# EVENT DETECTION VIA TRACKING THE CHANGE IN COMMUNITY STRUCTURE, COMMUNICATION TRENDS, AND GRAPH EMBEDDINGS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

RIZA AKTUNÇ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

NOVEMBER 2022

Approval of the thesis:

**EVENT DETECTION VIA TRACKING THE CHANGE IN COMMUNITY STRUCTURE, COMMUNICATION TRENDS, AND GRAPH EMBEDDINGS**

submitted by **RIZA AKTUNÇ** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Computer Engineering Department, Middle East Technical University** by,

Date: 04.11.2022

## ABSTRACT

## EVENT DETECTION VIA TRACKING THE CHANGE IN COMMUNITY STRUCTURE, COMMUNICATION TRENDS, AND GRAPH EMBEDDINGS

Aktunç, Rıza
Ph.D., Department of Computer Engineering
Supervisor: Prof. Dr. Pınar Karagöz
Co-Supervisor: Prof. Dr. İsmail Hakkı Toroslu

November 2022, 124 pages

Event detection is a popular research problem aiming to detect events from various data sources, such as climate records, traffic data, news texts, social media postings or social interaction patterns. In this work, event detection is studied on social interaction and communication data via tracking changes in *community structure*, *communication trends*, and *graph embeddings*. With this aim, various community structure, communication trend, and graph embedding based event detection methods are proposed. Additionally, a new strategy called *community size range based change tracking* is presented such that the proposed algorithms can focus on communities with different size ranges, and considerable time efficiency can be obtained. The event detection performance of the proposed methods are analyzed using a set of real world and benchmark data sets in comparison to previous solutions in the literature. The experiments show that the proposed methods have higher event detection accuracy than the baseline methods. Additionally their scalability is presented through analysis by using high volume of communication data. Among the proposed methods, CN-NEW, which is a community structure based method, performs the best on the overall. The proposed communication trend based methods perform better mostly on communication data sets (such as CDR), whereas community structure based methods tend to perform better on social media-based data sets. The proposed graph embedding based methods have the potential to produce higher accuracy values with generally low execution times on small communication data sets.

Keywords: Event detection, community detection, temporal network, network features, change tracking, community structure, communication trend, graph embedding, ensemble model

# ÖZ

## TOPLULUK YAPISINDAKİ, İLETİŞİM TRENDLERİNDEKİ VE ÇİZGE TEMSİLLERİNDEKİ DEĞİŞİKLİĞİ İZLEME YOLUYLA OLAY TESPİTİ

Aktunç, Rıza
Doktora, Bilgisayar Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Pınar Karagöz
Ortak Tez Yöneticisi: Prof. Dr. İsmail Hakkı Toroslu

Kasım 2022 , 124 sayfa

Olay tespiti, iklim kayıtları, trafik verileri, haber metinleri, sosyal medya gönderileri veya sosyal etkileşim kalıpları gibi çeşitli veri kaynaklarından olayları tespit etmeyi amaçlayan popüler bir araştırma problemidir. Bu çalışmada, topluluk yapısındaki, iletişim eğilimlerindeki ve çizge temsillerindeki değişiklikleri izleyerek olay tespiti yapılması amaçlanmıştır. Bu amaçla, çeşitli topluluk yapısı, iletişim eğilimi ve çizge temsili tabanlı olay tespit yöntemleri önerilmiştir. Ek olarak, önerilen algoritmaların farklı büyüklük aralıklarına sahip topluluklara odaklanabilmesi ve önemli ölçüde zaman verimliliği elde edilebilmesi için topluluk büyüklük aralığına dayalı değişiklik izleme adı verilen yeni bir strateji sunulmaktadır. Önerilen yöntemlerin olay algılama performansı, literatürdeki önceki çözümlere kıyasla bir dizi gerçek dünya ve kıyaslama veri seti kullanılarak analiz edilmiştir. Deneyler, önerilen yöntemlerin temel yöntemlerden daha yüksek olay algılama doğruluğuna sahip olduğunu göstermektedir. Ayrıca ölçeklenebilirlikleri, yüksek hacimli iletişim verileri kullanılarak analiz edilmiş ve sunulmuştur. Önerilen yöntemler arasında topluluk yapısı temelli bir yöntem olan CN-NEW genel olarak en iyi performansı göstermektedir. Önerilen iletişim trendine dayalı yöntemler, çoğunlukla iletişim veri setlerinde (CDR gibi) daha iyi performans gösterirken, topluluk yapısına dayalı yöntemler sosyal medya tabanlı veri setlerinde daha iyi performans gösterme eğilimindedir. Önerilen çizge temsili tabanlı yöntemler, genellikle düşük yürütme sürelerine ve küçük iletişim veri kümelerinde daha yüksek doğruluk değerleri üretme potansiyeline sahiptir.

Anahtar Kelimeler: Olay algılama, topluluk algılama, zaman dizili ağ, ağ özellikleri, değişiklik izleme, topluluk yapısı, iletişim eğilimi, çizge temsili, karma model

In dedication to my creator, Allah the Almighty, and my great teacher/messenger,
the holy Prophet Muhammad (may Allah bless and grant him)

# ACKNOWLEDGMENTS

In the name of Allah, the most gracious and the most merciful.

All praises to Allah and his blessing for the completion of this thesis. I thank god for all the opportunities, trials and strength that have been showered on me to finish writing the thesis. My humblest gratitude to the holy Prophet Muhammad (peace be upon him) whose way of life has been a continuous guidance for me.

I would like to sincerely thank my supervisors Prof. Dr. Pinar Karagoz and Ismail Hakki Toroslu for their guidance, understanding, and patience. They have been incredible mentors to me in this journey and kept me on the road. It has been a great pleasure and honour to have them as my supervisors.

**TABLE OF CONTENTS**

# LIST OF TABLES

TABLES

## LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| SLM | Smart Local Moving Algorithm |
| dSLM | Dynamic Smart Local Moving Algorithm |
| InfoMap | Map Equation Optimizer Algorithm |
| CDR | Call Detail Records |
| GSM | Global System for Mobile Communications |
| IN | Initial Network: the graph for the initial time window (G0) |
| IILR | Inter-Intra Links Ratio |
| CN | Central Nodes |
| SD | Standard Deviation |
| ENR | Enron Data Set |
| BBM | Boston Bombing Tweet Mention Data Set |
| BBR | Boston Bombing Tweet Retweet Data Set |
| RMS | Reality Mining SMS Data Set |
| RMV | Reality Mining Voice Data Set |
| TGSM | Turkish GSM Operator Data Set |
| BCS | Turkish GSM Operator Data Set: Big Central Square Portion |
| CS | Turkish GSM Operator Data Set: Central Square Portion |

# CHAPTER 1

# INTRODUCTION

*Event detection* has been a trending research topic [2, 1, 3, 4, 5], since it can be useful for a variety of domains such as planning emergency actions for disasters, managing urban transportation effectively [6], and getting the trending and up-to-date news such as financial changes [7]. *Event* refers to a happening that takes place at a certain time and at a certain location, and attracts people's attention. One can find variations of this definition depending on the domain, such as event can happen at a time period instead of a time instance, or place can be multiple or even virtual. But in all definitions, event is considered as an interesting happening and hence affects people's situation and also behavior.

Events can be detected from a variety of resources. One of the most popular data resources is textual contents in the social media and the web [8, 9, 2, 10, 3, 11], such that events trigger social posts and provide a rich resource to detect the events in a much more rapid way than traditional news media. Another valuable resource for event detection is the social interaction and communication structures and their evolution through time. Changes in social communications can indicate *events* since such happenings can trigger interaction among individuals out of daily and ordinary routines. One of the approaches for event detection from communications is based on modeling communication traces as temporal graphs and analyzing the changes through graph features [12, 13, 1, 14].

Recently, event detection studies analyzing the change in the community structure [15, 16] and communication trend [17] on the communication graphs have produced effective solutions. They offer an alternative to feature analysis on communication graphs, and focus on the structural changes on the graph. In [15, 16], event detection problem is modeled as change detection in the community structures extracted from a given communication network. The communities are determined for predefined time intervals, such as daily or weekly communication traces, and the striking changes within the detected communities are marked as event. Community detection is also a popular research problem, and there is already variety of successful methods in the literature, such as SLM [18], dSLM [19], Louvain [20], InfoMap [21], and more advanced models [22, 23, 24], which can be used for community detection based event detection solutions. On the other hand, in [17], an event is modeled as a strong change of communication patterns within the communication network. The communities are detected once at the beginning of the time span and the changes in inter-intra community communications are tracked at each time interval. Moreover, there are also more recent event detection methods are proposed based on graph embeddings. A subset of these methods focus on temporal graph data sets such as streaming climate [4], traffic [25], news [5], and social media [11] data sets. They propose to embed the graphs at the consecutive time steps and then find the anomalous embedding vectors to detect the event time steps. Although all these 3 approaches provide promising results, there are still plenty of possibilities for further improvement and there is a need for a thorough and comparative analysis on a variety of data sets.

In order to fill in this gap, this work proposes new concepts and approaches, as well as improvements over previous community structure, communication trend, and graph embedding based event detection approaches. For all the proposed solutions, a communication graph, which is constructed from communication or social interaction data, and evolution of such graph along the timeline constitute the basis. For each time interval, a communication graph is generated such that nodes represent the users and edges represent the communications or interactions between the users. In the community structure based solutions, at each time interval, communities within the graph are extracted and tracked, whereas in communication trend based solutions, graph of each time interval is considered as a whole and compared against the initial graph constructed at the beginning of the timeline. In the graph embedding based solutions, the graph of each time interval is embedded with various graph embedding methods and then the distances between those embedded vectors are calculated with various vector distance computation methods. In this setting, the novelty and contributions provided by this study are as follows:

- For community structure based event detection, determining the central nodes of the communities and tracking the change in terms of central nodes is shown to be a promising approach in the previous studies [15, 16], however in such earlier studies, node centrality measurement is not performed with well-structured graph centrality metrics. In this study, a new community structure based event detection approach is proposed, such that *central nodes* of a community are determined with PageRank algorithm. By this way, the changes in the central nodes are tracked in terms of several indicators, such as inclusion and drop of central nodes from communities. Changes in such indicators are used for detecting events.

- In community structure based event detection, this study hypothesizes that considering all sizes of communities might prevent detecting events effectively since there are very small (with one or two members) and very large size communities in the daily routine. To overcome this problem, the concept of *community size range based change tracking* is proposed, such that, under the community size range parameters, a method can focus on certain sizes of communities. The experiments show that this mechanism can provide increase in event detection accuracy as well as time efficiency for some of the cases.

- The resolution of the time intervals (i.e., time windows) is a crucial parameter in analysis environment. The effect of time resolution (such as daily, 2-day, 3-day, etc.) on event detection performance for data sets of varying nature has been experimentally analyzed. The experiments reveal that relaxing the time resolution to a certain limit positively affect the accuracy and time efficiency, whereas the event detection performance deteriorates when the resolution is relaxed too much.

- Two novel approaches are proposed for constructing ensemble models. The first one combines the variations of the method that tracks the number of communities with different community size ranges. The second ensemble approach combines the methods tracking the number of communities with the methods tracking the central nodes.

- Various change tracking techniques using the *modularity* of the graphs are proposed, as alternatives to the communication trend based method in [17] that tracks the change in inter-intra communication ratio. Modularity is a function that gives higher values for graphs, as the community structures get denser with sparser interactions between the communities. According to the conducted experiments, in general, the proposed change tracking techniques are more effective.

- A new time step values comparison method is introduced to the communication trend based methods. The standard deviation value is used as the time step values comparison method in the communication trend based method in [17]. In this work, comparing the values of previous and current time steps is also introduced as time step values comparison method. It is based on a hypothesis that the change in consecutive time steps communication trend might indicate an event in the latter time step.

- Besides the changes in community structure and communication trend, the changes in graphs overall structures might indicate an event. To test this hypothesis, a new graph embedding based event detection method is proposed. This method first embeds the structural properties of the graph of each time step with several graph embedding methods and thus produce a vector for each time step. Then, it calculates the distances between time steps vectors and the reference vector, either mean of the vectors or the initial graphs vector, with several vector distance computation methods. Finally, the distances are compared against a threshold value and the time steps whose values are above the threshold value are marked as event time steps.

- A comprehensive set of experiments are conducted comparing the proposed and previous event detection approaches over data sets representing different forms of communications including CDR (Turkish GSM Operator and Reality Mining), network of social media posts (Boston Bombing) and e-mail communication (Enron) data sets. The performance analyses are conducted in terms of accuracy, time efficiency and scalability, aiming to determine which method to prefer under which communication data characteristics. According to the experiments, proposed communication trend based methods tend to perform better than the competitors on CDR based data sets, whereas proposed community structure based methods tend to perform better on social media-based data sets. In general, community structure based methods have lower execution time and better scalability than the communication trend based methods. The graph embedding based methods tend to improve best accuracy results obtained by community structure and communication trend based methods for CDR based data sets.

This thesis is organized as follows. In Chapter 2, related studies are summarized. In Chapter 3, the event detection, community detection, and graph embedding related concepts that are used in this thesis are explained. In Chapter 4, the proposed event detection methods by tracking the change in community structure, communication trend, and graph embedding are described. In Chapter 5, the data sets and the results of the experiments are presented and discussed. Finally, the paper is concluded with an overview in Chapter 6, and all combined community structure and communication trend based event detection experiment results per data set in Appendices.

# CHAPTER 2

# RELATED WORK

In the literature, there is a variety of studies that focus on detecting events from data collections. Some of them take the textual content into account and aim to detect events by analyzing the textual content [8, 9, 2, 10, 3, 26], whereas some others focus on detecting events in temporal graphs and analyze the graph properties [15, 16, 17, 1]. A subset of these studies first detect the communities in temporal graphs and track the community based features to detect events in temporal graphs [15, 16, 17].

Community detection is also a popular and recent research problem [19, 20, 27, 28, 18]. In all these studies, different ways are used to model and extract community structures. They can be grouped under the categories of modularity, compression, significance, and diffusion based community detection methods [29]. In this work, dSLM and InfoMap are used as community detection algorithms. dSLM is a modularity based community detection algorithm, whereas InfoMap is a compression based method.

For temporal graph based event detection, methods presented in [16, 17, 1] can be considered as related to the community structure and communication trend based event detection parts of this study. These methods compute a value for each time interval in temporal graph, and aim to detect the time intervals whose values differs from others with respect to a given threshold. The difference between these studies are basically the graph values considered and tracked at each time interval. Some of them use graph features such as eigen-behavior, node degrees [30, 31, 32, 1], whereas some others use community structure based properties [16], or communication trends such as inter-intra ratio links based on initial community structure [17]. These studies also differ in the way of computing the change and comparison. Some of them consider the change with respect to the previous time interval [16], while the others consider mean and standard deviation for comparison [17].

Rayana et. al. propose an ensemble of five basic graph feature based event detection methods [1]. These five basic methods are eigen-behavior based event detection (EBED), probabilistic time series anomaly detection (PTSAD) [30], Streaming Pattern DIscoveRy in multIple Time-Series (SPIRIT) [32], anomalous subspace based event detection (ASED) [31], and moving-average based event detection (MAED). They make experiments on Reality Mining data sets and provide the average precision values. This study also makes experiments on these data sets, provides average precision values and compares the results based on both event detection and execution time performance.

Moriano et. al. propose that events can be detected by tracking the change in inter and intra community communications in temporal graphs [17]. The authors also introduce the resolution concept which expands granularity of time intervals. The experiments are conducted on Enron and Boston Bombing tweets data sets. The authors provide precision recall curves for analysis on Enron data sets, however analysis on Boston Bombing tweets data sets is presented as a case study. This study also performs experiments on both Enron and Boston Bombing tweets data sets and compares the results in terms of event detection and execution time performance.

