12 splits for month dimension. Accordingly, the *Unifacet Temporal Slabs* are built via merging of the similar splits.

### 3.1.6 Definition 6 (multifacet temporal slab)

Given a set of unifacet temporal slabs belonging to $t$ hierarchical temporal dimensions, a *Multifacet Temporal Slab* is designated by a combination of $b$ unifacet temporal slabs.

### 3.1.7 Definition 7 (vector representation)

Given the corpus of textual contents $\mathcal{C}$, the word embedding can convey a real-valued vector $\vec{v_i}$ to every vocabulary $v_i \in \mathbb{V}$. Here $\vec{v_i}$ is the vector which represents the word $v_i$.
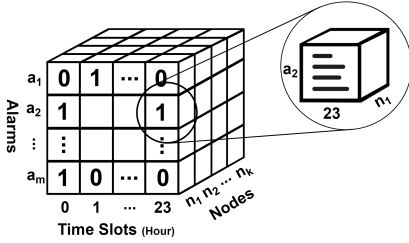


Fig. 1. $\mathbb{NTA}$ Cube

### 3.2 Problem Definition

The notations $\mathbb{N}$, $\mathbb{T}$, and $\mathbb{A}$ are respectively the set of **N**odes, **T**emporal facets, and **A**larms. Given a temporal dimension, the alarm time-stamps including the textual contents can be recorded in the dimension-specific $\mathbb{NTA}$ cube. Fig. 1 illustrates the hour dimension cube, where each cell includes the possible textual contents of the alarms and the binary value to signify whether the sample node $n_i$ has raised the alarm $a_j$ at the particular temporal split or not. Since each node can get infected by a limited number of contagions, the $\mathbb{NTA}$ cube can be extremely sparse. Consequently, we can put a *proposition* forward: *if two splits in a dimension are highly similar, we can predict unforeseen propagations in one of them based on the data recorded for the other.* Hence, we can demote the sparsity through merging of the similar splits. The problems addressed in this paper are as follows:

**Problem 1.** *(extracting multifacet temporal slabs) Given the set of nodes $\mathbb{N}$, alarms $\mathbb{A}$, and the temporal dimensions $\mathbb{T}$, our goal is to extract all multifacet temporal slabs through merging highly similar splits in each of t temporal dimensions.*

**Problem 2.** *(mining correlation weights) Given a set of temporal dimensions $\mathbb{T}$ and the set of multifacet temporal slabs $\tau^b$, our aim is to compute the edge weight between each pair of nodes $(n_i, n_j)$ using both temporal and textual data in propagation network.*

**Problem 3.** *(exploiting highly correlated subgraphs) Given a bilateral network $G$ over which the alarms ($\mathbb{A}$) spread, we aim to mine a set of subgraphs ($\mathbb{S}$), where the nodes in each of exploited subgraphs are highly correlated.*

Intuitively, the problem of subgraph mining (Prob. 3) can be divided into two steps: Firstly, to calculate the edge weights between nodes (Prob. 2). Secondly, to employ a graph-cut algorithm to optimize the number of exploited subgraphs and maximize the intra-subgraph correlations.

### 3.3 Framework Overview

Figure 2 illustrates our proposed unified framework that can exploit the subgraphs, with highly correlated nodes, from propagation networks. Through <u>offline</u> phase, both the *temporal* and *textual* information of the alarms is used

to retrieve the multi-aspect *grids* that reflect how similar each pair of temporal splits are. Subsequently, we utilize *hierarchical clustering* to combine similar intervals and form unified and then multi-aspect temporal *slabs*.
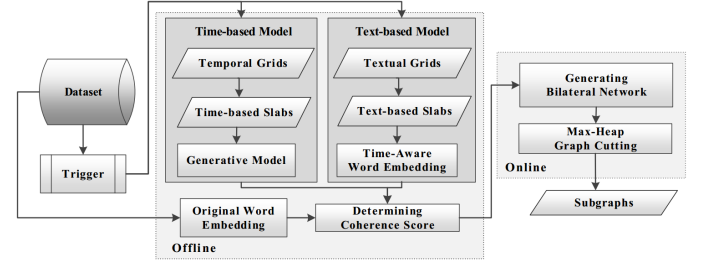


Fig. 2. Framework

In order to compute edge weights between nodes, On the one hand, we feed the *time-based slabs* to the *multifacet generative model* to compute the time-only correlations, and on the other hand, the *text-based temporal slabs* are consumed by the *time-aware word embedding* to address the challenge that the co-occurrence patterns of the word pairs may differ in various temporal aspects. Moreover, the *Original Word-Embedding* module takes the global co-occurrence matrix to retrieve non-temporal vector representation of the words in alarm contents. Nevertheless, the final *coherence score* can merge the edge weights computed by various methods.

In the <u>online</u> phase, the coherence score is used to merge the edge weights - computed by various models - to form a *single-edged undirected graph*, named as *Bilateral Network*. Finally, we employ the *Max-Heap Graph cutting algorithm* to leverage the subgraphs with highly correlated nodes.

## 4 METHODOLOGY

In general, exploiting of the subgraphs with the highest correlated nodes primarily requires a series of *isomorphic tests* to avoid double entries in the final list of subgraphs. Thus, *canonical labeling* uniquely distinguishes every subgraph with a vector of digits. Accordingly, where a pair of subgraphs is canonically labeled the same, they can be considered *isomorphic*. However, canonical labeling is computationally complex and declared as an *NP-complete* problem. Therefore, in this paper, we alternatively propose a diligent algorithm, which initially transfers the problem of frequent subgraph mining to the computation of edge-weights. Conclusively, we devise a straightforward graph-cut algorithm to effectively leverage the communities of meticulously correlated nodes. Nevertheless, since the likelihood for every vertex in propagation graph is assigned by one, as Eq. 1 shows, the probability for each node $n_i$ to spread a contagion to another node $n_j$ corresponds to their joint probability.

$$Pr(n_j|n_i) \propto Pr(n_i, n_j) \qquad (1)$$

We can collectively compute the edge weight between each pair of vertices (Fig. 3) using three algorithms, namely *Original Word Embedding* (Sec. 4.1.3), *Time-aware Word Embedding* (Sec. 4.1.4), and *Multifacet Generative Model* (Sec. 4.1.5).
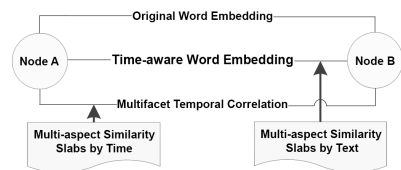


Fig. 3. Inner Nodes correlations

## 4.1 Offline Processing

### 4.1.1 Constructing Similarity Grids

Similarity grids consider the *Temporal* and *Textual* propagation rules in each temporal dimension to discover correlation rates between pair of splits. We employ two real-life datasets (Section 5.1) for empirical studies. **Textual**
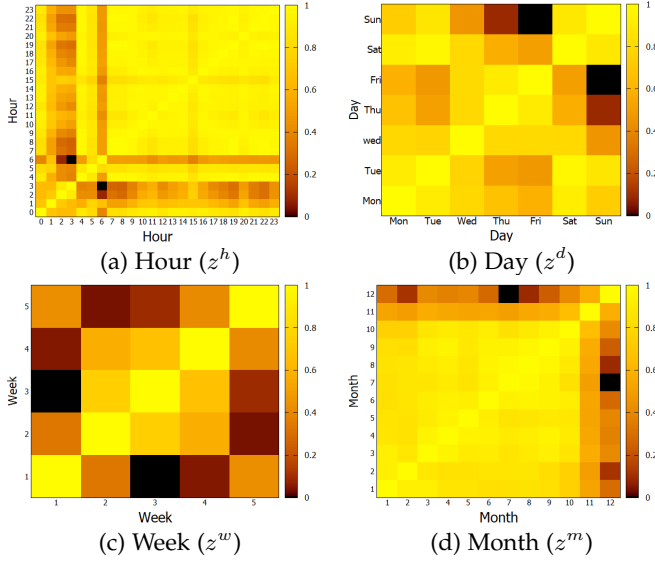


(a) Hour ($z^h$)  (b) Day ($z^d$)

(c) Week ($z^w$)  (d) Month ($z^m$)

Fig. 4. Similarity Grid - Textual Data - ST-0



(a) Hour ($z^h$)  (b) Day ($z^d$)

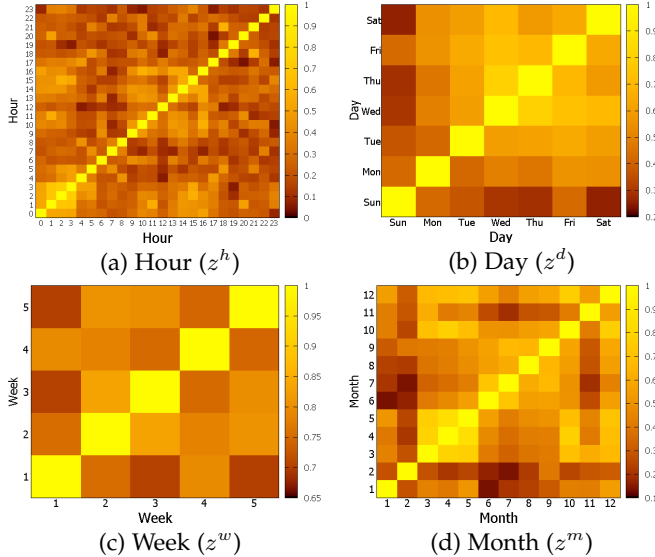(c) Week ($z^w$)  (d) Month ($z^m$)

Fig. 5. Similarity Grid - Temporal Data - ST-0

**Grid**: In order to construct the similarity grid using textual contents, we firstly aggregate the textual contents of all the nodes during each temporal split, say *5 am* in *hour* dimension. Hence, we associate an array of text with each latent facet. Note that, each cell in $\mathbb{NTA}$ cube additionally adds the textual contents of the alarms in the pertinent split. Subsequently, we employ a modified version of *TF-IDF*, to signify the weight of each word in every latent facet:

$$\hat{\mathbf{w}}(t_i, S_k^l) = \frac{f(t_i, S_k^l)}{Max_{(t \in S^l)}\{f(t, S_k^l)\}} \times Log\frac{N}{N(t_i)} \quad (2)$$

As verbalized in Eq. 2, $S_k^l$ constitutes the string contents of split $k$ in latent facet $l$. Also, $\frac{f(t_i, S_k^l)}{Max_{(t \in S^l)}\{f(t, S_k^l)\}}$ estimates the normalized term frequency in the split. Here, $N$ designates the total number of the splits and $N(t_i)$ is the number of splits comprising the term $t_i$. Given the computed list of

normalized weights, every split can be presented by a fixed-sized vector $\vec{S}_k^l$, wherein the entries show the weights of the terms. The split vectors are then employed by the *Cosine* metric to obtain the similarity between splits.