Bommakanti et. al. focus on community mining in temporal graphs and track the evolution of communities to detect events in temporal graphs [33]. The authors detect communities at each time step, identify similar communities through the time steps, and track their evolution. In order to detect event time steps, they compare communities against those in the previous time interval. The community changes can be in the form of appear (new community born), disappear (community lost), merge, split, and survive. They use Louvain algorithm which is a modularity based community detection algorithm [20]. Since identifying communities through time steps depends on similarity check, various similarity thresholds are experimented with. The study focuses on the individual community level events and not the global events. Thus, the work does not include results on network level event detection performance. Therefore, this method could not be used as a baseline for comparison.

Zhu et. al. approaches community detection based event detection differently [34]. Instead of detecting communities at each time interval, the authors consider each time interval as nodes of a larger network and detect communities in that large network. First, they extract graph features for the network of each time steps and thus construct a feature vector per time step. Then, they pairwise similarity of feature vectors are calculate to determine edge between them. Then, communities are detected on within in this network, a community label is assigned to each node. When the nodes are sorted in time order with respect to their community labels, if a change is detected in the community label, that time step is considered as change point (event). They experiment their methods on four small data sets and provide precision, recall and f1 measure values. Since their approach are considerably different than our methods and the results are obtained for only small data sets, it is not included as a baseline study in our experiments.

Methods presented in [26, 5, 35, 36, 37] are applying graph embedding methods for event detection on data sets that are not necessarily streaming on time series. They have a set of content and detect the anomalous contents in that set of content. For graph embedding based event detection focused on temporal graphs, methods presented in [4, 25, 38, 39, 40, 11] can be considered as related to the graph embedding based event detection parts of this study since it also focuses on temporal graphs. These methods focus on different kinds of temporal data sets such as climate, traffic, human pose, social network, and cyber security data sets. They make a way to convert the data points into graphs and then embed those data points. After this step, the anomalous data points are detected either by looking at similarities of the embedding vectors or making use of the clustering algorithms.

Li et. al. propose a method that constructs temporal graphs from climate report time series and detects abnormal time steps by using a dynamic graph embedding model [4]. They make experiments on both artificial and real world data sets and present the results in [4]. Similarly, Li et. al. also propose an anomaly detection method that is based on dynamic graph embedding that is applied on traffic data [25]. The graph data points in time instances of the given temporal traffic data graph is embedded with a dynamic graph embedding model based on the graph similarity. Then, the embedded vectors are clustered to find out the anomalous time instances which represent the traffic incidents. They make experiments on traffic data produced by the Simulation of Urban Mobility framework and present the results in [25].

Markovitz et. al. focus on human pose graphs in video sequences and proposes a method to detect abnormal human pose actions in a given video sequence [38]. They embed the human pose graphs and cluster them. This enables them to mark each human pose action as normal or not based on the clustering results. They evaluate the proposed method on two anomaly detection data sets and present the results in [38]. Genc et. al. constructs embedding from the user, date, and social content data collected from social networks. Then, they propose a graph embedding based event detection method that detects anomalous embedding values from these constructed embedding values [11].

Abd et. al. proposes a method to detect anomalies in a stream of heterogeneous graphs in real time. They also propose a graph embedding method for graph streams and a incremental similarity measure based on graph edit distance that are used in the steps of the proposed event detection method [39]. Kosan et. al. also focus on detecting the events in dynamic graphs in real time [40]. To this aim, they propose a graph embedding based deep learning model for event detection on dynamic graphs. Their approach combine temporal and structural properties and aim to detect both the structural and temporal aspects of events.

This study contributes to temporal graph based event detection in terms of community structure, communication trend, and graph embedding based change detection. In community structure based approach, the bucket and resolution concepts are proposed. Additionally, central node based change tracking approach is revised and modified by using PageRank for determining central nodes. In communication trend based approaches, using modularity besides IILR as communication trend metric is proposed. Additionally, change computation method is modified such that the graph value is compared against the previous time interval. The data set variety is also extended including one small, and one large social media data set, and one small, and one large CDR data sets. This enables us to have observations on which methods are more effective on which kind of data sets. In graph embedding based approaches, it is hypothesized that the distance between two consecutive graph instances in a temporal graph data set indicate an event in the latter graph instances time point. Based on this hypothesis, a graph embedding based event detection method that involves various graph embedding models and various vector distance computation methods is proposed. It is experimented on all the data sets in this study and results are compared with the community structure and communication trend based event detection methods.

# CHAPTER 3

# PRELIMINARIES

## 3.1 Basic Definitions

In this section, the basic concepts that are used in order to introduce the problem and the proposed methods are defined.

**Temporal Graphs:** The graphs that change over time are defined as temporal graphs. In our context, both nodes and edges can change (added or deleted) over discrete sequential time steps. Thus, temporal graph can be expressed as a sequence of graphs such as $< G_1, G_2, ..., G_t >$.

**Resolution:** The time window length used to construct the temporal graphs is defined as the resolution. The resolution can be chosen according to the data set and the requirements of the event detection problem. For instance, if the resolution is set as 1 day, then each graph instance in the temporal graph corresponds to communications per day.

**Community:** Although there is no universally accepted definition, in general, community in a graph is defined as a set of nodes that have links with each other more than they have with the other nodes of the network [41, 27, 28].

**Modularity:** Modularity is a measure about the community structure of a graph. In order to compute the modularity value of a network, firstly, a null model of that network should be generated. A null model of a network can be constructed by removing the existing edges, and replacing them with the new ones randomly while maintaining the original degrees of the nodes. For a given network, modularity is defined as the fraction of the difference between the total number of internal edges in the real network and the total number of internal edges in its null model, to the total number of the edges in the whole network. It is calculated as given in Equation 3.1. The measure is motivated from the idea that a randomly created network is not expected to have a good community structure. Thus, as the real network differs from its null model, its community structure is considered to be stronger [41, 42].

$$Q = \frac{1}{2e} \sum_{ij} (O_{ij} - N_{ij}) \delta(C_i, C_j) \tag{3.1}$$

In the equation, $O_{ij}$ is the number of edges between node $i$ and node $j$ in the original network and $N_{ij}$ is the number of edges between node $i$ and node $j$ in the null model. The $\delta$ function ensures taking only internal edges into account in the summation by generating $1$ if node $i$ and node $j$ are in the same community ($C_i = C_j$), and it returns $0$ otherwise. The summation iterates over all pairs of nodes of the network. Finally, $e$ represents the total number of edges of the network. Note that, the number of nodes and the number of edges are the same for both the original network and its null model, since null model preserves the nodes and their degrees.

**Initial Network (IN):** Given a sequence of time-stamped communications, *initial network* is the graph constructed for the initial time window ($G_1$) with respect to the given resolution.

**Inter-Intra Links Ratio (IILR):** For a given set of communities, inter-intra link ratio is defined as the difference between the number of inter community links and the number of intra community links to the total number of links in the full network. The calculation of the number of inter community links, the number of intra community links and IILR are as given in Equations 3.2, 3.3, and 3.4, respectively. In the equations, $O_{ij}$, $C_i$, $C_j$, and $\delta$ mean the same as in Equation 3.1. The $\omega$ function returns $0$ if node $i$ and node $j$ are in the same community ($C_i = C_j$), and, $1$ otherwise. This definition is used in [17] to detect events that trigger information flow changes across communities.

$$G_{inter} = \sum_{ij} O_{ij} \omega(C_i, C_j) \tag{3.2}$$

$$G_{intra} = \sum_{ij} O_{ij}\delta(C_i, C_j) \tag{3.3}$$

$$IILR = \frac{G_{inter} - G_{intra}}{G_{inter} + G_{intra}} \tag{3.4}$$

**Central Nodes (CN):** Central nodes of a community are the nodes whose centrality scores are higher than the average centrality score within the network. In our study, PageRank [43] is used as the centrality metric.

**Community Size Range (Bucket):** In determining the change in the community structures, taking all of the communities into account may lead to incorrect predictions since events may only affect communities of certain size. Additionally, for large data sets, using only the communities of certain sizes may positively affect execution time considerably compared to using all communities for change detection. With this motivation, the following groups (buckets) of communities are defined with respect to their sizes:

- Bucket A (all) includes all communities without any filtering.
- Bucket F (filtered) excludes the very small communities (of size one and two nodes) and very large ones and includes the rest of the communities. The filtered large communities contain less than 0.001 % of the total nodes.
- Bucket Q1 includes the first quarter of the filtered communities (Bucket F) (initial 25% partition) with respect to community distribution with respect to size.
- Bucket Q2 includes the second quarter of the filtered communities (the second 25% partition).
- Bucket Q3 includes the third quarter of the filtered communities (the third 25% partition).
- Bucket Q4 includes the last quarter of the filtered communities (the fourth 25% partition).

- Bucket H1 includes the first half of the filtered communities (initial 50% partition).
- Bucket H2 includes the second half of the filtered communities (the second 50% partition).

Note that H1 = Q1 ∪ Q2, H2 = Q3 ∪ Q4, and F = H1 ∪ H2. When change detection is applied for a given bucket, only the nodes of the communities in the bucket are considered.

**Event:** This study considers that an event is expressed with a signal (reflection) in its corresponding time interval.

**Event Detection:** It is aimed to determine the *time intervals that include event*. Hence, it is hypothesized that the change in the community structure and community communication trend within communication network provide signals (reflections) of events. Different ways to determine the change in community structure and community communication trend are elaborated.

## 3.2 Community Detection

In the literature, there are several community detection algorithms [19, 27, 21]. In this work, dSLM [19] and InfoMap [21] are used as the community detection methods.

### 3.2.1 dSLM: Dynamic Modularity Optimizer

Modularity is a metric generally used for evaluating the quality of community detection algorithms. It is a function that outputs higher values if a given network can be partitioned better as communities. In some of the community detection algorithms it is also used as the objective to be maximized. dSLM [19] is the dynamic and faster version of SLM algorithm [18] which aims to determine communities by optimizing the modularity values of networks. It optimizes modularity by applying the following steps [18]:

1. Assign each node as a singleton community (i.e., each node is a community itself).

2. For each node and its neighbours, in random order, move the node from its own community to its neighbor's community and recalculate the modularity value. If it is increased, keep the change, else, revert the movement. This step is called as local moving heuristic since it heuristically tries to improve the modularity value by moving the nodes among communities.

3. After step 2 is completed, mark each community as as a sub-network and apply step 1 and 2 to each one of the sub-networks. At the end of this step, a set of sub-networks, which have its own set of communities are generated.

4. Then, this structure is converted into a new (reduced) graph where each community is a node, and, each sub-network is a community.

5. Apply steps 2, 3, and 4 to this reduced network recursively until the network that cannot be reduced further.

### 3.2.2 InfoMap: Map Equation Optimizer

The InfoMap algorithm detects the communities by minimizing the map equation value of the given network. InfoMap detects community structure by minimizing the description length in the map equation. Since it is infeasible to check all possible partitionings and choose the one that minimizes the map equation value, it uses heuristic search to find the near optimum structure [21].

The map equation formula that InfoMap minimizes is given in Equation 3.5.

$$L(M) = q_{\curvearrowright} H(\theta) + \sum_{i=1}^{m} p_{\circlearrowright}^{i} H(\alpha^{i}) \tag{3.5}$$

In the equation, M is a proposed community structure of the network such that each node belongs to a community. L(M) is the average length to describe an infinite random walk on the network whose community structure is M.

The map equation calculates the minimum description length of a random walk on the network based on the partition M. The first part of the equation gives the average number of bits necessary to describe movement between communities, and the second part determines the average number of bits necessary to describe movement within communities [21].

## 3.3 Graph Embedding

There are two main ways to represent a graph as either set of vectors or only one vector. The first one is to get an embedding vector of each node in the graph, thus get a matrix for the inputted graph by combining each node's embedding vector. The second one gets only one vector for the inputted graph by embedding whole graph into one vector.

There are algorithms such as DeepWalk [44], node2vec [45], struc2vec [46] that embed the graph and output a matrix where each vector represents a node in the graph.

DeepWalk is based on skip-gram model of Word2vec [47] algorithm which aims to convert words into vectors. It hypothesizes that similar nodes should have similar embedding vectors such that similar words should have similar embedding vectors. It uses random walks that start from selected nodes then move to random neighbours for a certain number of times to sample the graph. These random walks are used as training input to the skip-gram model. Then, the hidden layer of the skip-gram model is outputted as the embedding vector for each node in the graph.

Node2vec is also based on skip-gram model of Word2vec [47] algorithm. It differs from DeepWalk by doing the random walks in a biased way whereas DeepWalk does unbiased, completely random walks. By this way, it tries to produce embedding vector that better preserves the local neighbourhood of the input node.

Struc2vec includes the position of the node in the network structure as an information during the node embedding process. This way, it can capture the nodes structural position information in their embedding vectors. This helps for the tasks where the nodes structural position matters. Node2vec and DeepWalk algorithms tend to fail such tasks that require nodes structural position information since they do not encode this information into nodes embedding vectors.

The algorithms such as graph2vec [48], GL2vec [49], FEATHER [50], LDP [51] embed the given graph as a whole and produce one vector.

Graph2vec is based on a neural embedding model as inspired by skip-gram model of doc2vec [52]. It learns the embedding vectors in an unsupervised way. Thus, it could be used for graph classification, clustering, and aiding event detection on temporal graphs. It samples and labels all sub graphs in the given graph. These sub graphs are then fed to the skip-gram model as training input. Finally, the result of the hidden layer of skip-gram model is produced as the embedding vector for the whole graph.

GL2vec is proposed as an updated version of graph2vec algorithm. [49] claims that graph2vec has two limitations such that it cannot handle edge labels and does not preserve enough structural information to compute structural similarity. It aims to overcome these limitations of graph2vec algorithm by constructing a line graph and attaching its embedding vector to the embedding vector of the original graph. So, basically it uses the graph2vec algorithm on both original graph and its generated line graph version and then combine their embedding vectors. A line graph L(G) [53] for a graph G is constructed as follows: create a node in L(G) for each edge in G, then create an edge between two nodes in L(G) if their corresponding edges in G has a node in common.

FEATHER is a recent graph embedding algorithm that uses characteristic functions of node features with random walk weights to describe node neighborhoods. Combinations of these node embedding values result in aggregated graph descriptors such that an embedding vector for the whole graph. [50]

LDP is another recent graph embedding algorithm that computes the histograms of degree profiles, then concatenates them to form the embedding vector for the whole graph. It is designed for non-attributed graphs and thus has low performance on tasks that focus on attributed graphs. On the other hand, it is very efficient on execution time aspect by having linear computation time. [51]

In this study, the approach that produce a vector for a graph is used since comparing vectors is easier than comparing matrices of a set of given graphs. Besides, the order of the embedding vector representations of the nodes cannot be preserved in the matrix representations of the consecutive graphs. Moreover, each time steps graph may have different number of nodes and this may lead to having different size of matrices to compare. This is also another obstacle in using the first approach that produce a matrix for a graph for the proposed graph embedding based event detection methods. So, given all these reasons, the second graph embedding approach that produce a vector for a graph is used for the proposed graph embedding based event detection methods. In this category, all graph2vec [48], GL2vec [49], FEATHER [50], LDP [51] methods are used and experimented.