Figure 4 illustrates the textual similarity grids. Since the contrast between the textual contents of the splits in hour and month dimensions are low, our model grants more significance to other facets (day and week) for ST-0 dataset. **Temporal Grid**: In order to generate the temporal similarity grid for each latent facet (e.g. Hour), we find the nodes that are infected in each pair of the splits. Take the hour facet, where the same alarm type happens in both hours, we can claim an evidence that two hours are partially similar. In contrary, if the number of mutual alarm types between a pair of hours is zero, one can conclude that the two hours are mutually exclusive or at least we cannot decide whether we can cluster them together or not. Nevertheless, we use the *linear correlation measure* to compute the final similarity weight between each split pairs in each dimension. Note that the resulting unifacet temporal slabs are affected by the sampling model based on which the similarity evidences are collected. Accordingly, we employ the *stratified sampling model* [46]. Aiming to achieve $q$ similarity evidences between split pairs, we process $p\%$ of the nodes through a *replacing iterative process*. Furthermore, we utilize the *None-Negative Matrix Factorization (NMF)* to compensate missing entries in each similarity grid. Figure 5 demonstrates the time-only similarity grids for datasets ST-0. While neighboring intervals show identical similarity weights in recommendation systems [18] [47], we interestingly observe in Figure 5 that the similarity grids in diffusion networks differ and consequently need to get updated via the *Trigger* frequently.

### 4.1.2 Acquiring Temporal Slabs

Given the similarity grids, computed using temporal and textual evidences, and also the density of dataset, we employ the bottom-up *Hierarchical Agglomerative Clustering* (*HAC* via *complete linkage*) to retrieve temporal similarity slabs. In this way, we address Problem 1 and extract multifacet temporal slabs. The number of dimensions, e.g. quadrilateral facets ($z^h, z^d, z^w, z^m$) can be affirmed based on the sparsity of dataset. Nevertheless, the clustering *threshold* is utterly important as it can incorporate irrelevant temporal splits into the same slab or toss pertinent splits to separate clusters. To address this challenge, we merge the splits in each dimension based on various thresholds that can construct two series of slabs, i.e. *small* and *big clusters*. Subsequently, either of big or small clusters, that can ensure the best effectiveness in subgraph mining (Section 5.2), will be consumed by the inference models (Section 5.3.1).



(a) ST-0 Day HAC ($z^h$)  (b) ST-1 Day HAC ($z^d$)  (c) ST-0 Week HAC ($z^w$)  (d) ST-1 Week HAC ($z^m$)
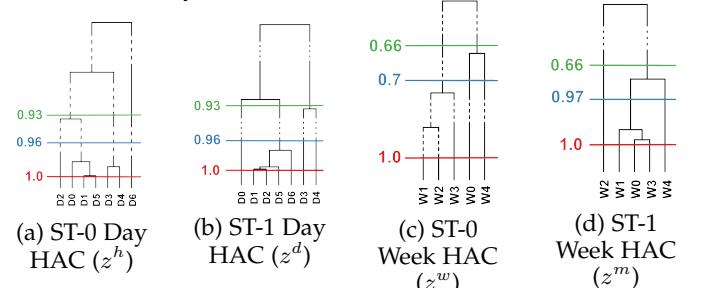
Fig. 6. Sample dendrograms - Big versus Small clusters

Considering two daily and weekly facets, Fig. 6 illustrates distinctive thresholds which will generate either of

5

small (middle line) or big clusters (upper line). Accordingly, Table 4 reports generated clusters in four-fold temporal dimensions. As empirically witnessed in Section 5.3.1, compared with small clusters, the big clusters can better exploit the tightly connected subgraphs in both datasets.

TABLE 4
Small and Big Clusters(Textual evidences)

| | Station | Big Clusters | Small Clusters |
|---|---|---|---|
| Hour Slabs | ST-0 | {1}{2,3}{6}{0, 11,15}{4,5,7,8,9, 10,12,13,14,16,17, 18,19,20,21,22,23} | {1}{2,3}{5}{6} {0,11,15}{4,7,8,9, 18,19,21,22,23}{10, 12,13,14,16,17,20} |
| | ST-1 | {0}{1,4,5}{2,3, 6,7,8,9,10,11,12, 13,14,15,16,17,18, 19,20,21,22,23} | {0}{1,4}{5}{6,23} {2,3,7,8,9,10,11, 12,13,14,15,16, 17,18,19,20,21,22} |
| Day Slabs | ST-0 | {Mon,Tue,Wed,Sat} {Thu,Fri}{Sun} | {Mon,Tue,Sat}{Wed} {Thu,Fri}{Sun} |
| | ST-1 | {Tue,Wed,Sat,Sun} {Mon}{Thu,Fri} | {Tue,Wed,Sat,Sun} {Mon}{Thu}{Fri} |
| Week Slabs | ST-0 | {1,2,3}{0,4} | {0}{1,2,3}{4} |
| | ST-1 | {2}{0,1,3,4} | {0,1,3}{2}{4} |
| Month Slabs | ST-0 | {0,1,6,7}{10,11} {2,3,4,5,8,9} | {0,1}{6,7}{10,11} {2,3,4,5,8,9} |
| | ST-1 | {0}{11}{1,2,3, 4,5,6,7,8,9,10} | {0}{11}{1,2,10} {3,4,5,6,7,8,9} |

**Trigger**. Owing to diffusions in propagation networks, the similarity values between temporal splits may change. The *Trigger* can thus estimate under what intervals we must update the similarity grids. This can negatively affect leveraged slabs. Hence, we propose an effective algorithm based on *divide and expansion* to approximate trigger interims.

---

**Algorithm 1** Trigger Interval Estimation

**Input:** $\epsilon$, $\mathbb{D}_e$, $\delta$
**Output:** $\Delta$

1: $\mathbb{H} = \text{Split}(\mathbb{D}_e, 2)$
2: $\tilde{G} = Sub_g(\mathbb{H}[0])$
3: $\kappa^0 = \boldsymbol{E}_v(\tilde{G})$
4: $q_s = \mathbb{H}[0]$
5: **for** each $i$ in Range(1, $|\mathbb{H}[1]|/\delta$) **do**
6:     $q_s = q_s + \text{Subset}(\mathbb{H}[1], i)$
7:     $\tilde{q}_s = Sub_g(q_s)$
8:     $\kappa^i = \boldsymbol{E}_v(\tilde{q}_s)$
9:     **if** $\kappa^0 - \kappa^i >= \epsilon$ **then**
10:         Return $i * \delta$
11:     **end if**
12: **end for**
13: Return $|\mathbb{H}[1]|$

---

As demonstrated in Algorithm 1, we first take the whole or a portion of the dataset (denoted by $\mathbb{D}_e$) and *divide* it into two equal subsets ($\mathbb{H}=\{\mathbb{H}[0], \mathbb{H}[1]\}$). The algorithm is capable to consume a subset of the dataset to accomplish the task quicker. To start, we use the first subset $\mathbb{H}[0]$ to construct the similarity grids and acquire the temporal slabs, based on which we can employ any model $Sub_g$ to exploit the set of subgraphs $\tilde{G}$. The evaluation function $\boldsymbol{E}_v$ can afterward measure the effectiveness $\kappa^0$ using benchmark (Section 5.2). While consuming the same input grids, we then incrementally *expand* the first half $\mathbb{H}[0]$ by the size of $\delta$, which is the portion from the second half $\mathbb{H}[1]$. After each expansion, we reevaluate the effectiveness $\kappa^i$ in the current iteration $i$. In summary, the deficiency rate ($\kappa^0 - \kappa^i$) elucidates how the unchanged similarity grids can escalate

errors when the data size grows. Consequently, when the error size is greater than $\epsilon$, the algorithm will return the final *tolerance threshold* which is the expansion size of the current iteration $i * \delta$. Therefore, when the size of the data grows by the tolerance threshold, it will enforce the rebuilding of the input grids. If the error rate never exceeds the $\epsilon$, the returned value $|\mathbb{H}[1]|$ will indicate that either the initial portion, i.e. $\mathbb{D}_e$, is chosen incorrectly or the similarity changes are negligible.