## 3.4 Distance Metrics on Vectors

In this work, during the event detection based on graph embedding method's embedding vector distance calculation phase, the Bray-Curtis, Canberra, Chebyshev, City-block (Manhattan), Correlation, Cosine, Euclidian, and Minkowski distance methods are used. All distance methods take 2 vectors denoted as $u$ and $v$ as required parameters. They also take a weight vector that is used as a weight for each value in $u$ and $v$ as optional parameter. When, it is not provided, each value has equal 1.0 weight.

The Bray-Curtis distance [54] is in the range [0, 1] if all values in the input vectors are positive which is the case for the experiments in this study. It is defined in Equation 3.6.

$$D_{Bray-Curtis}(u,v) = \frac{\sum_i |u_i - v_i|}{\sum_i |u_i + v_i|} \tag{3.6}$$

The Canberra distance is defined in Equation 3.7.

$$D_{Canberra}(u,v) = \sum_i \frac{|u_i - v_i|}{|u_i| + |v_i|} \tag{3.7}$$

The Chebyshev distance is defined in Equation 3.8.

$$D_{Chebyshev}(u,v) = \max_i |u_i - v_i| \tag{3.8}$$

The Cityblock (Manhattan) distance is defined in Equation 3.9.

$$D_{Cityblock}(u,v) = \sum_i |u_i - v_i| \tag{3.9}$$

The Correlation distance is defined in Equation 3.10 where $\overline{u}$ is the mean of the elements of $u$ and $u.v$ is the dot product of $u$ and $v$.

$$D_{Correlation}(u,v) = 1 - \frac{(u-\overline{u}).(v-\overline{v})}{\|(u-\overline{u})\|_2 \, \|(v-\overline{v})\|_2} \tag{3.10}$$

The Cosine distance is defined in Equation 3.11 where $u.v$ is the dot product of $u$ and $v$.

$$D_{Cosine}(u,v) = 1 - \frac{u.v}{\|u\|_2 \, \|v\|_2} \tag{3.11}$$

The Euclidean distance is defined in Equation 3.12.

$$D_{Euclidean}(u,v) = (\sum_i |u_i - v_i|^2)^{1/2} \tag{3.12}$$

The Minkowski distance is defined in Equation 3.13. When parameter $p$ is set as 2 which is the default, it is same as the Euclidian distance. With the $p$ parameter, it is more flexible than the Euclidian distance.

$$D_{Minkowski}(u,v) = (\sum_i |u_i - v_i|^p)^{1/p} \tag{3.13}$$

# CHAPTER 4

# PROPOSED EVENT DETECTION METHODS

This study considers that a significant change in the community structure or the communication trend between the graphs of consecutive time steps indicates an event. To this aim, several basic and ensemble methods are proposed to track the changes in the graphs and detect events. The overview of the proposed approach's architecture is presented in Figure 4.1. As given in the figure, for the proposed methods, the event detection pipeline is composed of three basic steps: Pre-processing, community detection and applying a set of change tracking methods.

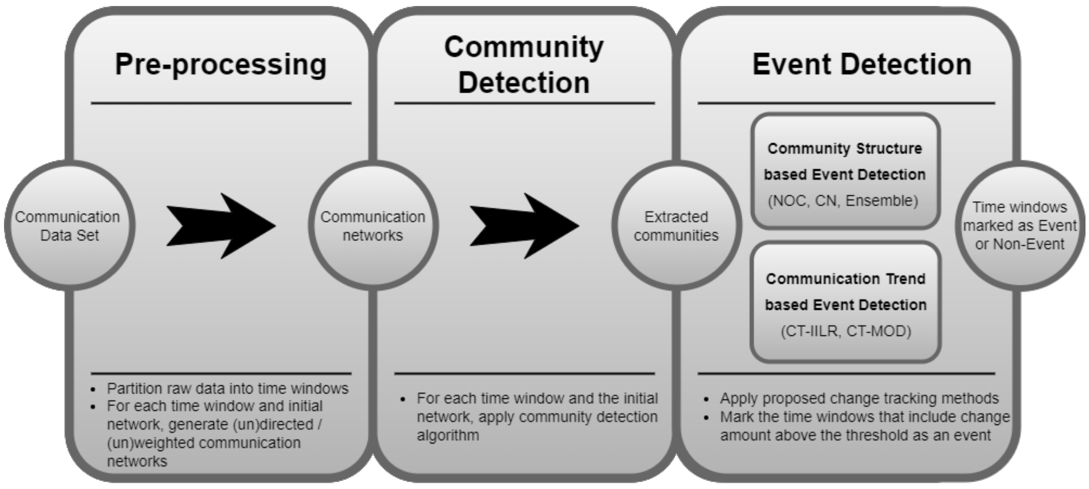
Pre-processing
Community Detection
Event Detection
Communication Data Set
Communication networks
Extracted communities
Community Structure based Event Detection (NOC, CN, Ensemble)
Communication Trend based Event Detection (CT-IILR, CT-MOD)
Time windows marked as Event or Non-Event
Partition raw data into time windows
For each time window and initial network, generate (un)directed / (un)weighted communication networks
For each time window and the initial network, apply community detection algorithm
Apply proposed change tracking methods
Mark the time windows that include change amount above the threshold as an event

Figure 4.1: The Overview of the Method

## 4.1 Overall Architecture and Pipeline

### 4.1.1 Pre-processing & Graph Generation

For the proposed methods, the communication data is represented as a graph. To this aim, in order to analyze the effect of graph type, directed, undirected, weighted, unweighted graphs are constructed from raw data sets. Since the nature of raw data sets differ from each other, different processes must be applied to form these graphs. For example CDR data set is a collection of call records between two users containing their IDs, the time of the call, and additional other information in each data instance. For each time interval, separate graphs are constructed by considering the records having timestamps in the interval. By including the number of calls, or the duration of the calls, weighted graph can be formed. By distinguishing the caller and callee, directed graph can be generated.

### 4.1.2 Community Detection

For the community structure and communication trend based event detection methods, from time intervals of the data sets, *directed weighted*, *directed unweighted*, and *undirected weighted* graphs are generated as described above. The dSLM algorithm [19] is used as the community detection technique for all data sets. The InfoMap community detection method is also used in order to compare our proposed methods with the method proposed in [17].

### 4.1.3 Event Detection

Basically, there are 3 different kinds of change structures tracked in our method.

In the first one, changes in the community structure are tracked. Two basic community structure features, *Number of Communities* and *Central Nodes* are considered for tracking. Besides tracking these two features, ensembles of these methods are developed as well. The first type of ensembles is the combination of change tracking on different buckets (community size ranges). The second type is ensemble of the change tracking of the central nodes and the change tracking of the number of communities methods.

As the second main tracking method, the changes in the communication trends among the communities are tracked. Here, communities are detected once and the changes in community interactions are examined. The modularity and inter-intra link ratio [17] are used in order to measure the change in communication trends.

The third main tracking objective is the graph embeddings of the time steps in the temporal graphs. Each time steps graph is embedded with several graph embedding methods and this way a vector is produced for each time step. Then, the distance between the vector of each time step and the reference vector is computed. The time steps whose distances are more than a threshold value are marked as event time steps.

The details of pre-processing and proposed event detection methods are described in the rest of this section.

## 4.2 Pre-processing & Graph Generation

Since the nature of the raw data sets differs from each other, pre-processing step also includes different sub-tasks for different data sets. The details for each of the data sets used in the study are as follows:

1. **Enron** This well-known data set is a collection of email records of the Enron company collected between 1999-01-01 and 2002-04-30. It is structured as a network such that nodes represent the employees and edges represent the emails sent/received between employees. The number of exchanged emails is used as the weight of the edge. Therefore, a directed weighted network is obtained. For this data set, time interval granularity set as one week.
2. **Boston Bombing Tweets** This data set contains the networks of retweets and mentions on bombing event happened during the Boston marathon in April 2013. The data set is partitioned daily, thus 30 networks for each of the retweets and mentions collection are generated. The generated networks are directed and weighted where weights are the mention and retweet counts, respectively.

3. **Reality Mining** This data set is a Call Detail Record (CDR) collection including date of the call, caller id and call receiver id. The time interval is set as a week, since the ground truth is provided over weeks. In the constructed graphs, nodes represent IDs of the participants, and the edges represent the calls between them. The weight of an edge represents the number of calls made between the participants. The data set contains both SMS and Voice Call collections. For both of them, directed weighted, directed unweighted and undirected weighted networks are generated, resulting with 6 networks in total.

4. **Turkish GSM Operator** This data set is also a CDR data, including the records of GPRS, SMS and voice calls made in a large metropolitan city in Turkey in September 2012. The voice call records are used in this study. The data set is partitioned into days, and a directed weighted graph is constructed for each partition. In the constructed graphs, nodes represent the the users, the edges represent the calls. The weight of an edge represents the number of calls made between the users.

## 4.3 Event Detection Methods by Tracking the Change in the Community Structures

In order to track the change in community structure between two consecutive time intervals, the following two indicators are considered: change in the number of communities and change in the central nodes of communities.

### 4.3.1 Tracking the Change in the Number of Communities

The focus of this method is determining the change in the number of communities generated for consecutive time windows. The algorithm for the method is given in Algorithm 1. It takes set of communities, bucket, and event detection threshold values as parameters. For each time window, the algorithm finds the number of communities that are in the range of given bucket. Then, the change ratio in the number of communities is calculated by taking the absolute difference between the number of communities in the previous time step and the current time step and dividing the result by the number of communities in the previous time step. If a time window's change ratio is greater than or equal to the given event detection threshold value, that time window is marked to include an event.

**Algorithm 1** Event Detection via Change on # of Comm

```
Require: setOfComm, bucket, threshold
Ensure: events
  prev ← NOC(setOfComm($t_1$), bucket)
  for i ← 2 to timeWindowCount do
    cur ← NOC(setOfComm($t_i$), bucket)
    change ← abs(cur − prev)/prev
    if change $\geq$ threshold then
      add $i$ to events
    end if
    prev ← cur
  end for
  return events
```

### 4.3.2 Tracking the Change in the Central Nodes

In this method, the focus is on the central nodes of the communities within the communication network and their change across time windows. The algorithm for this method is presented in Algorithm 2. Similar to the previous one, it takes set of communities, change detection method, bucket, and event detection threshold values as input parameters. For each time window, the algorithm finds the number of communities that are in the range of given bucket; furthermore, the central nodes of the communities are also determined. Based on the given change detection method, the algorithm can find change ratios in four different ways:

1. **SIZE:** If the given change detection method is based on Size, the change ratio is calculated according to the difference between the number of central nodes in the previous time window and the number of central nodes in the current time window.

2. **NOT_ANY_MORE:** If the given change detection method is Not Any More, the change ratio is calculated by computing the number of central nodes of previous time step that are not any more central nodes in the current time step.

3. **NEW:** If the given change detection method is New, the change ratio is calculated by computing the number of newly introduced central nodes in the current time window.

4. **NOT_ANY_MORE_NEW:** If the given change detection method is Not Any More and New, the change ratio is calculated by computing the number of central nodes of previous time step that are not anymore central nodes in the current time step added up with the number of newly introduced central nodes in the current time window.

As in the previous algorithm, time window is marked to include an event, if a time window's change ratio is greater than or equal to the given event detection threshold value.

**Algorithm 2** Event Detection via Change On Central Nodes

```
Require: setOfComm, changeMethod, bucket, threshold
Ensure: events
  prev ← setOfComm($t_1$).centralNodes
  for i=2 to timeWindowCount do
    cur ← setOfComm($t_i$).centralNodes
    change ← comp(changeMethod, prev, cur)
    if change $\geq$ threshold then
      add $i$ to events
    end if
    prev ← cur
  end for
  return events
```

### 4.3.3 Tracking the Change in the Community Members

We consider the change in the community members as another candidate for event indicator. However, community detection techniques do not assign global identifiers to communities to be tracked over time windows. To be able to track the change within the same community, we assume that two communities in consecutive windows are the same if they have common central nodes. Therefore, we firstly find the communities of consecutive windows around similar central nodes. Then, we compute the change in the number of members within the community. We compute the change in three different ways: the *minimum (MIN)* of change magnitude, the *average (AVG)* of the change magnitude, and the *maximum (MAX)* change magnitude. The algorithm of the method is given in Algorithm 3 and Algorithm 4.

**Algorithm 3** Event Detection via Change on Comm Members

```
Require: setOfComm, changeCompMethod, changeThreshold
Ensure: events
  prev = setOfComm($t_1$).members
  for i=2 to timeWindowCount do
    cur = setOfComm($t_i$).members
    change = ComputeChangeOnMembers(
    changeCompMethod, prev, cur)
    if change $\geq$ changeThreshold then
      add $i$ to events
    end if
    prev = cur
  end for
  return events
```

**Algorithm 4** Compute Change on Community Members

```
Require: changeCompMethod, prev, cur
Ensure: change
  if changeCompMethod is MIN then
    return min of the change values on the comm members
  else if changeCompMethod is AVERAGE then
    return avg of the change values on the comm members
  else if changeCompMethod is MAX then
    return max of the change values on the comm members
  end if
```

### 4.3.4 Early Ensemble Strategies

#### 4.3.4.1 Combining the Variations of the Change Tracking in the Number of Communities

In the first ensemble method, we hypothesize that events can be detected better if the results of the variations of the number of communities based event detection method are merged. In the basic method (given in Section 4.3.1), we hypothesize that events might arise from communities of *certain size*. Our motivation behind this hybrid approach is to extend the initial hypothesis to cover the cases that an event might arise from set of communities of certain *several* sizes. For instance an event may involve communities that contain 3-5 and 11-20 people. The initial basic method would not be able to detect such kind of events, whereas this ensemble method can detect the kind of events that involve communities of different sizes.

The ensemble method takes the communication network, range of minimum and maximum community sizes, change threshold and detection threshold parameters as input. It executes the basic event detection method based on the change tracking in the number of communities with the given parameters. If the count of events of a week fulfills the predefined detection threshold, the ensemble method marks that week to include an event. The details of the first ensemble method can be seen in Algorithm 5.

**Algorithm 5** Event Detection via Ensemble of Change on # of Comm

```
Require: setOfComm, listOf(minCommSize, maxCommSize),
  changeThreshold, detectionThreshold
Ensure: events
  events = emptyList()
  listOfEventList = emptyList()
  for each minCommSize, maxCommSize in listOf(minCommSize, maxComm-
  Size) do
    eventList = EventDetectionviaChangeon#ofComm(
    setOfComm, minCommSize, maxCommSize, changeThreshold)
    add eventList to listOfEventList
  end for
  for i=2 to timeWindowCount do
    detectionCount = 0
    for each eventList in listOfEventList do
      if eventList contains $i$ then
        detectionCount = detectionCount + 1
      end if
    end for
    if detectionCount $\geq$ detectionThreshold then
      add $i$ to events
    end if
  end for
  return events
```

#### 4.3.4.2 Combining the Change Tracking in the Number of Communities and the Change Tracking in the Central Nodes

This hybrid approach is based on the idea that an event might be related with both central node changes and number of communities changes. Therefore, we propose an ensemble method that tracks the changes both in the central nodes and in the number of communities in order to mark a week for event inclusion.