### 4.1.3 Original Word Embedding (OWE)
When the semantic intersection between $O_u$ and $O_v$ as the respective textual contents of the node pair $(u, v)$ is insignificant, the state-of-the-art text mining models, e.g. [14], cannot perceive intra-nodes correlations. However, the semantic vector space models [17] [25] adopt the dataset-wide knowledge of word co-occurrences to retrieve the vector representation of each word and construct the list $\vec{w}_i^o$ of similar words to $w_i$ in descending order. Accordingly, we can produce a newly formed *encyclopedic semantic representation* $O'_u$ for the each vertex $u$ through replacing any word $w_i \in O_u$ by the top $\zeta$ most similar words from $\vec{w}_i^o$. To this end, we utilize *Original Word Embedding* (OWE) [28], which is a weighted least square regression model (Eq. 3).

$$J_{owe} = \sum_{\forall w_i, w_j \in \mathbb{V}} F(X_{ij}) \times ((\vec{w}_i^m)^T (\vec{w}_j^c) + b_i^m + b_j^c - Log X_{ij})^2$$
(3)

Here, $X$ is the co-occurrence matrix and every entry $X_{ij}$ counts the number of times any word $w_j$ occurs in the context of $w_i$. Also, $\vec{w}_i^m$ and $\vec{w}_j^c$ outline the main ($w_i$ in the center of sliding-window) and context ($w_i$ co-occurs with any other word in the center of window) vectors. Moreover, the cutoff function $F(X_{ij})$ avoids overweighting of the rare or frequent co-occurrence patterns. Finally, each word $w_i$ will be assigned by the median of main $\vec{w}_m^i$ and context $\vec{w}_c^i$ vectors [16], denoted by $\vec{w}_i$.

$$(\vec{w}_i^m)^T (\vec{w}_j^c) + b_i^m + b_j^c = Log X_{ij} \quad (4)$$

While randomly initialized $\vec{w}_i^m$ and $\vec{w}_j^c$ vectors converge toward $Log(X_{ij})$, the bias parameters of $b_i^m$ and $b_j^c$ ensure the symmetry. Our framework utilizes OWE module to infer non-temporal co-occurrence patterns in word embedding.

### 4.1.4 Time-aware Word Embedding (TWE)
As explained in Section 1, the word-word co-occurrence patterns are temporally skewed and change in various latent facets. However, current word embedding approaches [16] [17] [28], including *OWE*, neglect such temporal dynamics. Thus, we devise a novel time-aware embedding module to track co-occurrence alterations and further infer the *multi-aspect temporal-semantic balance* between the nodes. We first despise the biases in OWE [28] and changes Eq. 4 to Eq. 5.

$$(\vec{w}_i^m)^T (\vec{w}_j^c) \propto Log X_{ij} \quad (5)$$

Also, Eq.6 forms the *single slab* time-aware version of Eq. 5.

$$\forall w_i, w_j \in \mathbb{V}_k^l, (\vec{w}_i^{lm})^T (\vec{w}_j^c) \propto \mathcal{F}(w_i, w_j, z_k^l) \times \rho$$
$$\propto Log(X_{k\{i,j\}}^l) \quad (6)$$

From one side, the *slab-based coefficient function* $\mathcal{F}(w_i, w_j, z_k^l)$ corresponds to the inner product of the main and context vectors, and from the other side, it represents the log-based tally for word pair $(w_i, w_j)$ co-occurrence in the same slab $z_k^l$. For instance, as depicted in Fig. 7, $z_2^h$ is the second slab in

6

the hour facet. Nevertheless, for ease of exposition, we will explain the types of coefficient functions later this Section.
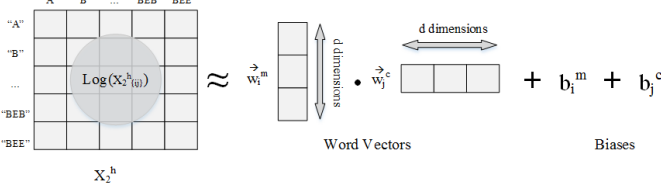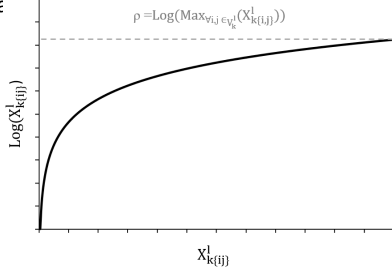


Fig. 7. Time-a



Fig. 8. Log chart

As Fig. 8 shows, the upper bound parameter $\rho$ imposes that both sides of Eq. 7 will get bounded from above ($U_b$). Where the maximum value for the slab-based coefficient is 1, the value for $\rho$ can be affirmed by Eq. 8.

$$U_b(\mathcal{F}(w_i, w_j, z_k^l) \times \rho)^{\uparrow} = U_b(Log(X_{k\{i,j\}}^l))^{\uparrow} \quad (7)$$

$$\rho = Log(Max_{\forall w_i, w_j \in \mathbb{V}_k^l}(X_{k\{i,j\}}^l)) \quad (8)$$

We can then attain Eq. 9 via substituting of Eq. 8 in Eq. 6.

$$\forall w_i, w_j \in \mathbb{V}_k^l, (\vec{w_i^m})^T(\vec{w_j^c}) \propto \mathcal{F}(w_i, w_j, z_k^l) \\ \times Log(Max_{\forall w_i, w_j \in \mathbb{V}_k^l}(X_{k\{i,j\}}^l)) \quad (9)$$

The symmetry in Eq. 9 is preserved via the log-based value of $Max_{\forall w_i, w_j \in \mathbb{V}_k^l}(X_{k\{i,j\}}^l)$ on the right side. Considering the independent appearance for $i$ and $j$, adding $b_i^m$ and $b_j^c$ to Eq. 9 can further maintain the symmetry in the final Eq. 10.

$$\forall w_i, w_j \in \mathbb{V}_k^l, (\vec{w_i^m})^T(\vec{w_j^c}) + b_i^m + b_j^c = \mathcal{F}(w_i, w_j, z_k^l) \\ \times Log(Max_{\forall w_i, w_j \in \mathbb{V}_k^l}(X_{k\{i,j\}}^l)) \quad (10)$$

To continue, we clarify our solution for slab-based coefficient function. Our idea is to invoke two attributes for the slab-based coefficiency (i.e. $\mathcal{F}(w_i, w_j, z_k^l)$) to better infer correlation intensity between each pair of words $(w_i, w_j)$.

- *Contiguity* ($\mathcal{F}_{con}(w_i, w_j, z_k^l)$) explains how extended each pair of words co-occur together in each of the temporal slabs.
- *Depth* ($\mathcal{F}_{dep}(w_i, w_j, z_k^l)$) infers how a pair of words co-occur in each slab when the impacts from parent temporal dimension(s) are inevitably considered.

Subsequently, we put forward a distinctive weighted least square regression model for either of contiguity (Eq. 11) and depth (Eq. 12) attributes of the slab-based coefficient.

$$L_\phi(z_k^l) = \sum_{\forall w_i, w_j \in \mathbb{V}_k^l} f(X_{k\{ij\}}^l)((\vec{w_i^m})^T(\vec{w_j^c}) + b_i^m + b_j^c \\ - \mathcal{F}_{con}(w_i, w_j, z_k^l) \times Log(Max(\forall X_{k\{ij\}}^l)))^2 \quad (11)$$

$$L_\theta(z_k^l) = \sum_{\forall w_i, w_j \in \mathbb{V}_k^l} f(X_{k\{ij\}}^l)((\vec{w_i^m})^T(\vec{w_j^c}) + b_i^m + b_j^c \\ - \mathcal{F}_{dep}(w_i, w_j, z_k^l) \times Log(Max(\forall X_{k\{ij\}}^l)))^2 \quad (12)$$

$$F(x) = \begin{cases} (x/x_{max})^\alpha & if\ x < x_{max} \\ 1 & otherwise. \end{cases} \quad (13)$$

Furthermore, as formalized in Eq. 13, we incorporate the weighting cut-off function $F(X_{k\{ij\}}^l)$ [28] into the regression model and restrict mutual frequencies to the current slab (i.e. slab $k$ in temporal dimension of $l$). The class function precludes overweighting of the exceptional or periodic co-occurrence patterns. Moreover, we apply similar parameter settings of [28] and [17].

Finally, we propose two loss functions of $J_{twe}^c$ and $J_{twe}^c$ that respectively address the problem of time-aware word embedding through contiguity and the depth attributes. As observed in Eq. 14, both regression models iterates through the slabs and latent temporal facets.

$$J_{twe}^c = \sum_{z^l \in Z = \{z^h, z^d, z^w, z^m\}} \sum_{z_k^l \in z^l} L_\phi(z_k^l) \\ J_{twe}^d = \sum_{z^l \in Z = \{z^h, z^d, z^w, z^m\}} \sum_{z_k^l \in z^l} L_\theta(z_k^l) \quad (14)$$

To conclude, we clarify the way we compute each of the twin attributes of the slab-based coefficiency. For the *contiguity* model, we adopt the Jaccard coefficient [18] to assess how the word pairs collectively appear together in various slabs. As Eq. 15 shows, $w_i^{z_k^l}$ counts the number of instances where $w_i$ has been used in slab $k$ of temporal facet $l$, and $w_i^{z^l}$ represents the total number of usages for $w_i$ in dimension $l$, i.e. in any slab of dimension $l$.

$$\mathcal{F}_{con}(w_i, w_j, z_k^l) = \frac{\left| w_i^{z_k^l} \bigcap w_j^{z_k^l} \right|}{\left| w_i^{z^l} \bigcup w_j^{z^l} \right|} \quad (15)$$

For *depth*($\mathcal{F}_{dep}(w_i, w_j, z_k^l)$), we are inspired by the concept of multi-aspect temporal influence [18]. While we include the effects from parent temporal latent facets, we declare (Eq. 16) the depth property as the joint probability for each pair $(w_i, w_j)$ to mutually evolve in the same slab of $k$ in dimension $l$, i.e. $Pr(w_i, w_j, z_k^l)$. Based on subset rule $z^l \subset z^{l+1} \subset z^{l+2} \dots z^{t-1} \subset z^t$, where for instance, we consider the second slab in *hour* latent facet, we will be required to further embrace the impacts from the slabs in the parent latent facets of *day*, *week* and *month*.

$$Pr(w_i, w_j, z_k^l) \propto Pr(w_i)Pr_\omega(w_j|w_i) \\ Pr(z_k^l | z^{\{l+1, l+2, \cdots, t\}}, w_i, w_j) \\ Pr(z^{l+1} | z^{\{l+2, l+3, \cdots, t\}}, w_i, w_j) \cdots \\ Pr(z^{t-1} | z^{\{t\}}, w_i, w_j)Pr(z^t | w_i, w_j) \quad (16)$$

In general, since all the words are treated equally, we assign $Pr(w_i)$ by one. Also, $Pr_\omega(w_j|w_i)$ is the non-temporal weight for $w_j$ to co-occur in the context of $w_i$ that is computed using *Collaborative Filtering*. Moreover, we assume that $l + 1$ is the parent dimension of $l$, e.g. if $l$ is *hour* latent factor, $l + 1$ and $l + 2$ will be respectively the *day* and *week* temporal facets. Hence, the unifacet temporal slab $z_k^l$ will be hierarchically impacted by each of the slabs in parent dimensions, i.e. $z^{l+1}, z^{l+2}, \dots, z^t$. Lastly, the log form of Eq. 16 substitutes the multiplication by summation in both sides of the equation. This can prevent the value in right-hand side to get demoted to less than digital minima. Note, that compilers, e.g. $C\#$ set values less than digital minima to 0.

### 4.1.5 Multifacet Generative Model

In this section, we devise a Multifacet Generative Model (MGM) which elaborately engages numerous temporal

dimensions to infer the correlation weight ($\omega_{mgm}$) between each pair of the nodes in the propagation network. The model initially includes unlimited temporal facets ($z^1, z^2, \cdots, z^{t-1}$). Hence, the proposed model in this section can infer the *time-only correlation* between the node pairs. Note that each temporal aspect, e.g. hour, is a latent factor and based on *multifacet temporal dynamics* [5], the joint probability for a pair of nodes ($n_i, n_j$) to spread a contagion is collectively inferred through how the propagation is occurred in multiple latent temporal facets (Eq. 17). Accordingly, the weight is computed based on the behavior of the nodes in each of multi-facet temporal slabs ($\tau^b$). Moreover, the proposed approach integrates the sequential time feature [10] [11] to further improve the accuracy of the edge weights.

$$Pr(n_i, n_j) \propto \sum_{\sigma^{iterate}_{\{1,2,3,\cdots,t\}} \in \tau^b} Pr(n_i, n_j, z) \tag{17}$$

Intuitively, the temporal subset property [18] ($z^1 \subset z^2 \subset z^3 \cdots z^{t-1} \subset z^t$) justifies that each temporal dimension is affected by its parent latent factor(s). For instance, as verbalized in Eq. 18, given four latent factors of hour $z^h$, day $z^d$, week $z^w$, and month $z^m$, an outbreak in hour dimension may differ in various days, weeks, and months ($z^{\{d,w,m\}}$).

$$\begin{aligned} Pr(z|n_i, n_j) = &Pr(z^h|z^{\{d,w,m\}}, n_i, n_j) \\ &Pr(z^d|z^{\{w,m\}}, n_i, n_j) \\ &Pr(z^w|z^{\{m\}}, n_i, n_j)Pr(z^m|n_i, n_j) \end{aligned} \tag{18}$$

Accordingly, Eq. 19 generalizes Eq. 18 to comprise infinite $t$ temporal dimensions.

$$\begin{aligned} Pr(z|n_i, n_j) = &Pr(z^1|z^{\{2,3,\cdots,t\}}, n_i, n_j) \\ &Pr(z^2|z^{\{3,4,\cdots,t\}}, n_i, n_j) \cdots \\ &Pr(z^{t-1}|z^{\{t\}}, n_i, n_j)Pr(z^{\{t\}}|n_i, n_j) \end{aligned} \tag{19}$$

Basically, the multilateral joint probability for each node pair can be formalized by the probabilistic *chain rule* Eq. 20.

$$Pr(n_i, n_j, z) \propto Pr(n_i)Pr_{Seq}(n_j|n_i)Pr(z|n_i, n_j) \tag{20}$$

The impact of all the nodes in propagation is the same (i.e. $Pr(n_i) = 1$). Furthermore, we integrate the *sequential temporal property* [4], [10] to infer diffusions in each node $n_j$ given another node $n_i$ ($Pr_{Seq}(n_j|n_i)$). Hence, the proposed model in this section considers miscellaneous features including multi-aspect and sequential temporal properties. On the one hand, the computed edge weight counts how the nodes transfer the contagions during each of multi-facet temporal slabs ($Pr(z|n_i, n_j)$), and on the other hand, given the propagation threshold, the model infers how each node may spread a contagion to another ($Pr_{Seq}(n_j|n_i)$). To conclude, we can determine the likelihood for each node pair to bilaterally get infected by the same contagion via replacing Eq. 19 into Eq. 20, which results in the final Eq. 21.

$$\begin{aligned} Pr(n_i, n_j, z) \propto &Pr(n_i)Pr_{Seq}(n_j|n_i)Pr(z^1|z^{\{2,3,\cdots,t\}}, n_i, n_j) \\ &Pr(z^2|z^{\{3,4,\cdots,t\}}, n_i, n_j) \cdots Pr(z^{t-1}|z^{\{t\}}, n_i, n_j)Pr(z^t|n_i, n_j) \end{aligned} \tag{21}$$

Finally, by substituting Eq. 21 into Eq. 17, our recommended model can generally determine the degree of correlation between each pair of nodes ($n_i, n_j$). The log-likelihood from both sides of the Eq. 21 can avoid the outcome multiplication of the possible tiny-valued parameters to fall less than language defined digital minima, which is intrinsically the main cause for illegitimate zero-valued edge weights.

$$\Xi(M) = \sum_{<n_i, n_j> \in <\mathbb{N}, \mathbb{N}>} Log(Pr(n_i, n_j; M)) \tag{22}$$

Furthermore, referring to the industry use-case, the majority of the nodes raise scattered alarms in diverse times, which results in data insufficiency. To address this issue, our MGM model exploits temporal slabs to recompense the inner nodes missing diffusion paths. Moreover, given Eq. 21, let $M$ denote the set of parameters and include $Pr(z^1|z^{\{2,3,\cdots,t\}}, n_i, n_j), Pr(z^2|z^{\{3,4,\cdots,t\}}, n_i, n_j), \cdots,$ $Pr(z^{t-1}|z^{\{t\}}, n_i, n_j)$, and $Pr(z^t|n_i, n_j)$. Correspondingly, we can adopt the generalized Expectation-Maximization(EM) model which is proposed in [18] to maximize the log-likelihood of $\Xi(M)$ (Eq. 22). In this way, we can also infer the best values for each parameter $\mu$ in $M$.

### 4.2 Online Processing

In the online phase, we firstly generate the Temporal-Textual Multi-attribute Graph which comprises diverse edges between each pair of the vertices in dynamical processes. Subsequently, the *coherence score* determines how the trilateral edges of the multi-attribute graph can aggregate to form the latter single-edged *Bilateral Network*. Eventually, we introduce a *Max-Heap Graph cutting* algorithm to retrieve the final subgraphs with highly correlated nodes.

#### 4.2.1 Coherence Score and Bilateral Network

The *Coherence Score* is decided via training in offline phase (Eq. 23) and collectively measures the final correlation weight between each pair of nodes. In this work, we consider three types of coherence: *global semantical coherence* (Section 4.1.3), *multi-aspect temporal-semantical balance* (Section 4.1.4), and *time-only correlation* (Section 4.1.5).

$$\omega_{\mathbf{C}}(u, v) = (1 - \lambda - \beta) \times \omega_{twe} + \lambda \times \omega_{owe} + \beta \times \omega_{mgm} \tag{23}$$

As formulated in Eq. 23, we believe that two vertices in a propagation graph are tightly cohesive if three conditions hold. Firstly, the vector representation of their textual contents are similar, that is measured by Original Word Embedding ($\omega_{owe}$), Secondly, given associated textual contents of the contagions for the pair, the multi-aspect temporal co-occurrence pattern of the words, computed using Time-aware Word Embedding ($\omega_{twe}$), are highly correlated. Thirdly, the two vertices get infected to the same-typed contagions in homogeneous multi-aspect temporal facets, that is assembled through Multifacet Generative Model ($\omega_{mgm}$). Accordingly, we can either opt for a tuning procedure to learn the values for $\lambda$ and $\beta$ or borrow a *page-rank* like model to automatically achieve the final edge weights that can also maximize the effectiveness in subgraph mining. Based on the acquired coherence score, we can consequently combine the three weights in a hybrid manner to retrieve the ultimate undirected Bilateral network. Note that there is only one edge between each pair of vertices in the bilateral network, i.e. no duplicate edges and no loops.

#### 4.2.2 Max-Heap Graph-Cutting

Intuitively, exploiting subgraphs with highly correlated nodes requires *canonical labeling* which is followed by numerous subgraph *isomorphic tests* to assert that no subgraph can be returned twice or more. Nevertheless, the *canonical labeling* algorithm is computationally declared as NP-hard

problem. Furthermore, given multi-attribute graph, mining of the subgraphs with the highest edge weights necessitates to perform enumeration on every possible batch of nodes and decide whether any derived subgraph includes tied in edge weights or not, which further justifies why the problem is NP-hard. Consequently, since subgraph mining is typically regarded as an online continuous procedure and also an underlying phase in various tasks like preventative maintenance or immunization, we need to reduce the complexity of the proposed solution. Accordingly, our proposed framework initially transfers the subgraph mining problem to the computation of edge-weights between the pair of nodes. Subsequently, the proposed *Max-Heap graph-cutting* algorithm (Algorithm 2) can efficiently process the bilateral network $G$ to leverage the final set of tied in subgraphs.