**Algorithm 6** Event Detection via Ensemble of Change on # of Comm and Change On Central Nodes

```
Require: setOfComm, minCommSize, maxCommSize, changeComptMethod
  changeThreshold, detectionThreshold
Ensure: events
  events = emptyList()
  numberOfCommEvents = EventDetectionviaChangeon#ofComm(
  setOfComm, minCommSize, maxCommSize, changeThreshold)
  centralNodeEvents = EventDetectionviaChangeonCentralNodes(
  setOfComm, changeComptMethod, changeThreshold)
  for i=2 to timeWindowCount do
    detectionCount = 0
    if numberOfCommEvents contains $i$ then
      detectionCount = detectionCount + 1
    end if
    if centralNodeEvents contains $i$ then
      detectionCount = detectionCount + 1
    end if
    if detectionCount $\geq$ detectionThreshold then
      add $i$ to events
    end if
  end for
  return events
```

This ensemble method takes the network, the range of minimum and maximum community size, central node change detection type, change threshold and detection threshold parameters as input. The basic event detection method on the change tracking in the number of communities is applied on the data under the given input parameters. Similarly, the basic method on the change tracking in the central nodes is also executed on the same data. A week is marked to include an event, if its event detection count fulfills the given detection threshold parameter. In fact, for this ensemble method, practically the only applicable event detection thresholds are 1 and 2 in terms of event count, as each of the two methods can return either 0 or 1 as the output. In other words, we can state that count threshold value one refers to or/union, and count threshold value two refers to and/intersection of the basic methods' results. The details of this algorithm are given in Algorithm 6.

### 4.3.5 Improved Ensemble Strategies

#### 4.3.5.1 Ensemble of Tracking the Change in the Number of Communities Method with Different Bucket Variations

In order to detect complex events that can effect communities of only certain size ranges, an ensemble algorithm that combines change tracking of number of communities for different buckets is generated. For example an event may effect the communities in the buckets Q1 and Q3. Then, an ensemble of change tracking for bucket Q1 and change tracking for bucket Q3 can be constructed. At this point, there are two alternative ways to determine the final decision of the ensemble: ANDing vs ORing the results from individual event detectors. Since all kinds of communication data can be processed to generate buckets, this ensemble approach can be used very easily regardless of the data size or structure.

#### 4.3.5.2 Ensemble of Tracking the Change in the Number of Communities and Tracking the Change in the Central Nodes

As another complex event type, there may be cases that affect both the number of communities and the central nodes. Therefore, as a natural extension, this study generates an ensemble combining two basic methods, one tacking the number of communities, and the other tracking the central nodes. As in the previous ensemble method, in this one, the final decision can be determined through logical operations of ANDing and ORing the results of the individual event detectors. Similar to the above approach, it is also possible to determine changes in community structures between consecutive time steps to all kinds of data sizes and structures.

## 4.4 Event Detection Methods by Tracking the Change in the Communication Trends

In this method, firstly, the initial network is constructed. Then, the community detection algorithms, either dSLM or InfoMap, is applied on this initial network. Rather than applying community detection at each time interval, communities are determined only once, and then in each of the subsequent time intervals, the change in the communication trends is computed based on this initial structure.

The change is quantified using two different metrics: inter-intra links ratio (IILR) and modularity (MOD). Two different comparison strategies are used for change detection. In the first one, the standard deviation (SD) value obtained in a time interval is compared against a given threshold. Note that this value can be computed either with IILR (CT-IILR-SD) or modularity (CT-MOD-SD). In the second one, rather than the standard deviation value obtained for the network of the time interval, the amount of change with respect to the previous time interval is compared against a given threshold. For this strategy, the value of the network can be determined by IILR (CT-IILR) or modularity (CT-MOD). A similar approach is used by Moriano et al. in [17] such that the authors use InfoMap as the community detection algorithm, IILR as the value to track and standard deviation as the comparison method. This study extends the method in [17] by including dSLM as the community detection algorithm, modularity as the value to track and the change from the previous time interval as the comparison strategy.

The algorithm using the standard deviation as comparison method is given in Algorithm 7, whereas the other one using the change from the previous time interval is given in Algorithm 8. In both algorithms, the tracked value of graph corresponding to the given time interval can be computed by either IILR or modularity (denoted with

**Algorithm 7** Event Detection via Change on CT (St. Dev.)

```
Require: IN, setOfGraphs, equation, bucket, threshold
Ensure: events
  for i ← 1 to timeWindowCount do
    cur ← CT(IN, setOfGraphs($t_i$), equation, bucket)
    if cur $\geq$ threshold then
      add $i$ to events
    end if
  end for
  return events
```

*equation* parameter in the algorithms).

**Algorithm 8** Event Detection via Change on CT (Cur-Prev)

```
Require: IN, setOfGraphs, equation, bucket, threshold
Ensure: events
  prev ← CT(IN, setOfGraphs($t_1$), equation, bucket)
  for i=2 to timeWindowCount do
    cur ← CT(IN, setOfGraphs($t_i$), equation, bucket)
    change ← abs(cur $-$ prev)/prev
    if change $\geq$ threshold then
      add $i$ to events
    end if
    prev ← cur
  end for
  return events
```

## 4.5 Event Detection Methods by Tracking the Change in the Graph Embeddings

This study also hypothesizes that the changes that are above some threshold in the graph structure of a temporal graph might indicate an event. The changes in the graph structure can be computed by comparing two graphs in the set of graphs in a temporal graph data set. This comparison can be done by measuring the minimum number of graph operations required to transform one graph to the other graph [55]. However, such methods are proven to be NP-hard in execution time aspect [56]. To overcome the execution time limitation, instead of comparing the raw original graphs, the embedding vectors of the graphs are compared to detect the changes in the graph structure. Comparing two vectors are much easier and much less time consuming than comparing two graphs.

In this method, the community detection step is not needed since it does not focus on communities but focus on the embedding vector of each time step's network. Though, the initial "Pre-processing & Graph Generation" step explained in Section 4.2 is still needed to generate the graphs. This event detection method that is based on graph embeddings is applied to a given data set by following steps:

1. The graphs for each time step are generated as described in Section 4.2.
2. The various graph embedding methods such as Graph2vec [48], GL2vec [49], FEATHER [50], LDP [51] are applied on the generated graphs of each time step and the initial network (IN). Thus, for each graph embedding method, we now have a set of vectors that represent the time steps' graphs for the given data set and also a separate vector for the initial network.
3. For each set of vectors that represent the time steps' graphs:
   (a) The mean vector is computed and selected.
   (b) The distance between each vector in the set and the mean vector is computed and stored. All the vector distance computation methods defined in Section 3.4 are used in this step and all the results are stored.
   (c) The distance between each vector in the set and the initial networks vector is computed and stored. All the vector distance computation methods defined in Section 3.4 are used in this step and all the results are stored.

   At this point, there are 2 set of distance values for each graph embedding method and vector distance computation method variations. One set is for distances from the mean vector and the other is for the distances from the initial networks vector.

4. For each set of distances, from minimum value to maximum value 100 threshold values are computed. Based on each threshold, the time steps whose distances that are higher than the threshold value are marked as event time steps. Given 100 threshold values, it produces 100 precision and recall values. The average precision value is computed by calculating the area under precision recall curve.

Thus, for each graph embedding method, each vector distance computation method, and each reference vector, either mean or initial networks vector, the average precision value is produced.

# CHAPTER 5

# EXPERIMENTS ON EVENT DETECTION PERFORMANCE

The effectiveness of the proposed methods is analyzed in terms of a variety of aspects including community size, group of communities, time resolution and scalability, through a set of experiments. In this section, the data sets are presented followed by experiment settings. Then the results of the experiments analyzing different aspects are given in separate subsections.

## 5.1 Data Sets & Ground Truths

In this section, the data sets used in the experiments and the ground truth events within the data sets are presented.

**Enron.** This is an email communication data set containing more than 125,000 emails sent by 184 employees of Enron company between 1999-01-01 and 2002-04-30. Enron is a U.S company that has filed for bankruptcy just before 2000 [57]. There are seven significant events marked in this time interval [58, 17]. The following dates include ground truth events: 2001-05-17, 2001-07-12, 2001-08-03, 2001-10-16, 2001-12-02, 2002-02-14, 2002-04-09. The details of the events are given in Table 5.1 which is taken from Moriano et. al. [17].

**Boston Bombing Tweets.** The data set contains tweet communication networks extracted from more than 456 million English tweets posted in April 2013 [17]. The mention networks contain around 7 million nodes and 10 million edges for each day.

Table 5.1: Enron ground truth events

| Event ID | Date | Description |
|---|---|---|
| 1 | 2001-05-17 | Schwarzenegger, Lay, Milken meeting. |
| 2 | 2001-07-12 | Quarterly conference call. |
| 3 | 2001-08-03 | Skilling makes a bullish speech on Enron Energy Services. That afternoon, he lays off 300 employees. |
| 4 | 2001-10-16 | Enron reports a 618 million third-quarter loss and declares a 1.01 billion non-recurring charge against its balance sheet, partly related to “structured finance” operations run by chief financial officer Andrew Fastow. In the analyst conference call that day, Lay also announces a 1.2 billion cut in shareholder equity. |
| 5 | 2001-12-02 | Enron, at the time the largest bankruptcy in U.S. history, files for Chapter 11 bankruptcy protection. |
| 6 | 2002-02-14 | Sherron Watkins, the Enron whistleblower, testifies before a Congressional panel against Skilling and Lay. |
| 7 | 2002-04-09 | David Duncan, Arthur Andersen’s former top auditor, pleads guilty to obstruction. |

The retweet networks contain around 3.5 million nodes and 4.2 million edges for each day. There are two major events in this time interval. These ground truth events are as follows:

- 2013-04-15 Bombing
- 2013-04-19 Manhunt

In the results, mention network of this data set is abbreviated as BBM. Similarly, retweet network is shown as BBR.

**Reality Mining.** This data set is a CDR collection containing 99,633 call record instances. It includes the communication data of 97 faculty, student, and staff at MIT, recorded by the software on their mobile devices over 50 weeks from August 2004 to July 2005 [59]. There are 50 weeks in this data set's time span and the weeks having id 6, 12, 13, 15, 16, 17, 19, 20, 21, 22, 23, 27, 31, 32, 34, 35 are marked as ground truth event weeks. These weeks involve semester breaks, exam and sponsor weeks, and holidays [1]. The name of this data set is abbreviated as RM in the result tables. The sub communication network obtained by SMS interactions is shown with RMS and the sub-network of voice calls is shown as RMV. Different graph structures based on edge directions (directed-D vs. undirected-U), and edge weight (weighted-W vs. unweighted-U) are constructed.

**Turkish GSM Operator.** There are 297,009,183 call records in this CDR data set. In total, there are 12,521,352 individuals (nodes in the communication network), and 56,316,192 phone call relations (edges) recorded in September 2012[1]. When the days in the data set are labelled from 1 to 30, the days with id 2, 3, 4, 5, 6, 11, 16, 17, 18, 22, 24, 25, 29, 30 are marked as ground truth event days. These events are extracted from public news resources in Turkey for September 2012. This full data set is denoted as TGSM in the result tables. This data set corresponds to calls covering a large area including a smaller and much more crowded city center. Some events in this data set are related with the whole country whereas some others are related with the city center. In order to evaluate the effect of considering only central area, two subsets of the data set are constructed. The whole area covered by the communication network is represented as a 12 by 12 grid. Then, the middle 4 by 4 grid region is marked as *Big Central Square (BCS)*, and 2 by 2 inmost grid is marked as *Central Square* as shown in Figure 5.1. The details of these subsets are as follows:

- **Big Central Square (BCS).** It contains the records collected from 11545 base stations in the central region out of total 13281 base stations. In total there are 2,853,473 nodes and 11,376,594 edges in this communication network.
- **Central Square (CS).** It contains the records collected from the central 85 base stations out of total 13281 base stations. In total there are 1,335,945 nodes and 3,568,967 edges in this subset.

[1] The 56,316,192 edges correspond to distinct interactions within 297,009,183 entries.

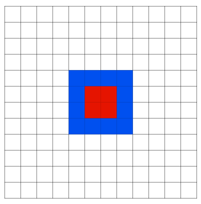

Figure 5.1: Illustration of Big Central Square and Central Squares subsets of Turkish GSM Operator data set

## 5.2 Early Experiments of Community Structure Based Methods with RM Data Sets

Event detection experiments are conducted on weighted and unweighted versions of the network under varying change thresholds. The experiments of the method that detects events via tracking the change on community members have the same results for both weighted and unweighted versions of the network. Therefore, we present the results for this method as a single table.

In the tables, the first column includes the change model parameter analyzed. For basic model, this column includes the community size, the type of change in central node, or the type of change in the community members, depending on the model to be analyzed. For ensemble methods, the first column includes a set of change parameters that belong to the models that are combined. For the ensemble of number of communities methods' variations, this includes a set of community size ranges. For the ensemble of number of communities and central nodes methods, this column includes the community size range and the change detection method. In the ensemble methods, additionally, the first column includes event detection threshold value specified after *DT* prefix. In all the tables, the second column shows the applied change threshold. For each method, we present the best three results with respect to F1-measurement, per parameter setting.

### 5.2.1 Experiments on Basic Methods

For the first basic method of change detection in the number of communities, initially, we grouped the communities with respect to the size by observing the number of communities of each size range on several week samples. The results for this type of partitioning are given in Table 5.2 and 5.4 for weighted and unweighted networks, respectively.

Table 5.2: Change in # of Communities on Weighted Net. (Initial Partitioning)

| Type | Thr | Precision | Recall | F1-meas. |
|---|---|---|---|---|
| 3-5 nodes | 0.05 | 0.62 | 0.81 | 0.70 |
| 3-5 nodes | 0.15 | 0.62 | 0.81 | 0.70 |
| 3-5 nodes | 0.25 | 0.6 | 0.75 | 0.67 |
| 6-10 nodes | 0.05 | 0.38 | 0.56 | 0.46 |
| 6-10 nodes | 0.15 | 0.38 | 0.56 | 0.46 |
| 6-10 nodes | 0.25 | 0.36 | 0.5 | 0.42 |
| 11-20 nodes | 0.05 | 0.48 | 0.88 | 0.63 |
| 11-20 nodes | 0.15 | 0.48 | 0.88 | 0.63 |
| 11-20 nodes | 0.25 | 0.5 | 0.75 | 0.60 |
| 21-30 nodes | 0.05 | 0.32 | 0.44 | 0.38 |
| 21-30 nodes | 0.15 | 0.32 | 0.44 | 0.38 |
| 21-30 nodes | 0.25 | 0.32 | 0.44 | 0.38 |
| 31-40 nodes | 0.05 | 0.55 | 0.38 | 0.45 |
| 31-40 nodes | 0.15 | 0.55 | 0.38 | 0.45 |
| 31-40 nodes | 0.25 | 0.55 | 0.38 | 0.45 |

We further analyzed the effect of partitioning under several variations, and observed that the best result is obtained under the partitioning with respect to the following size ranges: 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90. The results under this partitioning are given in Table 5.3 and Table 5.5 for weighted and unweighted graphs, respectively. We have observed that this change in the partitioning did not lead to an increase in the maximum F1-measure, however, the average performance of the size ranges has increased. Additionally, these new size ranges provide better results in ensemble method.