---

**Algorithm 2** Max-Heap Graph Cutting (MaxHGC)

---

**Input:** $G = (\mathbb{N}, \mathbb{E}); W(\mathbb{E}) = \{w(e) | e \in \mathbb{E}\}$
**Output:** $G' = (\mathbb{N}', \mathbb{E}')$
1:  $\mathbb{N}' = \emptyset, \mathbb{E}' = \emptyset, \mathbb{E}'' = \mathbb{E}, \mathbb{N}'' = \mathbb{N}$
2:  $MaxHeap\ H = \emptyset$
3:  **while** $\mathbb{E}'' \neq \emptyset$ **do**
4:      add $e = (u, v) \in \mathbb{E}''$ *into H based on* $w(e)$
5:      remove $e$ from $\mathbb{E}''$
6:  **end while**
7:  **while** $\mathbb{N}'' \neq \emptyset$ **do**
8:      remove $e = (u, v)$ with maximum $w(e)$ from $H$
9:      $Sort(H)$
10:     **if** $u | v \in \mathbb{N}''$ **then**
11:         add $e = (u, v)$ into $\mathbb{E}'$
12:         remove $u, v$ from $\mathbb{N}''$
13:     **end if**
14:     **if** $u, v \notin \mathbb{N}'$ **then**
15:         add $u, v$ into $\mathbb{N}'$
16:     **end if**
17: **end while**
18: return $G'$

---

Algorithm 2 runs as follows: First, given $G = (\mathbb{N}, \mathbb{E})$ as the bilateral network, the probability for every pair $e = (u, v)$ will be proportional to the edge weight between them. Correspondingly, the bigger the weight, the higher the probability for the pair to appear in the same subgraph. As the next step, we duplicate $\mathbb{N}$ and $\mathbb{E}$, as the respective set of nodes and edges, into $\mathbb{N}''$ and $\mathbb{E}''$ to maintain the primary bilateral network $G$ unaffected. After making the primary Max Heap $H$ empty, given the weight function $w(e)$, it adds each edge $e \in \mathbb{E}''$ into $H$ and removes the edge from $\mathbb{E}''$ afterward. Subsequently, it removes the edge $e$ with the largest weight from the heap $H$, Reheapifies $H$, adds the corresponding nodes of $e$ to $\mathbb{N}'$, and removes the nodes from $\mathbb{N}''$. The eliminated edge $e$ from the max-heap is then added to $\mathbb{E}'$. This process is repeated until no nodes exist in $\mathbb{N}''$. Finally, it returns the graph $G'$ with an optimized number of nodes $\mathbb{N}'$ and edges $\mathbb{E}'$, where each group of connected nodes can form an exploited subgraph. In a nutshell, Firstly, the procedure in algorithm 2 ensures that all the edges in $G'$ are among the top $k$ highest weights from the input bilateral graph $G$. Secondly, the resulting graph $G'$ contains all the nodes of $G$, and eventually maintains a minimum number of required edges. Therefore, the proposed max-heap graph-cutting algorithm optimizes the number of exploited

subgraphs and maximizes the intra-subgraph correlations. In other words, it maximizes the correlation inside each of exploited subgraphs, and at the same time, minimizes the number of edges that are utilized in subgraph mining.

# 5 EXPERIMENTS

We conducted extensive experiments on two real-world datasets to examine the performance of our approach in mining of the subgraphs with tightly correlated nodes. We developed our algorithms using C#, Python, and Microsoft Structured Query Language. Aiming to reduce the number of database queries, we utilized the .NET Language-Integrated Queries (LINQ). The experiments were executed on a server with 4.20GHz Intel Core i7-7700K CPU and 64GB of RAM. We publicly share[1] the <u>code</u> and <u>data</u>.

## 5.1 Data

We used two real-life datasets (ST-0 and ST-1) for experiments. The datasets, collected in 2016, report how the faults are propagated between the nodes in each of two Singaporean train stations. The contagions in each of datasets are propagated over more than 1 million nodes, where each node is at most associated with up to 300 contagions.

## 5.2 Benchmark

Intuitively, we define statistical hypothesis parameters to evaluate the effectiveness of our proposed framework in subgraph mining. In summary, we investigate whether all the nodes in each discovered subgraph are correctly chosen, and no missing node is left out. To setup, we dedicate 80% of each dataset for parameter setting and accomplish the test on the remaining 20%. Take $n_i$ as a node in a detected subgraph $g_i$ and presume that the contagion is transmitted if the type of different contagions is the same as $y_i$. For instance, influenza can be transferred by three categories of A, B, and C, but actually, all of them are of the same disease. We can define the statistical parameters as follows:
*True Positive* is observed when $n_i$ is reported to get infected by a contagion $c_i$ of type $y_i$ in time $t$ and at least one of its detected neighbors in the subgraph receives a contagion of the same type $y_i$ during period of $(t - \theta_t, t + \theta_t)$ where the model also estimates that $n_i$ has got infected by $y_i$ typed contagion. *False Positive* can, likewise, represent the number of cases when $n_i$ does not raise the alarm of type $y_i$ and at least one of its neighbors signifies the alarm of the type $y_i$ during the temporal period, but in fact the model asserts that $n_i$ raises the alarm of the type $y_i$. Using similar logics we can also define *True Negative* and *False Negative*. Given four parameters of TP, FP, TN, and FN, we can calculate performance metrics of Precision, Recall, and F-measure. We determine the best model by F-measures.

## 5.3 Effectiveness

In this section, we elucidate the impact of influencing parameters on our proposed framework. We then report how our approach overcomes the performance of other rivals.

### 5.3.1 Effectiveness of Big and Small Clusters

Given the textual contents of the contagions, the Time-aware Word Embedding ($\omega_{twe}$) module of our framework takes the temporal word-word co-occurrence patterns into consideration to infer the correlation weight between each pair of nodes. One prerequisite of time-aware word embedding is
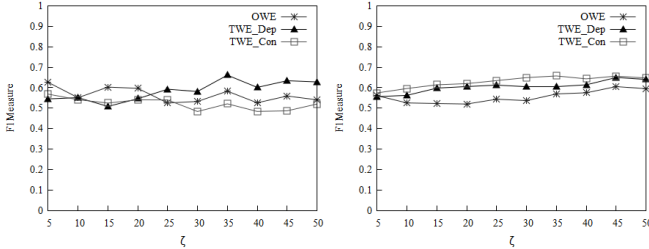
---

to utilize the similarity grids (Sec. 4.1.1) to extract the multi-aspect temporal slabs (Sec. 4.1.2). In a nutshell, clustering of the highly similar splits in $t$ temporal dimensions constructs the slabs and reduces the sparsity in $\mathbb{NTA}$ cube. Nevertheless, the threshold of the agglomerative clustering is quite important as it can integrate mutually exclusive splits into the same slab or assign related splits to separate clusters. As explained in Section 4.1.2, the clustering generates two series of slabs, namely, *small* and *big clusters*. Hence, for each dataset, we train the most suitable time-aware embedding approach (depth or contiguity regression models as verbalized in Section 4.1.4), and compare the performance metric of big and small clusters as defined in Section 5.2.

TABLE 5
Effectiveness of Big and Small Clusters

| $\zeta$ | ST-0 | | ST-1 | |
|---|---|---|---|---|
| | Small Clusters | Big Clusters | Small Clusters | Big Clusters |
| 5 | 0.5850 | **0.5956** | 0.5333 | **0.5420** |
| 10 | **0.5669** | 0.5589 | **0.5564** | 0.5561 |
| 15 | **0.5681** | 0.5431 | 0.5454 | **0.5635** |
| 20 | 0.5381 | **0.5501** | 0.5458 | **0.5732** |
| 25 | 0.5302 | **0.5489** | 0.5575 | **0.5831** |
| 30 | **0.5404** | 0.5342 | 0.5490 | **0.5959** |
| 35 | 0.5448 | **0.5589** | 0.5676 | **0.6048** |
| 40 | 0.5083 | **0.5228** | 0.5643 | **0.6077** |
| 45 | 0.5211 | **0.5682** | 0.5719 | **0.6218** |
| 50 | 0.5146 | **0.5603** | 0.5755 | **0.6174** |
| Comparison | 3 | 7 | 1 | 9 |

Table 5 compares the effectiveness of $\omega_{twe}$ using small and big clusters. Moreover, the experiment illustrates the impact of including different top $\zeta$ items from the vector representation of the words in textual contents of contagions. From Table 5 we can observe that for most $\zeta$ values, the big clusters can better obtain the highly connected subgraphs than the small clusters. Consequently, we opt for big clusters to maximize the effectiveness of the $\omega_{twe}$ module. From another perspective, we observe that the performance metric of the time-aware approach in ST-1 is better than ST-0. The reason is that the ST-1 dataset is denser than ST-0 and includes a bigger number of spreads in various time-stamps. On the other hand, the propagations in ST-1 comprehensively occurred in various temporal slabs of different dimensions.



ST-0 F-measure Changes    ST-1 F-measure Changes
Fig. 9. Effectiveness of embedding models

### 5.3.2 Impact of $\zeta$ on Embedding
Intuitively, either of the slab-based coefficient attributes (*depth* or *contiguity*) can possibly be observed in a dataset. From one perspective, the word-word co-occurrence patterns in a temporal slab may get influenced by the *parent* temporal slab(s) (depth attribute). For instance, the number of trips to the downtown in an hour can differ in various days. Here the hour dimension is affected by its parent

dimension (i.e. day). From another perspective, the occurrence pattern of the word pairs in the slab may neglect the impact from parent temporal aspect (contiguity attribute). The amount of shopping for the essential requirements of the families is approximately the same during the week-days and weekend. Hence, to address the perturbations, as elucidated in Section 4.1.4, we propose two distinctive weighted least square regression models to delineate the contiguity ($TWE\_Con$) and depth ($TWE\_Dep$) attributes. Furthermore, we compare the time-aware regression models with the state-of-the-art original word embedding ($OWE$). We also incorporate the effect of the top $\zeta$ items from word vectors in the performance analysis of subgraph mining. Note that with regard to the parameter setting for OWE, we use the same approach in [28]. Thus, the value of $\alpha$ is assigned by $\frac{3}{4}$ and $x_{max}$ equates to 100. As Fig. 5.3.1 depicts the best performance in ST-0 dataset is achieved by $TWE\_Dep$ at $\zeta = 35$. Similarly, for ST-1 dataset, the best effectiveness in exploiting of the highly correlated subgraphs is achieved by the contiguity-based time-aware approach $TWE\_Con$ at $\zeta = 35$. On the one hand, $TWE\_Con$ overcomes other models in ST-1 dominantly (for all $\zeta$ values), and on the other hand, for ST-0, the $TWE\_Dep$ proves superiority over both OWE and $TWE\_Con$ where the word vectors are represented by greater than 20 items ($> 20$). It is worth noting that the range $[25, 50]$ in both datasets can reasonably demonstrate good performance in vector representation.

### 5.3.3 Impact of Embedding initialization on accuracy
Intuitively, the embedding cost function aims to minimize the error between the co-occurrence of the word pairs (e.g. $(w_i, w_j)$) and the inner product of their corresponding main and context vectors, denoted by $(\vec{w}_i^m)^T$ and $(\vec{w}_j^c)$. However, the word embedding approaches *randomly initialize* the primary vector space for each instance word $w_i$. As explained in Section 4.1.3, the primary random vector is important as it can regulate the final biases, i.e. $b_i^m$ and $b_j^c$.

Surprisingly, the randomly initialized matrix (whose vectors are associated with each of the words) results in a different local maxima every time that the embedding procedure is executed. Therefore, unlike *inheritance algorithms* which perform *mutation* to avoid local maxima, the word-embedding models turn out to be partially dependent on the primary matrix which can negatively affect the performance of the framework and result in a *marginal error*. What is described to put the validity of the *count-based* Neural word embedding models (e.g. GloVe) under strict scrutiny. Since we, likewise, integrate a count-based time-aware embedding module into our framework, we logically need to examine the unsigned size of the marginal error using the easy to interpret *Mean Absolute Percent Error* (*MAPE*), Eq. 24.

$$MAPE = [\frac{1}{n}\sum \frac{|Actual - Forecast|}{Actual}] \times 100 \quad (24)$$

In practice, where we subsequently run both original and time-aware models ($TWE\_Con$ and $TWE\_Dep$) for 10 rounds (denoted by $n$ in Eq. 24), we estimate the *Actual* effectiveness by the *mean* and *median* of all F-measures. Correspondingly, we assign the *Forecast* by the F-measure in each round of execution. Finally, the MAPE is signified by the average of the deviation size between actual and

Forecast metrics. Table 6 presents the results of the experiment on two real datasets. To gain the fairness, for each competitor model, the setting for $\zeta$ that can maximize the subgraph mining effectiveness has been applied.

As evident in Table 6, the average rate of error (MAPE) caused by the random initialization is remarkably inflated. Regardless of the fact that either MAPE rate for our model is less ($TWE\_Con$ in ST-1) or more ($TWE\_Dep$ in ST-0) than the basic baseline (OWE), the outcome effectiveness of our proposed time-aware approaches can still overcome OWE.

TABLE 6
Accuracy Range After Iterative Runs

| Station | Method | $\zeta$ | Forecast value | Accuracy range with MAPE |
|---------|--------|---------|----------------|--------------------------|
| ST-0 | $OWE$ | 5 | Mean | $0.57814 \pm 0.03467$ |
| | | | Median | $0.57529 \pm 0.03415$ |
| | $TWE\_Dep$ | 35 | Mean | $0.65797 \pm 0.05178$ |
| | | | Median | $0.64383 \pm 0.05092$ |
| ST-1 | $OWE$ | 45 | Mean | $0.57248 \pm 0.03932$ |
| | | | Median | $0.58261 \pm 0.03706$ |
| | $TWE\_Con$ | 35 | Mean | $0.58701 \pm 0.02369$ |
| | | | Median | $0.58611 \pm 0.02350$ |

### 5.3.4 Effectiveness of subgraph mining

In general, the problem regarding mining of the subgraphs with highly correlated nodes can be mitigated to the so-called edge weight calculation problem. In other words, the subgraph mining models typically run in two steps: First, they compute the edge weights between the nodes; Second, they extract the subgraphs using a graph-cutting algorithm. We now employ the benchmark (Section 5.2) to examine the performance of seven algorithms in exploiting of the tightly correlated subgraphs:

- *Pair-wise Temporal Model* (PTM, WSDM-2010): the model employs sequential temporal property [4], [10], [11], [12] to infer edge weights.
- *Concurrent Temporal Approach* (CTA, WWW-2017): the generative model [18] infers propagation in two dimensions of the hour and day.
- *Bidirectional Diffusion Inference* (BDI): the bidirectional version of diffusion inference model [10] (KDD-2010) which detects the cascades in quadrilateral dimensions [18] of the hour, day, week and month.
- *Multi-Attribute Time trends to Exploit Subgraphs* (MATES): the unsupervised hybrid model adjusts the sequential and Multi-Aspect temporal parameters to learn the edge weights.
- *Original Word Embedding* (OWE, EMNLP-2014): attains semantic links between nodes via GloVe [28].
- *Time-aware Word Embedding* (TWE): our propositional algorithm that is elucidated in Section 4.1.4.
- *Temporal-Textual Embedding Approach to Leverage Subgraphs* (TEALS): our framework in Section 3.3.

Figure 10 compares the effectiveness of various competitors in subgraph mining. For temporal models, we believe BDI overcomes CTA because firstly it incorporates more latent temporal facets (four dimensions) than CTA (two latent factors of the hour and day) and secondly it specifically uses the cascades to glean the diffusions in both directions. Moreover, PTM gains the least performance as it only employs the sequential temporal property. Recall that the sequential spread of contagions can change in various temporal layers, e.g. the propagation between a pair of nodes may

differ during the weekdays and weekends. Nevertheless, on the contrary, since MATES fuses both sequential and quadrilateral temporal properties, compared to other time-only approaches (PTM, CTA, and BDI), it achieves a better F-measure in subgraph mining. Interestingly, the textual approaches gain a better performance compared to the temporal models. Even the most straightforward original word embedding approach overcomes the best temporal model (MATES). This explains that the correlation between the textual semantics of the nodes deems to be more authoritative than the singular temporal data. However, since TWE improves the OWE with the temporal techniques (i.e. quadrilateral temporal impact), even though the amount of improvement for recall metric is less than Precision, TWE shows its evident superiority over OWE in F-measure.
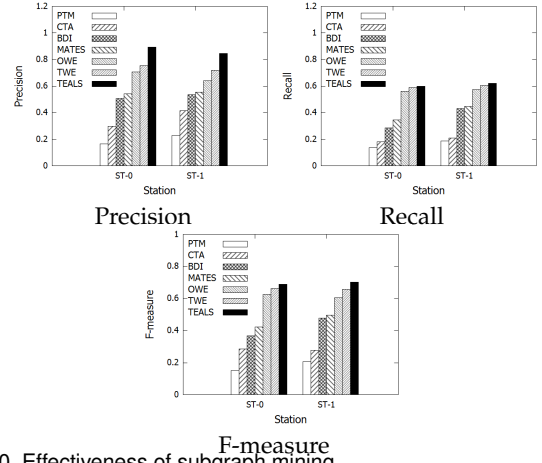


Precision    Recall

F-measure
Fig. 10. Effectiveness of subgraph mining

Note that given each pair of vertices $(u,v)$, our framework (TEALS) (Section 4.2.1) collaboratively integrates three weights of *global semantic coherence* ($\omega_{owe}$), *multi-aspect temporal-semantic balance* ($\omega_{twe}$), and *time-only correlation* ($\omega_{mgm}$) to learn the final edge weight. Hence, aiming to demonstrate the impacts, as reported in Table 7, we train the best values for $\lambda$ and $\beta$ to maximize the effectiveness of the proposed framework. Nevertheless, we fuse the triad weights via a *page rank* like algorithm. Most notably, while the amount of improvement in the recall is less than precision, TEALS gains a substantial improvement in F-measure.

TABLE 7
Mixture Modal Optimized Values

| | F-measure | |
|---|---|---|
| | $\lambda$ | $\beta$ |
| ST-0 | 0.2 | 0.4 |
| ST-1 | 0.1 | 0.6 |

Furthermore, as noticed in both datasets, $\omega_{owe}$ module plays a less significant role in the framework (0.2 in ST-0 and 0.1 in ST-1). This leads to larger influences for time-aware modules ($\omega_{twe}$ and $\omega_{mgm}$). On the other hand, since the sum of coefficients for time-aware modules are respectively 0.8 and 0.9 in ST-0 and ST-1, we can admit that both datasets are time-intensive. From another perspective, the performance metrics of the embedding models (OWE and TWE) are considerably higher than time-only approaches. Hence, given the task of subgraph mining in dynamical processes, the context semantics appear to be more dominant than the initial temporal features.