Table 5.3: Change in # of Communities on Weighted Net. (Improved Partitioning)

| Type | Thr | Precision | Recall | F1-meas. |
|---|---|---|---|---|
| 3-5 nodes | 0.05 | 0.62 | 0.81 | 0.70 |
| 3-5 nodes | 0.10 | 0.62 | 0.81 | 0.70 |
| 3-5 nodes | 0.15 | 0.62 | 0.81 | 0.70 |
| 6-8 nodes | 0.45 | 0.71 | 0.62 | 0.67 |
| 6-8 nodes | 0.50 | 0.71 | 0.62 | 0.67 |
| 6-8 nodes | 0.55 | 0.71 | 0.62 | 0.67 |
| 12-16 nodes | 0.35 | 0.50 | 0.62 | 0.56 |
| 12-16 nodes | 0.40 | 0.50 | 0.62 | 0.56 |
| 12-16 nodes | 0.60 | 0.56 | 0.56 | 0.56 |
| 32-90 nodes | 0.05 | 0.64 | 0.44 | 0.52 |
| 32-90 nodes | 0.10 | 0.64 | 0.44 | 0.52 |
| 32-90 nodes | 0.15 | 0.64 | 0.44 | 0.52 |
| 10-11 nodes | 0.05 | 0.38 | 0.50 | 0.43 |
| 10-11 nodes | 0.10 | 0.38 | 0.50 | 0.43 |
| 10-11 nodes | 0.15 | 0.38 | 0.50 | 0.43 |
| 17-31 nodes | 0.05 | 0.38 | 0.50 | 0.43 |
| 17-31 nodes | 0.10 | 0.38 | 0.50 | 0.43 |
| 17-31 nodes | 0.15 | 0.35 | 0.44 | 0.39 |
| 9-9 nodes | 0.05 | 0.46 | 0.38 | 0.41 |
| 9-9 nodes | 0.10 | 0.46 | 0.38 | 0.41 |
| 9-9 nodes | 0.15 | 0.46 | 0.38 | 0.41 |

As shown in Table 5.6 and Table 5.7, the method of tracking change in central nodes present a similar performance for weighted and unweighted graphs. On the overall, although the recall is high, the precision value is lower than the method on the number of communities, due to high number of false positives.

Table 5.4: Change in # of Communities on Unweighted Net. (Initial Partitioning)

| Type | Thr | Precision | Recall | F1-meas. |
|---|---|---|---|---|
| 3-5 nodes | 0.05 | 0.62 | 0.81 | 0.70 |
| 3-5 nodes | 0.15 | 0.62 | 0.81 | 0.70 |
| 3-5 nodes | 0.25 | 0.62 | 0.81 | 0.70 |
| 6-10 nodes | 0.05 | 0.43 | 0.62 | 0.51 |
| 6-10 nodes | 0.15 | 0.43 | 0.62 | 0.51 |
| 6-10 nodes | 0.25 | 0.43 | 0.62 | 0.51 |
| 11-20 nodes | 0.05 | 0.54 | 0.94 | 0.69 |
| 11-20 nodes | 0.15 | 0.54 | 0.94 | 0.69 |
| 11-20 nodes | 0.25 | 0.58 | 0.88 | 0.7 |
| 21-30 nodes | 0.05 | 0.48 | 0.69 | 0.57 |
| 21-30 nodes | 0.15 | 0.48 | 0.69 | 0.57 |
| 21-30 nodes | 0.25 | 0.48 | 0.69 | 0.57 |
| 31-40 nodes | 0.05 | 0.53 | 0.56 | 0.55 |
| 31-40 nodes | 0.15 | 0.53 | 0.56 | 0.55 |
| 31-40 nodes | 0.25 | 0.5 | 0.5 | 0.5 |

Table 5.5: Change in # of Communities on Unweighted Net. (Improved Partitioning)

| Type | Thr | Precision | Recall | F1-meas. |
|---|---|---|---|---|
| 3-5 nodes | 0.05 | 0.62 | 0.81 | 0.70 |
| 3-5 nodes | 0.10 | 0.62 | 0.81 | 0.70 |
| 3-5 nodes | 0.15 | 0.62 | 0.81 | 0.70 |
| 32-90 nodes | 0.05 | 0.60 | 0.75 | 0.67 |
| 32-90 nodes | 0.10 | 0.60 | 0.75 | 0.67 |
| 32-90 nodes | 0.15 | 0.58 | 0.69 | 0.63 |
| 12-16 nodes | 0.60 | 0.57 | 0.75 | 0.65 |
| 12-16 nodes | 0.55 | 0.55 | 0.75 | 0.63 |
| 12-16 nodes | 0.05 | 0.43 | 1.00 | 0.60 |
| 9-9 nodes | 0.05 | 0.89 | 0.50 | 0.64 |
| 9-9 nodes | 0.10 | 0.89 | 0.50 | 0.64 |
| 9-9 nodes | 0.15 | 0.89 | 0.50 | 0.64 |
| 17-31 nodes | 0.25 | 0.50 | 0.69 | 0.58 |
| 17-31 nodes | 0.20 | 0.48 | 0.69 | 0.56 |
| 17-31 nodes | 0.05 | 0.46 | 0.69 | 0.55 |
| 6-8 nodes | 0.05 | 0.41 | 0.44 | 0.42 |
| 6-8 nodes | 0.10 | 0.41 | 0.44 | 0.42 |
| 6-8 nodes | 0.15 | 0.41 | 0.44 | 0.42 |
| 10-11 nodes | 0.55 | 0.33 | 0.38 | 0.35 |
| 10-11 nodes | 0.60 | 0.33 | 0.38 | 0.35 |
| 10-11 nodes | 0.65 | 0.33 | 0.38 | 0.35 |

Table 5.6: Change in the Central Nodes on Weighted Net.

| Type | Thr | Precision | Recall | F1-meas. |
|---|---|---|---|---|
| NEW | 0.05 | 0.33 | 0.94 | 0.49 |
| NEW | 0.15 | 0.33 | 0.94 | 0.49 |
| NEW | 0.25 | 0.33 | 0.94 | 0.49 |
| NOT_ANY_MORE | 0.05 | 0.33 | 0.94 | 0.49 |
| NOT_ANY_MORE | 0.15 | 0.33 | 0.94 | 0.49 |
| NOT_ANY_MORE | 0.65 | 0.39 | 0.75 | 0.52 |
| NOT_ANY_MORE_NEW | 0.05 | 0.33 | 0.94 | 0.49 |
| NOT_ANY_MORE_NEW | 0.15 | 0.33 | 0.94 | 0.49 |
| NOT_ANY_MORE_NEW | 0.25 | 0.33 | 0.94 | 0.49 |
| SIZE | 0.05 | 0.32 | 0.75 | 0.45 |
| SIZE | 0.15 | 0.36 | 0.5 | 0.42 |
| SIZE | 0.25 | 0.27 | 0.19 | 0.23 |

Table 5.7: Change in the Central Nodes on Unweighted Net.

| Type | Thr | Precision | Recall | F1-meas. |
|---|---|---|---|---|
| NEW | 0.35 | 0.34 | 0.94 | 0.5 |
| NEW | 0.45 | 0.34 | 0.94 | 0.5 |
| NEW | 0.55 | 0.34 | 0.94 | 0.5 |
| NOT_ANY_MORE | 0.05 | 0.33 | 0.94 | 0.49 |
| NOT_ANY_MORE | 0.15 | 0.33 | 0.94 | 0.49 |
| NOT_ANY_MORE | 0.65 | 0.39 | 0.81 | 0.53 |
| NOT_ANY_MORE_NEW | 0.05 | 0.33 | 0.94 | 0.49 |
| NOT_ANY_MORE_NEW | 0.15 | 0.33 | 0.94 | 0.49 |
| NOT_ANY_MORE_NEW | 0.25 | 0.33 | 0.94 | 0.49 |
| SIZE | 0.05 | 0.31 | 0.75 | 0.44 |
| SIZE | 0.15 | 0.36 | 0.5 | 0.42 |
| SIZE | 0.25 | 0.25 | 0.19 | 0.22 |

Among the three basic change tracking methods, the one on the change in the community members has the lowest event detection performance (Table 5.8). Among the variations of this basic method, considering the minimum change can not capture any event, which is an expected result since small change in the community may be observed under ordinary behaviour as well. The performance under the maximum change and average change is just the same, showing a strong bias towards marking week to include an event.

Table 5.8: Change in the Community Members

| Type | Thr | Precision | Recall | F1-meas. |
|---|---|---|---|---|
| AVG | 0.05 | 0.33 | 1 | 0.5 |
| AVG | 0.15 | 0.33 | 1 | 0.5 |
| AVG | 0.25 | 0.33 | 1 | 0.5 |
| MAX | 0.05 | 0.33 | 1 | 0.5 |
| MAX | 0.15 | 0.33 | 1 | 0.5 |
| MAX | 0.25 | 0.33 | 1 | 0.5 |
| MIN | 0.05 | 0 | 0 | 0 |
| MIN | 0.15 | 0 | 0 | 0 |
| MIN | 0.25 | 0 | 0 | 0 |

### 5.2.2 Experiments on Ensemble Methods

Among the basic community structure change models, since the first two perform better, we constructed two ensemble methods by combining variations of them. The first one combines the variations of the first basic method, whereas the second ensemble method combines the best performing variations of the first and the second basic methods.

The results of the first ensemble method are given in Table 5.9 and 5.10 on weighted and unweighted graphs, respectively. Similarly, the performance for weighted and unweighted versions of the graph for the second ensemble method are given in Table 5.11 and 5.12, respectively. The event detection performance in terms of F1-measure is higher for unweighted graphs for both of the ensemble methods.

Table 5.9: Ensemble # of Communities on Weighted Net.

| Type | Thr | Prec. | Rec. | F1. |
|---|---|---|---|---|
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 3 | 0.05 | 0.62 | 0.94 | 0.75 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 3 | 0.10 | 0.62 | 0.94 | 0.75 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 3 | 0.15 | 0.62 | 0.94 | 0.75 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 2 | 0.45 | 0.48 | 0.94 | 0.64 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 2 | 0.50 | 0.48 | 0.94 | 0.64 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 2 | 0.55 | 0.50 | 0.81 | 0.62 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 4 | 0.05 | 0.64 | 0.56 | 0.60 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 4 | 0.10 | 0.64 | 0.56 | 0.60 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 4 | 0.15 | 0.62 | 0.50 | 0.55 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 1 | 0.70 | 0.42 | 0.94 | 0.58 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 1 | 0.75 | 0.42 | 0.94 | 0.58 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 1 | 0.55 | 0.40 | 1.00 | 0.57 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 5 | 0.25 | 0.67 | 0.25 | 0.36 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 5 | 0.30 | 0.67 | 0.25 | 0.36 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 5 | 0.35 | 0.67 | 0.25 | 0.36 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 6 | 0.05 | 1.00 | 0.19 | 0.32 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 6 | 0.10 | 1.00 | 0.19 | 0.32 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 6 | 0.15 | 1.00 | 0.19 | 0.32 |

Table 5.10: Ensemble # of Communities on Unweighted Net.

| Type | Thr | Pre. | Rec. | F1. |
|---|---|---|---|---|
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 4 | 0.05 | 0.68 | 0.94 | 0.79 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 4 | 0.10 | 0.68 | 0.94 | 0.79 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 4 | 0.25 | 0.70 | 0.88 | 0.78 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 3 | 0.35 | 0.52 | 0.88 | 0.65 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 3 | 0.05 | 0.48 | 0.94 | 0.64 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 3 | 0.10 | 0.48 | 0.94 | 0.64 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 5 | 0.20 | 0.89 | 0.50 | 0.64 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 5 | 0.25 | 0.89 | 0.50 | 0.64 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 5 | 0.30 | 0.89 | 0.50 | 0.64 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 1 | 0.85 | 0.46 | 1.00 | 0.63 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 1 | 0.90 | 0.46 | 1.00 | 0.63 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 1 | 0.95 | 0.46 | 1.00 | 0.63 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 2 | 0.55 | 0.44 | 1.00 | 0.62 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 2 | 0.45 | 0.42 | 1.00 | 0.59 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 2 | 0.50 | 0.42 | 1.00 | 0.59 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 6 | 0.05 | 1.00 | 0.19 | 0.32 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 6 | 0.10 | 1.00 | 0.19 | 0.32 |
| 3-5, 6-8, 9-9, 10-11, 12-16, 17-31, 32-90, DT: 6 | 0.15 | 1.00 | 0.19 | 0.32 |

Table 5.11: Ensemble # of Communities and Central Nodes on Weighted Net.

| Type | Chng | Pre. | Rec. | F1. |
|---|---|---|---|---|
| 3-5, SIZE, DT: 2 | 0.05 | 0.60 | 0.75 | 0.67 |
| 3-5, NEW, DT: 2 | 0.05 | 0.60 | 0.75 | 0.67 |
| 3-5, NOT_ANY_MORE, DT: 2 | 0.05 | 0.60 | 0.75 | 0.67 |
| 3-5, NOT_ANY_MORE_NEW, DT: 2 | 0.05 | 0.60 | 0.75 | 0.67 |
| 6-8, SIZE, DT: 2 | 0.05 | 0.60 | 0.56 | 0.58 |
| 6-8, NEW, DT: 2 | 0.05 | 0.60 | 0.56 | 0.58 |
| 6-8, NOT_ANY_MORE, DT: 2 | 0.05 | 0.60 | 0.56 | 0.58 |
| 6-8, NOT_ANY_MORE_NEW, DT: 2 | 0.05 | 0.60 | 0.56 | 0.58 |
| 12-16, SIZE, DT: 2 | 0.05 | 0.45 | 0.62 | 0.53 |
| 12-16, NEW, DT: 2 | 0.05 | 0.45 | 0.62 | 0.53 |
| 12-16, NOT_ANY_MORE, DT: 2 | 0.05 | 0.45 | 0.62 | 0.53 |
| 12-16, NOT_ANY_MORE_NEW, DT: 2 | 0.05 | 0.45 | 0.62 | 0.53 |
| 32-90, SIZE, DT: 2 | 0.05 | 0.64 | 0.44 | 0.52 |
| 32-90, NEW, DT: 2 | 0.05 | 0.64 | 0.44 | 0.52 |
| 32-90, NOT_ANY_MORE, DT: 2 | 0.05 | 0.64 | 0.44 | 0.52 |
| 32-90, NOT_ANY_MORE_NEW, DT: 2 | 0.05 | 0.64 | 0.44 | 0.52 |
| 3-5, SIZE, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 3-5, NEW, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 3-5, NOT_ANY_MORE, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 3-5, NOT_ANY_MORE_NEW, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 6-8, SIZE, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 6-8, NEW, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 6-8, NOT_ANY_MORE, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 6-8, NOT_ANY_MORE_NEW, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 17-31, SIZE, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 17-31, NEW, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 17-31, NOT_ANY_MORE, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |
| 17-31, NOT_ANY_MORE_NEW, DT: 1 | 0.05 | 0.34 | 1.00 | 0.51 |

Table 5.12: Ensemble # of Communities and Central Nodes on Unweighted Net.

| Type | Chng | Pre. | Rec. | F1. |
|---|---|---|---|---|
| 3-5, SIZE, DT: 2 | 0.05 | 0.62 | 0.81 | 0.70 |
| 3-5, NEW, DT: 2 | 0.05 | 0.62 | 0.81 | 0.70 |
| 3-5, NOT_ANY_MORE, DT: 2 | 0.05 | 0.62 | 0.81 | 0.70 |
| 3-5, NOT_ANY_MORE_NEW, DT: 2 | 0.05 | 0.62 | 0.81 | 0.70 |
| 9-9, SIZE, DT: 2 | 0.05 | 1.00 | 0.50 | 0.67 |
| 9-9, NEW, DT: 2 | 0.05 | 1.00 | 0.50 | 0.67 |
| 9-9, NOT_ANY_MORE, DT: 2 | 0.05 | 1.00 | 0.50 | 0.67 |
| 9-9, NOT_ANY_MORE_NEW, DT: 2 | 0.05 | 1.00 | 0.50 | 0.67 |
| 32-90, SIZE, DT: 2 | 0.05 | 0.58 | 0.69 | 0.63 |
| 32-90, NEW, DT: 2 | 0.05 | 0.58 | 0.69 | 0.63 |
| 32-90, NOT_ANY_MORE, DT: 2 | 0.05 | 0.58 | 0.69 | 0.63 |
| 32-90, NOT_ANY_MORE_NEW, DT: 2 | 0.05 | 0.58 | 0.69 | 0.63 |
| 12-16, SIZE, DT: 2 | 0.05 | 0.44 | 0.94 | 0.60 |
| 12-16, NEW, DT: 2 | 0.05 | 0.44 | 0.94 | 0.60 |
| 12-16, NOT_ANY_MORE, DT: 2 | 0.05 | 0.44 | 0.94 | 0.60 |
| 12-16, NOT_ANY_MORE_NEW, DT: 2 | 0.05 | 0.44 | 0.94 | 0.60 |
| 17-31, SIZE, DT: 2 | 0.05 | 0.45 | 0.62 | 0.53 |
| 17-31, NEW, DT: 2 | 0.05 | 0.45 | 0.62 | 0.53 |
| 17-31, NOT_ANY_MORE, DT: 2 | 0.05 | 0.45 | 0.62 | 0.53 |
| 17-31, NOT_ANY_MORE_NEW, DT: 2 | 0.05 | 0.45 | 0.62 | 0.53 |
| 6-8, SIZE, DT: 1 | 0.05 | 0.35 | 1.00 | 0.52 |
| 6-8, NEW, DT: 1 | 0.05 | 0.35 | 1.00 | 0.52 |
| 6-8, NOT_ANY_MORE, DT: 1 | 0.05 | 0.35 | 1.00 | 0.52 |
| 6-8, NOT_ANY_MORE_NEW, DT: 1 | 0.05 | 0.35 | 1.00 | 0.52 |
| 32-90, SIZE, DT: 1 | 0.05 | 0.35 | 1.00 | 0.52 |
| 32-90, NEW, DT: 1 | 0.05 | 0.35 | 1.00 | 0.52 |
| 32-90, NOT_ANY_MORE, DT: 1 | 0.05 | 0.35 | 1.00 | 0.52 |
| 32-90, NOT_ANY_MORE_NEW, DT: 1 | 0.05 | 0.35 | 1.00 | 0.52 |

### 5.2.3 Experiments on Scalability

In order to analyze the scalability of the proposed method, we conducted an experiment to measure the time performance under varying data set size. The original data set contains nearly 100000 call detail records. For each week of this data, we constructed 5 different versions by constructing multiples of the data set by 1, 2, 4, 8, and 16 Each of the resulting 5 different version of data set contains nearly 100000, 200000, 400000, 800000, and 1,6M call detail records, respectively. As in the previous experiments, we performed the analysis for both weighted and unweighted graphs.

The execution times of the proposed method on the scaled data sets are presented in Table 5.13, Figure 5.2, and Figure 5.3. Table 5.13 lists all the durations in seconds, whereas Figure 5.2, and Figure 5.3 show the nature of the trend for weighted and unweighted graphs, respectively. As seen in the results, the quadratic complexity of the community detection method is reflected in the execution times.

Table 5.13: Data Set Scalability Performance Analysis

| Graph Type | Data Set Size (# of CDR) | Exec. Time (secs) |
|---|---|---|
| Weighted | 100,000 | 15 |
| Weighted | 200,000 | 27 |
| Weighted | 400,000 | 89 |
| Weighted | 800,000 | 308 |
| Weighted | 1,600,000 | 1231 |
| Unweighted | 100,000 | 12 |
| Unweighted | 200,000 | 27 |
| Unweighted | 400,000 | 91 |
| Unweighted | 800,000 | 296 |
| Unweighted | 1,600,000 | 1260 |

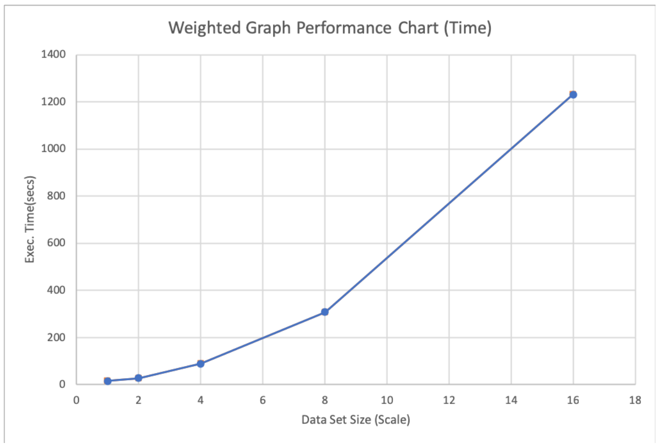

Figure 5.2: Weighted Graph: Exec. Time/Data Set Size

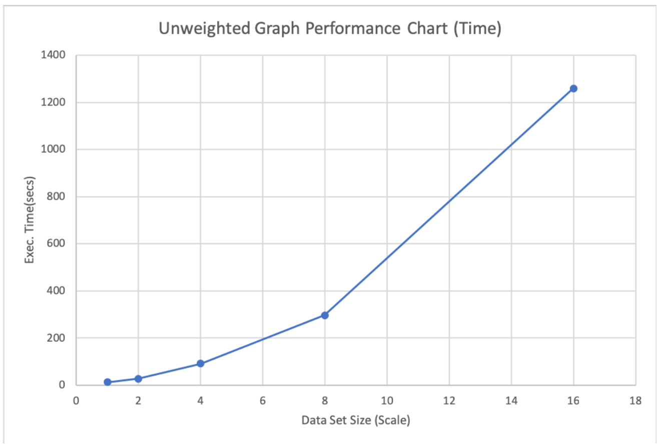

Figure 5.3: Unweighted Graph: Exec. Time/Data Set Size

### 5.2.4 Overview & Discussion

We can summarize the results of the experiments in this section as follows:

- Among basic change tracking models, event detection according to the change in number of communities has the highest scores in terms of precision and F1-measure. Although it is the simplest approach, it provides a stronger indicator for event. Additionally, it has better potential to be applied online due to its lower computationally cost.

- Change tracking on smallest communities (communities with 3-5 members) provides the highest precision and recall results. This indicates that an event triggers communication among small groups that emerges new, possibly short-lived communities.

- As a general observation for the first two basic change tracking methods, lower change threshold values provides the highest accuracy scores. Except for just a few cases, under increasing change threshold, both precision and recall values decrease.
- It is observed that the basic method on tracking the change of central nodes is not sensitive to change threshold. Although high recall values (recall 0.94) are obtained for this method, precision values remain low.
- As in the second method, the method on tracking the change in the community members appears to be insensitive to change threshold. The accuracy values remain the same for each of AVG, MAX and MIN settings. This method provides the highest recall score (recall value 1.00) under AVG and MAX, however, low precision value shows that it has tendency towards labeling time windows as event.
- The accuracy values under weighted and unweighted network structures do not indicate a strong difference. The results are the same for the method with change of community members for both of the graphs. For the other two basic and two ensemble methods, the accuracy values slightly higher for the unweighted network structure. Thus, unweighted network structure may be preferable due to its lightweight structure.
- The ensemble method combining the variations of change tracking on number of communities gives the best results by even exceeding the results of number of communities based basic method. The highest F1-measure value is 0.79 for unweighted and 0.75 for weighted directed network. Therefore, we can state that the ensemble method that we combines the results of number of communities change tracking basic method variations is successful.
- On the other hand, the ensemble method that combines the results of number of communities and central nodes based event detection method variations do not give better results than the basic methods. The highest F1-measure value is 0.67 for unweighted and 0.70 for weighted directed network.

- The scalability experiments reflect the quadratic complexity of the employed community detection method in increasing time costs. This time cost is considered to be similar to previous solutions that rely on the change the graph attributes, such as node centrality [60], [1]. However, such previous studies did not report time cost, we can not provide a quantified comparison for scalability analysis.

## 5.3 Broad Experiments of Community Structure and Communication Trend Based Methods with All Data Sets

In these experiments, the results are analyzed under precision, recall, f-measure, false positive rate, and true positive rate metrics. However, for the overall comparison the area under the Precision-Recall curve is used. The curve is plotted under 100 threshold values and the area under this curve is denoted as Average Precision [1].

The experiments are conducted on a computer with 32 GB RAM and Intel(R) Core(TM) i7-10750H CPU@2.60GHz processor.

A set of experiments is devised in order to answer the following research questions:

- RQ1. How effective are the proposed change tracking methods for event detection?
- RQ2. Is the size and amount of the communities to consider an effect for event detection? How does it affect the detection performance?
- RQ3. How does the time resolution affect the community detection?
- RQ4. Does community change comparison method effect the community detection performance?
- RQ5. Does community detection based event detection approach provide a feasible solution in terms of running time performance?

In the following subsections, the experiments conducted to answer the above research questions are presented. Additionally, the event detection performance of these experiments is compared with the previous studies.

The results of the proposed methods with various bucket configurations per data set are presented in Appendices. These results can be examined to drill on the performances of the proposed methods under different configurations per data set.

### 5.3.1 Event Detection Performance of the Methods under the Basic Settings

In this experiment, the performance of the proposed approaches are analyzed under the basic settings (i.e., full data set, all communities (Bucket A), finest time resolution (Resolution 1)). In Table 5.14, the list of methods and their abbreviations are presented. However, due to the high number of methods to list, in the initial analysis, only the best performing methods are demonstrated in Table 5.15. In this table, for Boston Bombing (BBM and BBR), Enron (ENR), Reality Mining (RMS and RMV) and Turkish GSM (TGSM) data sets, the best performance obtained for data set modeling variations is given. The detailed results for RM and TGSM data sets are presented in Table 5.16 and Table 5.17, respectively. In Table 5.16, name of the data sets reflect the graph structure used for modeling the network. For example S-DU denotes SMS communication network with Directed Unweighted graph.

Table 5.14: Abbreviations for the basic methods

| Abbreviation | Method Description |
|---|---|
| CN-NEW | Track the number of new central nodes |
| CN-SIZE | Track the number of central nodes |
| CN-NOT | Track the number of not anymore central nodes |
| CN-NOT-NEW | Track the number of not anymore + new central nodes |
| NOC | Track the number of communities |
| CT-IILR | Track the IILR by comparing with prev time |
| CT-MOD | Track the Modularity by comparing with prev time |
| CT-IILR-SD | Track the IILR with respect to standard deviation |
| CT-MOD-SD | Track the Modularity with respect to standard deviation |

According to the results, there is no single method that consistently outperforms the others on all data sets. CN-NEW gives the highest average precision on three data sets (BBM, RMS, RMV), and CN-NOT, CT-IILR, CT-MOD provide the best results

Table 5.15: Evaluation on Basic Methods (In Average Precision)

| Method/Dataset | BBM | BBR | ENR | RMS | RMV | TGSM |
|---|---|---|---|---|---|---|
| CN-NEW | **0.58** | 0.13 | 0.07 | **0.26** | **0.35** | 0.6 |
| CN-NOT | 0.11 | 0.05 | 0.07 | 0.19 | 0.32 | **0.62** |
| CN-NOT-NEW | 0.03 | 0.03 | 0.13 | 0.18 | 0.3 | 0.24 |
| CN-SIZE | 0.12 | 0.2 | 0.13 | 0.25 | **0.35** | 0.57 |
| CT-IILR | 0.17 | 0.32 | 0.07 | 0.25 | 0.23 | 0.3 |
| CT-MOD | 0.35 | 0.13 | **0.19** | 0.25 | 0.25 | 0.6 |
| NOC | 0.18 | **0.6** | 0.08 | 0.25 | 0.33 | 0.41 |

Table 5.16: Evaluation on Basic Methods: RM data sets (In Average Precision)

| Method/Dataset | S-DU | S-DW | S-UW | V-DU | V-DW | V-UW |
|---|---|---|---|---|---|---|
| CN-NEW | **0.26** | **0.34** | **0.4** | **0.35** | 0.26 | 0.3 |
| CN-NOT | 0.19 | 0.26 | 0.33 | 0.32 | 0.23 | 0.27 |
| CN-NOT-NEW | 0.18 | 0.24 | 0.23 | 0.3 | 0.23 | 0.17 |
| CN-SIZE | 0.25 | 0.27 | 0.36 | **0.35** | 0.23 | **0.33** |
| CT-IILR | 0.25 | 0.26 | 0.24 | 0.23 | 0.23 | 0.22 |
| CT-MOD | 0.25 | 0.22 | 0.24 | 0.25 | 0.23 | 0.25 |
| NOC | 0.25 | 0.26 | 0.21 | 0.33 | **0.33** | 0.24 |

Table 5.17: Evaluation on Basic Methods: TGSM data set (In Average Precision)

| Method/Dataset | TGSM | BCS | CS |
|---|---|---|---|
| CN-NEW | 0.6 | 0.62 | 0.5 |
| CN-NOT | **0.62** | 0.54 | 0.42 |
| CN-NOT-NEW | 0.24 | 0.24 | 0.22 |
| CN-SIZE | 0.57 | 0.56 | 0.53 |
| CT-IILR | 0.3 | 0.43 | 0.52 |
| CT-MOD | 0.6 | **0.75** | **0.62** |
| NOC | 0.41 | 0.55 | 0.58 |

for the other three data sets (TGSM, BBR, ENR), respectively.

At this point, it is worth examining the results given in Table 5.17. In these results, it is observed that using the central part of the geographical area increases the event detection performance up until 0.75 with CT-MOD on BCS. Narrowing down the area decreases the data set size as well. The effect of this change on running time is elaborated in Section 5.3.6.

### 5.3.2 Event Detection Performance of the Ensemble Methods under the Basic Settings

Detection performance of several ensemble event detectors are also analyzed under the basic setting. The list of methods with their abbreviations are presented in Table 5.18. The results are presented in Table 5.19. As in the previous analysis, in this table, the highest event detection values obtained for data set modeling variations of BB, ENR, RM and TGSM data sets are given. The detailed results for RM and TGSM data sets are presented in Table 5.20 and Table 5.21, respectively.