### 5.3.5 Effectiveness faces data variation

One of the most notable points about our framework is that the changes in similarity grids can initially affect the output temporal slabs and the credibility of subgraph mining process afterward. Since three time-aware models of MGM, TWE_Con, and TWE_Dep use the similarity grids, we need to estimate how frequently the similarity grids may evolve in time. As explained in Section 4.1.2, we divide the data in half and attain both temporal and textual similarity grids and measure the half-size (i.e. 50%) effectiveness in exploiting of the subgraphs with highly correlated nodes. More specifically, where we repeatedly increase the size of the data by $\delta$ (set to 10%) and apply the same similarity grids, we can experimentally study how the effectiveness changes in each interval. This will help us better adjust the Trigger interim which regenerate the similarity grids. Here we aim two goals: Firstly, to minimize the number trigger runs, and Secondly, to maximize the effectiveness of the subgraph mining process. Figure 11 shows how the effectiveness of the time-aware modules are negatively affected where the data size grows and the similarity grids remain untouched.
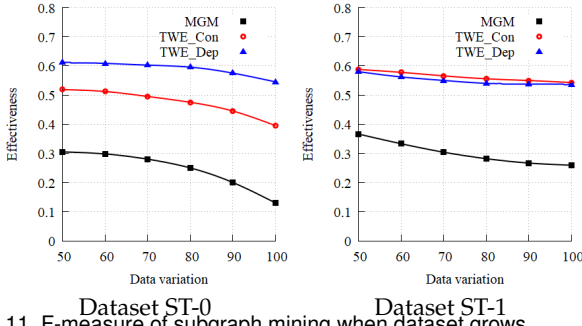


Fig. 11. F-measure of subgraph mining when dataset grows

Due to the bigger size of the corpus which leads to better training results, compared to ST-0, the effectiveness decreases less in dataset ST-1. As depicted, when the dataset expands, the models built on top of textual similarity grids (TWE_Con and TWE_Dep) demonstrate more robustness versus temporal grid approach (MGM). The reason is that while the temporal similarity grids change exquisitely by the sudden temporal skews, a limited number of words are only added to the corresponding co-occurrence matrices in TWE models. Furthermore, TWE models collectively gain the contextual support of the text and time, which help them better tolerate the change spikes. However, while TWE_Con employs the Jaccard function to compute the surface correlation between each pair of nodes, TWE_Dep, employs the Bayes theory through a comprehensive set of miscellaneous parameters, which can justify why the performance of TWE_Dep shrinks less in the course of time. Nevertheless, Figure 11 reflects how we can estimate the value for trigger interval ($\Delta$). For instance, if we set the threshold $\epsilon$ to 3% in ST-0, we will need to reinitialize both temporal and textual similarity grids when the size of the dataset grows by 10%. Similarly, for dataset ST-1, if we set the $\epsilon$ for temporal similarity grid to 4%, we will be required to regenerate the corresponding similarity grids for MDM and TWE (both TWE_Con and TWE_Dep) models after 10% and 30% of growth ($\Delta$) respectively. As a result, the number of trigger runs for temporal similarity grids consumed by the MGM will be three times bigger than the number of calls for the TWE triggers. Finally, we can convert the percentage

metric of $\Delta$ to its corresponding temporal period ($\Delta_t$). This will consequently impose that the trigger must be executed after each $\Delta_t$ interim.

## 5.4 Efficiency

Our framework exploits highly correlated subgraphs from propagation networks that is regarded as an online procedure that fosters applications like preventive medicine and preventative maintenance. Such applications require to continuously process millions of propagation events, which makes the time requirement of our framework eminently critical. Hence, in this section, we study the efficiency of our proposed algorithms.

### 5.4.1 Complexity Comparison

This section compares the time complexity of the original word embedding (OWE) versus our time-aware approach (TWE) from a theoretical computer science perspective. Intuitively, OWE consumes the complete set of vocabularies as a single slab which is denoted by $\mathbb{V}$. Let $\gamma$ be a constant multiplier and $\iota$ as the number of convergence iterations. In retrospect, it is quite obvious that OWE runs in $O(\gamma \times \iota \times |\mathbb{V}|^2)$ where the total number of vocabularies is denoted by $|\mathbb{V}|$. However, TWE divides the data into several smaller set of vocabularies where each of them corresponds to a distinct unifacet temporal slab. Remember that we propose a model to acquire temporal slabs in Section 4.1.2. Let $\mathbb{V}_k^l$ represent the vocabularies in textual contents of the slab $k$ of the latent facet $l$. The expected time for TWE to run each slab will thus yield in $O(\gamma \times \iota \times |\mathbb{V}_k^l|^2)$. Correspondingly, as Formula 25 shows, the time complexity for CPU-based TWE will be collectively calculated through all the slabs. For simplification, we limit the latent space to four dimensions (the value of $\gamma$) where $z^m, z^w, z^d$, and $z^h$ represent the month, week, day, and hour latent facets.

$$\gamma \times \iota \times \sum_{z^l \in Z = \{z^m, z^w, z^d, z^h\}} \sum_{z_k^l \in z^l} |\mathbb{V}_k^l|^2 \qquad (25)$$

Accordingly, due to the subtle connection between the size of the co-occurrence matrix and the time complexity, compared to OWE, TWE comes with two strong assurances in *efficiency*: First, each slab in TWE can be processed in parallel with other slabs that fosters parallelism. Second, compared to the massive size of co-occurrence matrix in OWE, TWE works on a much smaller set of slab-based matrices.

**Scalability.** as an interesting note, all methods are fast when the dataset is small, e.g. $|\mathbb{V}|$ = 10k. Nevertheless, in big data scenario where the number of vocabularies increases, the size of the input co-occurrence matrix will swell. As a result, the performance of OWE will negatively get affected. In contrast, when TWE faces the large-scale data, we observe that compared to the total number of vocabularies, the ratio of the shared vocabularies between temporal slabs mitigates, so-called *slab-vocabulary independence phenomena*. Consequently, the relative size of the slab-based matrices will decrease, which directly reduces the number of entries in slab-based co-occurrence matrices.

### 5.4.2 Impact of $\zeta$ on efficiency

The semantic representation of each node can continuously evolve by arising of the new contagions. Also note that the time-aware word embedding approaches should regenerate the word vectors after each trigger run. Nevertheless, we also need to know how the efficiency of embedding approaches change when each word $w_i$ in textual contents of

sample node $u$ $(O_u)$ is replaced with the top $\zeta$ most similar words from encyclopedic semantic representation of the node $O'_u$. To this end, we combine both datasets (Section 5.1) to study the impact of $\zeta$ on efficiency of word embedding algorithms: $OWE$, $TWE\_Dep$, and $TWE\_Con$.
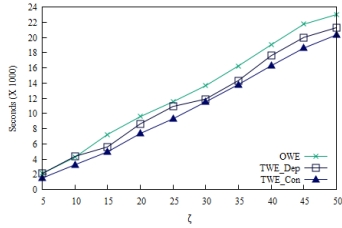


Fig. 12. Impact of $\zeta$ on efficiency of semantic representation

Figure 12 depicts that as the value for $\zeta$ increases, the latency of all three methods slightly grows. The reason is that choosing a bigger value for $\zeta$ increases the size of the semantic representation of each vertex (e.g. $O'_u$) which results in higher processing times. In general, $OWE$ as the original count-based model enumerates how frequently each instance word appears in a context and constructs the co-occurrence matrix afterward. Nevertheless, since the size of the non-temporal co-occurrence matrix in OWE is significantly larger than the slab-based temporal matrices in TWE_Con and TWE_Dep, operations on OWE matrix considerably take bigger times than time-aware approaches. From another perspective, as TWE_Dep utilizes an extra step to infer the optimized values for the Bayesian parameters, it requires more processing time compared to $TWE\_Con$. Nevertheless, the less processing time does not necessarily lead to higher effectiveness, as in terms of effectiveness, TWE_Dep outperforms the TWE_Con in ST-0. **Parallelization**. It is noteworthy that compared to OWE, our time-aware word embedding approaches ($TWE\_Con$ and $TWE\_Dep$) can be parallelized more conveniently. Accordingly, each core can process the comprising data in each distinguished temporal slab, where, in a GPU manner, the amount of final cost for the loss function will be collectively computed in all temporal facets ($z^1, z^2, \ldots, z^t$).

## 6 CONCLUSION

In this work, we devise a unified framework to effectively exploit a set of subgraphs with highly correlated nodes from propagation networks. More specifically, we divide the task of subgraph mining into two subtasks: edge weight computation and graph-cutting procedure. In summary, we firstly endorse a multi-aspect temporal generative model to compute the edge weights between the nodes in correlation networks. Secondly, we develop a neural network based time-aware word embedding algorithm which infers the co-occurrence patterns under an infinite number of temporal dimensions. Finally, we employ our max-heap graph-cutting algorithm to optimize the number of exploited subgraphs. The experimental results show that our proposed framework outperforms existing state-of-the-art approaches in the field of subgraph mining from diffusion networks.

## REFERENCES

[1] Y. Zhang, A. Adiga, A. Vullikanti, and B. A. Prakash, "Controlling propagation at group scale on networks," in *2015 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2015, pp. 619–628.

[2] A. Sepehr and H. Beigy, "Viral cascade probability estimation and maximization in diffusion networks," *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[3] X. Zhang, Y. Su, S. Qu, S. Xie, B. Fang, and P. Yu, "Iad: Interaction-aware diffusion framework in social networks," *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[4] S. Peng, G. Wang, Y. Zhou, C. Wan, C. Wang, and S. Yu, "An immunization framework for social networks through big data based influence modeling," *IEEE Transactions on Dependable and Secure Computing*, 2017.

[5] S. Hosseini, H. Yin, M. Zhang, Y. Elovici, and X. Zhou, "Mining subgraphs from propagation networks through temporal dynamic analysis," in *2018 19th IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 2018, pp. 66–75.

[6] J. Ni, W. Cheng, K. Zhang, D. Song, T. Yan, H. Chen, and X. Zhang, "Ranking causal anomalies by modeling local propagations on networked systems," in *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2017, pp. 1003–1008.

[7] J. Medlock and A. P. Galvani, "Optimizing influenza vaccine distribution," *Science*, vol. 325, no. 5948, pp. 1705–1708, 2009.

[8] C. Chen, H. Tong, B. Prakash, C. Tsourakakis, T. Eliassi-Rad, C. Faloutsos, and D. Chau, "Node immunization on large graphs: Theory and algorithms," *IEEE Transactions on Knowledge & Data Engineering*, no. 1, pp. 1–1, 2016.

[9] E. B. Khalil, B. Dilkina, and L. Song, "Scalable diffusion-aware optimization of network topology," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 1226–1235.

[10] M. Gomez Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 1019–1028.

[11] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "Learning influence probabilities in social networks," in *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010, pp. 241–250.

[12] B. A. Prakash, H. Tong, N. Valler, M. Faloutsos, and C. Faloutsos, "Virus propagation on time-varying networks: Theory and immunization algorithms," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2010, pp. 99–114.

[13] Y. Zhang, A. Ramanathan, A. Vullikanti, L. Pullum, and B. A. Prakash, "Data-driven immunization," in *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 2017, pp. 615–624.

[14] D. Q. Nguyen, R. Billingsley, L. Du, and M. Johnson, "Improving topic models with latent feature word representations," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 299–313, 2015.

[15] S. Hosseini, S. Unankard, X. Zhou, and S. Sadiq, "Location oriented phrase detection in microblogs," in *International Conference on Database Systems for Advanced Applications*. Springer, 2014, pp. 495–509.

[16] S. T. Dumais, "Latent semantic analysis," *Annual review of information science and technology*, vol. 38, no. 1, pp. 188–230, 2004.

[17] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, 2013.

[18] S. Hosseini, H. Yin, X. Zhou, S. Sadiq, M. R. Kangavari, and N.-M. Cheung, "Leveraging multi-aspect time-related influence in location recommendation," *World Wide Web*, pp. 1–28, 2017.

[19] H. M. Abachi, S. Hosseini, M. A. Maskouni, M. Kangavari, and N.-M. Cheung, "Statistical discretization of continuous attributes using kolmogorov-smirnov test," in *Australasian Database Conference*. Springer, 2018, pp. 309–315.

[20] W. Ling, C. Dyer, A. W. Black, and I. Trancoso, "Two/too simple adaptations of word2vec for syntax problems," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 1299–1304.

[21] M. Ebraheem, S. Thirumuruganathan, S. Joty, M. Ouzzani, and N. Tang, "Distributed representations of tuples for entity resolution," *Proceedings of the VLDB Endowment*, vol. 11, no. 11, pp. 1454–1467, 2018.

[22] D. Park, S. Kim, J. Lee, J. Choo, N. Diakopoulos, and N. Elmqvist, "Conceptvector: text visual analytics via interactive lexicon building using word embedding," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 361–370, 2018.

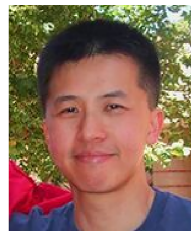[23] S. Hosseini, "Location inference and recommendation in social networks," 2017.

[24] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[25] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, sep 1990.

[26] W. Ling, Y. Tsvetkov, S. Amir, R. Fermandez, C. Dyer, A. W. Black, I. Trancoso, and C.-C. Lin, "Not all contexts are created equal: Better word representations with variable attention," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1367–1372.

[27] E. M. Talley, D. Newman, D. Mimno, B. W. Herr II, H. M. Wallach, G. A. Burns, A. M. Leenders, and A. McCallum, "Database of nih grants using machine-learned categories and graphical clustering," *Nature Methods*, vol. 8, no. 6, p. 443, 2011.

[28] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014.

[29] G. Zhu and C. A. Iglesias, "Exploiting semantic similarity for named entity disambiguation in knowledge graphs," *Expert Systems with Applications*, vol. 101, pp. 8–24, 2018.

[30] M. A. Maskouni, S. Hosseini, H. M. Abachi, M. Kangavari, and X. Zhou, "Auto-ces: An automatic pruning method through clustering ensemble selection," in *Australasian Database Conference*. Springer, 2018, pp. 275–287.

[31] R. Bamler and S. Mandt, "Dynamic word embeddings via skip-gram filtering," *stat*, vol. 1050, p. 27, 2017.

[32] H. Dubossarsky, D. Weinshall, and E. Grossman, "Outta control: Laws of semantic change and inherent biases in word representation models," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 1136–1145.

[33] G. D. Rosin, E. Adar, and K. Radinsky, "Learning word relatedness over time," *arXiv preprint arXiv:1707.08081*, 2017.

[34] B. A. Prakash, A. Beutel, R. Rosenfeld, and C. Faloutsos, "Winner takes all: competing viruses or ideas on fair-play networks," in *Proceedings of the 21st international conference on World Wide Web*. ACM, 2012, pp. 1037–1046.

[35] H. W. Hethcote, "The mathematics of infectious diseases," *SIAM review*, vol. 42, no. 4, pp. 599–653, 2000.

[36] D. Kempe, J. Kleinberg, and v. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003, pp. 137–146.

[37] A. Ganesh, L. Massouli, and D. Towsley, "The effect of network topology on the spread of epidemics," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 2. IEEE, 2005, pp. 1455–1466.

[38] R. Cohen, S. Havlin, and D. Ben-Avraham, "Efficient immunization strategies for computer networks and populations," *Physical review letters*, vol. 91, no. 24, p. 247901, 2003.

[39] S. Saha, A. Adiga, B. A. Prakash, and A. K. S. Vullikanti, "Approximation algorithms for reducing the spectral radius to control epidemic spread," in *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, 2015, pp. 568–576.

[40] S. Hosseini, H. Yin, N.-M. Cheung, K. P. Leng, Y. Elovici, and X. Zhou, "Exploiting reshaping subgraphs from bilateral propagation graphs," in *International Conference on Database Systems for Advanced Applications*. Springer, 2018, pp. 342–351.

[41] R. M. Anderson, R. M. May, and B. Anderson, *Infectious diseases of humans: dynamics and control*. Wiley Online Library, 1992, vol. 28.

[42] J. Yoo, S. Jo, and U. Kang, "Supervised belief propagation: Scalable supervised inference on attributed networks," in *Data Mining (ICDM), 2017 IEEE International Conference on*. IEEE, 2017, pp. 595–604.

[43] B. A. Prakash, L. Adamic, T. Iwashyna, H. Tong, and C. Faloutsos, "Fractional immunization in networks," in *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 2013, pp. 659–667.

[44] E. Shim, "Optimal strategies of social distancing and vaccination against seasonal influenza," *Mathematical biosciences and engineering*, vol. 10, pp. 1615–1634, 2013.

[45] Y. Zhang, A. Adiga, S. Saha, A. Vullikanti, and B. A. Prakash, "Near-optimal algorithms for controlling propagation at group scale on networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 12, pp. 3339–3352, 2016.

[46] Y. Yan, L. J. Chen, and Z. Zhang, "Error-bounded sampling for analytics on big sparse data," *Proceedings of the VLDB Endowment*, vol. 7, no. 13, pp. 1508–1519, 2014.

[47] S. Hosseini and L. T. Li, "Point-of-interest recommendation using temporal orientations of users and locations," in *International Conference on Database Systems for Advanced Applications*. Springer, 2016, pp. 330–347.

**Saeid Hosseini** won the Australian Postgraduate Award and received Ph.D. degree in Computer Science from the University of Queensland, Australia, in 2017. As a post-doc research scientist, his research interests include spatiotemporal database, dynamical processes, data and graph mining, big data analytics, recommendation systems, and machine learning. He has been a PC member in CSS and DASFAA and a reviewer in ICDM and TKDE.



**Saeed NajafiPour** is a researcher in the computational cognitive models laboratory at Iran University of Science and Technology (IUST). He is to receive his M.Sc. in Software Engineering from IUST and completed his B.S. degree in computer software engineering from the Bahonar University, Iran. His research interests include Natural Language Processing, data mining, deep learning, and trajectory analytics.



**Ngai-Man Cheung** received the Ph.D. degree in electrical engineering from the University of Southern California, in 2008. He is currently an Assistant Professor with the Singapore University of Technology and Design (SUTD). His research interests include signal, image, and video processing, computer vision and machine learning.



**Hongzhi Yin** works as a senior lecturer and an ARC DECRA Fellow with The University of Queensland, Australia. He has published 110+ papers and won 5 Best Paper Awards such as ICDE'19 Best Paper Award. He has published in reputed journals and top international conferences including VLDB Journal, ACM TOIS, IEEE TKDE, ACM TKDD.



**Mohammad Reza Kangavari** received Ph.D. in computer science from the University of Manchester in 1994. He is a lecturer in the Computer Engineering Department, Iran University of Science and Technology. His research interests include Intelligent Systems, Human-Computer-Interaction, Cognitive Computing, Data Science, Machine Learning, and Sensor Networks.



**Xiaofang Zhou** is a Professor of Computer Science at The University of Queensland, Leader of UQ Data Science Research Group and an IEEE Fellow. He received B.Sc. and M.Sc. in Computer Science from Nanjing University, China, and Ph.D. from UQ. His research interests include spatiotemporal and multimedia databases, data mining, query processing, big data analytics and machine learning, with over 300 publications in SIGMOD, VLDB, and ACM Transactions.