Table 5.18: Abbreviations for the ensemble methods

| Abbreviation | Method Description |
|---|---|
| NOC-qx..qy-OR | OR ensemble of buckets x..y in NOC |
| NOC-qx..qy-AND | AND ensemble of buckets x..y in NOC |
| NOC-NEW-OR | OR ensemble of NOC and CN-NEW |
| NOC-NEW-AND | AND ensemble of NOC and CN-NEW |
| NOC-NOT-OR | OR ensemble of NOC and CN-NOT |
| NOC-NOT-AND | AND ensemble of NOC and CN-NOT |
| NOC-NOT-NEW-OR | OR ensemble of NOC and CN-NOT-NEW |
| NOC-NOT-NEW-AND | AND ensemble of NOC and CN-NOT-NEW |
| NOC-SIZE-OR | OR ensemble of NOC and CN-SIZE |
| NOC-SIZE-AND | AND ensemble of NOC and CN-SIZE |

In the results, it is seen that there is performance improvement only for some of the data sets. The ensemble method bringing the improvement varies for each of these

Table 5.19: Evaluation on Ensemble Methods (In Average Precision)

| Method/Dataset | BBM | BBR | ENR | RMS | RMV | TGSM |
|---|---|---|---|---|---|---|
| NOC-q1q2-OR | 0.13 | 0.55 | 0.1 | 0.27 | 0.3 | 0.54 |
| NOC-q1q3-OR | 0.13 | 0.55 | 0.1 | 0.27 | 0.3 | 0.54 |
| NOC-q1q4-OR | 0.13 | NA | 0.1 | 0.27 | 0.3 | 0.54 |
| NOC-q2q3-OR | 0.13 | 0.55 | 0.1 | 0.27 | 0.3 | 0.54 |
| NOC-q2q4-OR | 0.13 | NA | 0.1 | 0.27 | 0.3 | 0.54 |
| NOC-q3q4-OR | 0.13 | NA | 0.1 | 0.27 | 0.3 | 0.54 |
| NOC-q1q2-AND | 0.19 | **0.6** | 0.08 | 0.2 | 0.29 | 0.4 |
| NOC-q1q3-AND | 0.19 | **0.6** | 0.08 | 0.2 | 0.29 | 0.4 |
| NOC-q1q4-AND | 0.19 | NA | 0.08 | 0.2 | 0.29 | 0.4 |
| NOC-q2q3-AND | 0.19 | **0.6** | 0.08 | 0.2 | 0.29 | 0.4 |
| NOC-q2q4-AND | 0.19 | NA | 0.08 | 0.2 | 0.29 | 0.4 |
| NOC-q3q4-AND | 0.19 | NA | 0.08 | 0.2 | 0.29 | 0.4 |
| NOC-q1q2q3-OR | 0.13 | 0.55 | **0.18** | 0.28 | 0.27 | 0.55 |
| NOC-q1q2q4-OR | 0.13 | NA | **0.18** | 0.28 | 0.27 | 0.55 |
| NOC-q2q3q4-OR | 0.13 | NA | **0.18** | 0.28 | 0.27 | 0.55 |
| NOC-q1q3q4-OR | 0.13 | NA | **0.18** | 0.28 | 0.27 | 0.55 |
| NOC-q1q2q3-AND | 0.16 | **0.6** | 0.08 | 0.08 | 0.25 | 0.43 |
| NOC-q1q2q4-AND | 0.16 | NA | 0.08 | 0.08 | 0.25 | 0.43 |
| NOC-q2q3q4-AND | 0.16 | NA | 0.08 | 0.08 | 0.25 | 0.43 |
| NOC-q1q3q4-AND | 0.16 | NA | 0.08 | 0.08 | 0.25 | 0.43 |
| NOC-q1q2q3q4-OR | 0.1 | NA | 0.1 | **0.33** | 0.27 | 0.57 |
| NOC-q1q2q3q4-AND | 0.17 | NA | 0.05 | 0.02 | 0.18 | 0.38 |
| NOC-NEW-OR | **0.58** | 0.13 | 0.08 | 0.28 | 0.32 | 0.6 |
| NOC-NEW-AND | 0.18 | **0.6** | 0.08 | 0.23 | 0.43 | 0.41 |
| NOC-NOT-OR | 0.11 | 0.05 | 0.08 | 0.26 | 0.32 | **0.62** |
| NOC-NOT-AND | 0.18 | **0.6** | 0.08 | 0.2 | 0.39 | 0.41 |
| NOC-NOT-NEW-OR | 0.03 | 0.03 | 0.12 | 0.27 | 0.32 | 0.24 |
| NOC-NOT-NEW-AND | 0.18 | **0.6** | 0.07 | 0.21 | 0.41 | 0.41 |
| NOC-SIZE-OR | 0.27 | 0.2 | 0.14 | 0.28 | 0.29 | 0.56 |
| NOC-SIZE-AND | 0.06 | **0.6** | 0.07 | 0.21 | **0.46** | 0.43 |

Table 5.20: Evaluation on Ensemble Methods: RM (In Average Precision)

| Method/Dataset | SDU | SDW | SUW | VDU | VDW | VUW |
|---|---|---|---|---|---|---|
| NOC-q1q2-OR | 0.27 | 0.26 | 0.31 | 0.3 | 0.28 | 0.29 |
| NOC-q1q3-OR | 0.27 | 0.26 | 0.31 | 0.3 | 0.28 | 0.29 |
| NOC-q1q4-OR | 0.27 | 0.26 | 0.31 | 0.3 | 0.28 | 0.29 |
| NOC-q2q3-OR | 0.27 | 0.26 | 0.31 | 0.3 | 0.28 | 0.29 |
| NOC-q2q4-OR | 0.27 | 0.26 | 0.31 | 0.3 | 0.28 | 0.29 |
| NOC-q3q4-OR | 0.27 | 0.26 | 0.31 | 0.3 | 0.28 | 0.29 |
| NOC-q1q2-AND | 0.2 | 0.1 | 0.15 | 0.29 | 0.22 | 0.18 |
| NOC-q1q3-AND | 0.2 | 0.1 | 0.15 | 0.29 | 0.22 | 0.18 |
| NOC-q1q4-AND | 0.2 | 0.1 | 0.15 | 0.29 | 0.22 | 0.18 |
| NOC-q2q3-AND | 0.2 | 0.1 | 0.15 | 0.29 | 0.22 | 0.18 |
| NOC-q2q4-AND | 0.2 | 0.1 | 0.15 | 0.29 | 0.22 | 0.18 |
| NOC-q3q4-AND | 0.2 | 0.1 | 0.15 | 0.29 | 0.22 | 0.18 |
| NOC-q1q2q3-OR | 0.28 | 0.27 | 0.31 | 0.27 | 0.27 | 0.3 |
| NOC-q1q2q4-OR | 0.28 | 0.27 | 0.31 | 0.27 | 0.27 | 0.3 |
| NOC-q2q3q4-OR | 0.28 | 0.27 | 0.31 | 0.27 | 0.27 | 0.3 |
| NOC-q1q3q4-OR | 0.28 | 0.27 | 0.31 | 0.27 | 0.27 | 0.3 |
| NOC-q1q2q3-AND | 0.08 | 0.07 | 0.13 | 0.25 | 0.15 | 0.11 |
| NOC-q1q2q4-AND | 0.08 | 0.07 | 0.13 | 0.25 | 0.15 | 0.11 |
| NOC-q2q3q4-AND | 0.08 | 0.07 | 0.13 | 0.25 | 0.15 | 0.11 |
| NOC-q1q3q4-AND | 0.08 | 0.07 | 0.13 | 0.25 | 0.15 | 0.11 |
| NOC-q1q2q3q4-OR | **0.33** | **0.3** | 0.28 | 0.27 | 0.25 | 0.3 |
| NOC-q1q2q3q4-AND | 0.02 | 0.02 | 0.01 | 0.18 | 0.11 | 0.02 |
| NOC-NEW-OR | 0.28 | 0.29 | **0.36** | 0.32 | 0.3 | **0.32** |
| NOC-NEW-AND | 0.23 | 0.28 | 0.23 | 0.43 | 0.28 | 0.22 |
| NOC-NOT-OR | 0.26 | 0.27 | 0.32 | 0.32 | **0.32** | 0.31 |
| NOC-NOT-AND | 0.2 | 0.25 | 0.24 | 0.39 | 0.23 | 0.22 |
| NOC-NOT-NEW-OR | 0.27 | 0.28 | 0.22 | 0.32 | 0.29 | 0.2 |
| NOC-NOT-NEW-AND | 0.21 | 0.26 | 0.23 | 0.41 | 0.25 | 0.22 |
| NOC-SIZE-OR | 0.28 | 0.27 | 0.34 | 0.29 | 0.28 | 0.31 |
| NOC-SIZE-AND | 0.21 | 0.24 | 0.24 | **0.46** | 0.26 | 0.27 |

Table 5.21: Evaluation on Ensemble Methods: TGSM (In Average Precision)

| Method/Dataset | TGSM | BCS | CS |
|---|---|---|---|
| NOC-q1q2-OR | 0.54 | 0.57 | 0.65 |
| NOC-q1q3-OR | 0.54 | 0.57 | 0.65 |
| NOC-q1q4-OR | 0.54 | 0.57 | 0.65 |
| NOC-q2q3-OR | 0.54 | 0.57 | 0.65 |
| NOC-q2q4-OR | 0.54 | 0.57 | 0.65 |
| NOC-q3q4-OR | 0.54 | 0.57 | 0.65 |
| NOC-q1q2-AND | 0.4 | 0.49 | 0.59 |
| NOC-q1q3-AND | 0.4 | 0.49 | 0.59 |
| NOC-q1q4-AND | 0.4 | 0.49 | 0.59 |
| NOC-q2q3-AND | 0.4 | 0.49 | 0.59 |
| NOC-q2q4-AND | 0.4 | 0.49 | 0.59 |
| NOC-q3q4-AND | 0.4 | 0.49 | 0.59 |
| NOC-q1q2q3-OR | 0.55 | 0.61 | **0.66** |
| NOC-q1q2q4-OR | 0.55 | 0.61 | **0.66** |
| NOC-q2q3q4-OR | 0.55 | 0.61 | **0.66** |
| NOC-q1q3q4-OR | 0.55 | 0.61 | **0.66** |
| NOC-q1q2q3-AND | 0.43 | 0.45 | 0.51 |
| NOC-q1q2q4-AND | 0.43 | 0.45 | 0.51 |
| NOC-q2q3q4-AND | 0.43 | 0.45 | 0.51 |
| NOC-q1q3q4-AND | 0.43 | 0.45 | 0.51 |
| NOC-q1q2q3q4-OR | 0.57 | **0.62** | 0.65 |
| NOC-q1q2q3q4-AND | 0.38 | 0.45 | 0.51 |
| NOC-NEW-OR | 0.6 | **0.62** | 0.54 |
| NOC-NEW-AND | 0.41 | 0.55 | 0.53 |
| NOC-NOT-OR | **0.62** | 0.54 | 0.45 |
| NOC-NOT-AND | 0.41 | 0.55 | 0.53 |
| NOC-NOT-NEW-OR | 0.24 | 0.24 | 0.25 |
| NOC-NOT-NEW-AND | 0.41 | 0.55 | 0.53 |
| NOC-SIZE-OR | 0.56 | 0.6 | 0.61 |
| NOC-SIZE-AND | 0.43 | 0.55 | 0.49 |

cases. For RMS data set, average precision increases from 0.26 to 0.33 by NOC-q1q2q3q4-OR. For RMV, there is increase from 0.35 to 0.46 average precision by NOC-SIZE-AND.

The same situation is also observed for the variations of TGSM data set, which are BCS and CS. Some of the ensemble methods provide improvement over basic methods. For BCS data set, the highest performance obtained by CT-MOD is not exceeded by any ensemble method. However, for CS data set, average precision increases from 0.62 to 0.66 by NOC-q1q2q3-OR.

### 5.3.3 Analysis on the Effect of Bucket Sizes

As described in Section 3.1, the detected communities are sorted in ascending order of community size and grouped under buckets. In this grouping, for instance, A refers to all communities, whereas Q3 refers to the set of communities in the third quartile. The results of the experiments under these buckets are presented in Table 5.22 for CDR data sets (RMS, RMV and TGSM) and in Table 5.23 for Social media data sets (BBM, BBR and ENR).

The results clearly show that, for CDR data sets, certain buckets of communities affect the event detection performance. In CDR data sets, for RMS data set, performance of CN-NEW method increases from 0.26 to 0.38 when H1 is used. Similarly, for RMV, the performance of CN-NEW rises from 0.35 to 0.48. However, the bucket to prefer is data dependent. On the other hand, for social media data sets, using buckets generally degrades the performance, the highest average precision results are obtained with all communities (bucket A).

### 5.3.4 Analysis on the Effect of Time Resolution

In our data sets, the events are marked per day or per week. Therefore, the changes in the communities are tracked on the time resolution given in the data set. However, this resolution can be too strict when the effect of the event on the communication network is propagated with a delay. In order to analyze the effect of time resolution

Table 5.22: Evaluation on Basic Methods: Bucket Size Effect (CDR data sets) (In Average Precision)

| Method/Dataset | RMS | | | | | | | | RMV | | | | | | | | TGSM | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bucket Size | A | F | H1 | H2 | Q1 | Q2 | Q3 | Q4 | A | F | H1 | H2 | Q1 | Q2 | Q3 | Q4 | A | F | H1 | H2 | Q1 | Q2 | Q3 | Q4 |
| CN-NEW | **0.26** | **0.35** | **0.38** | **0.33** | **0.28** | 0.2 | **0.33** | 0.2 | **0.35** | **0.4** | **0.48** | **0.41** | **0.4** | **0.49** | **0.32** | 0.42 | 0.6 | 0.61 | 0.61 | 0.46 | **0.78** | **0.62** | 0.5 | 0.37 |
| CN-NOT | 0.19 | 0.24 | 0.3 | 0.25 | 0.24 | 0.22 | 0.28 | **0.27** | 0.32 | 0.38 | 0.37 | 0.33 | 0.29 | 0.46 | 0.22 | 0.43 | **0.62** | 0.61 | 0.63 | 0.49 | 0.66 | 0.52 | 0.52 | 0.38 |
| CN-NOT-NEW | 0.18 | 0.23 | 0.21 | 0.17 | 0.16 | 0.16 | 0.22 | 0.15 | 0.3 | 0.36 | 0.26 | 0.29 | 0.21 | 0.23 | 0.2 | 0.29 | 0.24 | 0.24 | 0.24 | 0.22 | 0.25 | 0.24 | 0.24 | 0.25 |
| CN-SIZE | 0.25 | 0.24 | 0.3 | 0.27 | 0.26 | 0.24 | **0.33** | 0.25 | **0.35** | 0.37 | 0.36 | 0.33 | 0.3 | 0.41 | 0.3 | **0.46** | 0.57 | **0.71** | **0.67** | **0.67** | 0.64 | 0.56 | **0.71** | **0.63** |
| NOC | 0.25 | 0.21 | 0.18 | 0.25 | **0.28** | **0.28** | 0.32 | 0.12 | 0.33 | 0.25 | 0.26 | 0.18 | 0.1 | 0.37 | 0.26 | 0.4 | 0.41 | 0.51 | 0.5 | 0.51 | 0.57 | 0.52 | 0.49 | 0.41 |

Table 5.23: Evaluation on Basic Methods: Bucket Size Effect (Social media data sets) (In Average Precision)

| Method/Dataset | BBM | | | | | | | | BBR | | | | | | | | ENR | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bucket Size | A | F | H1 | H2 | Q1 | Q2 | Q3 | Q4 | A | F | H1 | H2 | Q1 | Q2 | Q3 | Q4 | A | F | H1 | H2 | Q1 | Q2 | Q3 | Q4 |
| CN-NEW | **0.58** | **0.52** | 0.04 | 0.09 | **0.04** | 0.04 | 0.08 | 0.04 | 0.13 | 0.06 | 0.05 | 0.05 | 0.05 | 0.05 | 0.01 | 0.01 | 0.07 | 0.06 | 0.14 | 0.07 | 0.06 | 0.04 | **0.09** | 0.01 |
| CN-NOT | 0.11 | 0.27 | 0.04 | 0.09 | **0.04** | 0.04 | 0.07 | 0.04 | 0.05 | 0.06 | 0.06 | 0.05 | 0.06 | 0.05 | 0.01 | 0.01 | 0.07 | 0.06 | 0.12 | 0.08 | 0.03 | 0.03 | 0.06 | 0.05 |
| CN-NOT-NEW | 0.03 | 0.03 | 0.03 | 0.04 | 0.03 | 0.03 | 0.04 | 0.04 | 0.03 | 0.03 | 0.03 | 0.03 | 0.03 | 0.04 | 0.01 | 0.01 | **0.13** | 0.07 | 0.05 | **0.11** | 0.04 | 0.04 | 0.08 | **0.07** |
| CN-SIZE | 0.12 | 0.1 | 0.07 | 0.11 | 0.01 | 0.1 | **0.14** | 0.05 | 0.2 | 0.28 | 0.27 | 0.3 | 0.27 | 0.35 | 0.06 | 0.06 | **0.13** | **0.11** | **0.18** | 0.06 | 0.11 | **0.06** | 0.08 | 0.03 |
| NOC | 0.18 | 0.09 | **0.08** | **0.12** | 0.03 | **0.13** | 0.11 | **0.06** | **0.6** | **0.55** | **0.55** | **0.55** | **0.55** | **0.56** | **0.54** | **0.54** | 0.08 | 0.08 | 0.17 | 0.02 | **0.13** | 0.02 | 0.05 | 0.01 |

for change detection, the setting for ENR and TGSM data sets is modified, such that, time window is set as 2 days and 3 days, in addition to daily time window (window size 1). The results of the analysis is given in Table 5.24. The analysis shows a considerable increase in the performance of CT-MOD, which provided the highest accuracy performance with the finest grained time resolution for ENR data set, specifically for window size 2. The trend is similar also for the other methods.

Table 5.24: Evaluation on Basic Methods: Resolution Effect (In Average Precision)

| Method/Dataset | ENRON | | | TGSM | | | BCS | | | CS | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Resolution | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| CN-NEW | 0.07 | 0.07 | 0.2 | 0.6 | 0.85 | 0.83 | 0.62 | 0.68 | 0.86 | 0.5 | 0.81 | 0.71 |
| CN-NOT | 0.07 | 0.18 | 0.17 | **0.62** | 0.76 | 0.73 | 0.54 | 0.84 | 0.73 | 0.42 | 0.8 | 0.67 |
| CN-NOT-NEW | 0.13 | **0.26** | 0.21 | 0.24 | 0.32 | 0.33 | 0.24 | 0.32 | 0.33 | 0.22 | 0.32 | 0.33 |
| CN-SIZE | 0.13 | 0.17 | 0.09 | 0.57 | 0.64 | 0.67 | 0.56 | 0.77 | 0.58 | 0.53 | 0.79 | 0.75 |
| CT-IILR | 0.07 | 0.18 | **0.29** | 0.3 | 0.06 | 0.08 | 0.43 | 0.5 | 0.08 | 0.52 | 0.72 | 0.58 |
| CT-MOD | **0.19** | 0.22 | 0.18 | 0.6 | **0.94** | **0.9** | **0.75** | **0.94** | **0.89** | **0.62** | **0.92** | **0.9** |
| NOC | 0.08 | 0.13 | 0.24 | 0.41 | 0.79 | 0.46 | 0.55 | 0.82 | 0.65 | 0.58 | 0.81 | 0.78 |

### 5.3.5 Analysis on the Effect of Community Change Comparison Method

In [17], Moriano et al. propose event detection through tracking communication trend changes in communities using InfoMap as the community detection algorithm, IILR as the value to track and standard deviation as the comparison method. In our proposed solution, solution of [17] is modified such that, dSLM is used as the community detection algorithm, modularity is used as the value to track and change from the previous time interval is used as change comparison method. In this analysis, the change comparison method in our setting is replaced with change from the previous time interval tracking in order to evaluate its effect on event detection performance. The results are presented in Table 5.25. The results reveal that the way change is computed is effective on the result. The proposed change tracking method has clear advantage for the CT-MOD method as it produces higher accuracy values for 6 out of 8 data sets, whereas it performs similar with the standard deviation based change tracking method for the CT-IILR method.

Table 5.25: Evaluation on Basic Methods: SD Effect (In Average Precision)

| Method/Dataset | BBM | BBR | ENRON | RMS | RMV | TGSM | BCS | CS |
|---|---|---|---|---|---|---|---|---|
| CT-IILR | 0.17 | **0.32** | 0.07 | 0.25 | 0.23 | 0.3 | 0.43 | 0.52 |
| CT-IILR-SD | 0.15 | 0.28 | 0.16 | **0.39** | 0.27 | 0.38 | 0.39 | 0.37 |
| CT-MOD | **0.35** | 0.13 | **0.19** | 0.25 | 0.25 | **0.6** | **0.75** | **0.62** |
| CT-MOD-SD | 0.19 | 0.09 | 0.05 | 0.34 | **0.5** | 0.36 | 0.44 | 0.51 |

### 5.3.6 Analysis on the Running Time Performance

Although all investigated methods include community detection as a core task, the change detection approaches and techniques differ considerably. This difference leads to variations in time costs for the methods. In Table 5.26, the running time duration of the basic methods are given in seconds. All the CN methods are computed in one run to increase performance. In the table, the average execution time for 4 CN based methods are reported. As seen in the results, the methods based on change tracking on the community structure (CN and NOC) are computationally lightweight compared to communication trend based methods (CT-IILR, CT-MOD).

The running time duration for variations of TGSM data set are given in Table 5.27. As expected, execution time decreases for BCS and CS, as the size of the processed data gets smaller compared to TGSM. As an interesting observation, focusing on central regions affects also event detection performance positively.

Table 5.26: Evaluation on Basic Methods: Run Times (In Seconds)

| Method/Dataset | BBM | BBR | ENR | RMS | RMV | TGSM |
|---|---|---|---|---|---|---|
| CN | 158.22 | 58.49 | **0.08** | 0.06 | 0.11 | 70.93 |
| CT-IILR | 470.9 | 181.41 | 0.24 | 0.09 | 0.14 | 154.47 |
| CT-MOD | 93.62 | 45.6 | 0.15 | 0.12 | 0.14 | 51.95 |
| NOC | **13.59** | **4.89** | 0.41 | **0.03** | **0.03** | **4.12** |

Table 5.27: Evaluation on Basic Methods: Run Times: TGSM data sets (In Seconds)

| Method/Dataset | TGSM | BCS | CS |
|---|---|---|---|
| CN | 70.93 | 5.43 | 1.26 |
| CT-IILR | 154.47 | 13.29 | 5.11 |
| CT-MOD | 51.95 | 3.33 | 0.86 |
| NOC | **4.12** | **0.65** | **0.28** |

### 5.3.7 Comparison with Baseline Studies

As the baseline methods, the methods by Moriano et al. in [17] and Rayana et al. in [1] are used.

**Comparison with [17].** As the basic differences from the communication trend based solution of our work, Moriano et al. present experiments by using the InfoMap community detection algorithm and standard deviation for change computing. In [17], the event detection performance on Enron data set is presented only in terms of precision-recall curve. Our results on the same data set with CT based methods using dSLM are given in Table 5.25. Additionally, the average precision with InfoMap community detection algorithm (using the original source codes provided by Moriano et al.) on Enron data set is 0.14.

Moriano et. al. also conducted experiments on Boston Bombing data set. They provide the results of tracking the inter-intra ratio visually [17], but precision-recall curve or the average precision result are not reported. In our analysis, average precision values obtained on this data set is relatively low. This is possibly due to noise in the data set such that there are fluctuations, and on day 8, the data includes another event not related with Boston bombing and hence not considered in the ground truth event set. Additionally, there is missing data for day 18. As for execution time performance, for this data set, our community detection step takes 356 minutes in total for both mention and retweet networks. The event detection step takes 103 minutes in total. Thus, experiments including both networks are completed in 459 minutes (7.6 hours).

**Comparison with [1].** When the original source code of the methods in [1] is executed on Boston Bombing data set, the mention network experiment results in out of memory error under 32 GB RAM. The retweet network data is read in about 1 hour, but the codes cannot process it even after 7 hours. Therefore, the experiment is canceled after 8 hours without obtaining any result.

On reality mining data set, 5 basic and 5 ensemble methods of [1] are executed on both directed weighted voice call and SMS networks. The analysis on the voice call network takes 258 seconds, whereas it takes 36 seconds on the SMS network experiment. Thus, in total, it takes 294 seconds for all of the analysis on this data set. When all the proposed methods are performed (7 basic and 2 ensemble methods, with all bucket size variations) on all Reality mining networks, the community detection step takes 4 seconds, and the event detection step takes 13 seconds. Hence, on the total, it takes 17 seconds for our methods to conduct the whole analysis for this data set.

When the source code of [1] is executed on Turkish GSM Operator data set, it results with out of memory error, as well, under 32 GB RAM. For BCS and CS subsets, the code execution was stuck at PTSAD method of [1]. Therefore, the part of the source code related with this method is commented out and other base methods are used. The analysis on BCS data set takes 1272 seconds, and on CS data set, it takes 448 seconds. The execution times of all our proposed methods on TGSM, BCS and CS data sets are given in Table 5.28 for three different time resolutions. As seen in the table, for BCS it takes 701 seconds and for CS data set it takes 246 seconds (in resolution 1). Therefore, the execution time efficiency advantage of the proposed methods is clear.

Table 5.28: Execution times of TGSM Experiments (7 basic, 2 ensemble methods)

| Dataset | Comm. Det. | Event Det. | Total Exec. |
|---|---|---|---|
| TGSM | 89 mins | 86 mins | 175 mins |
| BCS | 253 secs | 448 secs | 701 secs |
| CS | 60 secs | 186 secs | 246 secs |

Table 5.29 presents the event detection performance of the methods in [1] on Reality Mining, BCS and CS data sets, and the best results obtained by the proposed methods

on the same data sets. For BCS data set, CT-MOD on bucket A (abbreviated as CT-MOD-A in the table) provides the highest average precision with a gap of 8% (compared to EBED outdegree). For the other data sets, although the methods in [1] provide higher event detection performance, the results of the proposed methods can be considered comparable also considering their time performance advantage, particularly for RMS-DW and CS data sets.

Table 5.29: Evaluation of Methods in [1] (In Average Precision)

| Method/Dataset | RMS-DW | RMV-DW | BCS | CS |
|---|---|---|---|---|
| EBED indeg | 0.39 | 0.34 | 0.63 | **0.67** |
| PTSAD indeg | 0.51 | 0.48 | NA | NA |
| SpiritTest indeg | **0.68** | 0.56 | 0.48 | 0.49 |
| ASED indeg | 0.47 | 0.58 | 0.4 | 0.4 |
| MAED indeg | 0.53 | **0.66** | 0.48 | 0.47 |
| EBED outdeg | 0.37 | 0.47 | 0.67 | 0.65 |
| PTSAD outdeg | 0.39 | 0.53 | NA | NA |
| SpiritTest outdeg | 0.63 | 0.5 | 0.4 | 0.4 |
| ASED outdeg | 0.5 | **0.66** | 0.44 | 0.66 |
| MAED outdeg | 0.58 | 0.65 | 0.46 | 0.47 |
| Full Ensemble | 0.53 | 0.6 | 0.46 | 0.63 |
| SELECT-H Ensemble | 0.6 | 0.64 | 0.42 | 0.5 |
| SELECT-V Ensemble | 0.47 | 0.65 | 0.53 | 0.62 |
| DivE Ensemble | 0.49 | 0.5 | 0.63 | **0.67** |
| ULARA Ensemble | 0.57 | 0.61 | 0.52 | 0.62 |
| CT-IILR-SD - F | 0.6 | 0.25 | 0.39 | 0.35 |
| CT-MOD-SD - A | 0.23 | 0.52 | 0.44 | 0.51 |
| CT-MOD - A | 0.22 | 0.23 | **0.75** | 0.62 |

### 5.3.8 Overview & Discussion

The overview of the performance of all proposed methods on all data sets is presented in Table 5.30. Additionally, an overview of the performance for all buckets on all data sets is provided in Table 5.31. For each data set, the top result is colored with dark brown, the second top result is colored with orange, and the third top result is colored with light yellow. In Table 5.30, the methods are ordered based on the number of colored cells.

Table 5.30: The Overview of the Best Results: Method (In Average Precision)

| Method/Dataset | BBM | BBR | ENR | RMS-DU | RMS-DW | RMS-UW | RMV-DU | RMV-DW | RMV-UW | TGSM | BCS | CS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CN-NEW | 0.58 - A | | | 0.38 - H1 | 0.37 - H1 | 0.40 - A | 0.49 - Q2 | 0.45 - Q4 | 0.39 - H1 | 0.78 - Q1 | 0.67 - H1 | |
| NOC-NEW-OR | 0.58 - A | | 0.20 - H1 | 0.38 - H2 | | 0.40 - H1 | 0.49 - Q4 | | | 0.78 - Q1 | 0.67 - H1 | |
| CT-IILR-SD | | | | 0.39 - A | 0.60 - F | 0.55 - F | | | | | | |
| CT-MOD-SD | | | | | | | 0.50 - A | 0.52 - A | 0.46 - A | | | |
| CN-NOT | 0.27 - F | | | | | 0.39 - Q3 | | 0.46 - Q4 | 0.38 - Q3 | | 0.66 - Q1 | 0.69 - Q2 |
| NOC-NOT-AND | | 0.60 - A | | 0.37 - Q3 | | | 0.53 - Q4 | | | | | |
| CT-IILR | | | | | 0.43 - Q3 | | | | | 0.86 - Q2 | | 0.69 - H1 |
| CT-MOD | 0.35 - A | | 0.19 - A | | | | | | | | 0.75 - A | |
| NOC-SIZE-OR | 0.27 - F | | | 0.37 - Q3 | | | | | 0.38 - Q4 | 0.71 - F | 0.67 - Q2 | |
| NOC-SIZE-AND | | 0.60 - A | 0.21 - H1 | | | | | | | | | |
| NOC | | 0.60 - A | | | | | | | | | | 0.70 - Q4 |
| NOC-NEW-AND | | 0.60 - A | | | | | | 0.46 - Q3 | | | | |
| NOC-NOT-OR | 0.27 - F | | | | | | | | | | 0.66 - Q1 | 0.73 - Q2 |
| NOC-NOT-NEW-AND | | 0.60 - A | 0.20 - H1 | | | | | | | | | |
| CN-SIZE | | 0.35 - Q2 | | | | | | | 0.39 - F | 0.71 - F | | |
| NOC-q1q2-AND | | 0.60 - A | | | | | | | | | | |
| NOC-q1q3-AND | | 0.60 - A | | | | | | | | | | |
| NOC-q2q3-AND | | 0.60 - A | | | | | | | | | | |
| NOC-q1q2q3-AND | | 0.60 - A | | | | | | | | | | |
| NOC-q1q2-OR | | 0.55 - A | | | | | | | | | | |
| NOC-q1q3-OR | | 0.55 - A | | | | | | | | | | |
| NOC-q2q3-OR | | 0.55 - A | | | | | | | | | | |
| NOC-q1q2q3-OR | | 0.55 - A | | | | | | | | | | |

The points are given such that first results are 3, the second results are 2, and the third results are 1 point. Thus, each method has a total point. The methods are ordered descending by total points. Thus, the best method is on top.

Table 5.31: The Overview of the Best Results: Bucket (In Average Precision)

| Bucket/Dataset | BBM | BBR | ENR | RMS-DU | RMS-DW | RMS-UW | RMV-DU | RMV-DW | RMV-UW | TGSM | BCS | CS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 0.58 | 0.6 | 0.19 | 0.39 | | 0.4 | 0.5 | 0.52 | 0.46 | | 0.75 | |
| H1 | | 0.56 | 0.21 | 0.38 | | 0.4 | | 0.44 | 0.39 | | 0.67 | 0.69 |
| F | 0.52 | 0.56 | | | 0.6 | 0.55 | | | 0.39 | 0.71 | | |
| Q2 | | 0.57 | | | | 0.39 | 0.49 | | | 0.86 | 0.67 | 0.73 |
| Q3 | 0.25 | | | 0.37 | 0.56 | 0.39 | | 0.46 | 0.38 | 0.71 | | |
| Q4 | | | | | | | 0.53 | 0.46 | 0.39 | | | 0.7 |
| Q1 | | 0.56 | 0.14 | | | | | | | 0.78 | 0.66 | |
| H2 | | | | 0.38 | 0.53 | | | | | | | |

The points are given such that first results are 3, the second results are 2, and the third results are 1 point. Thus, each bucket has a total point. The buckets are ordered descending by total points. Thus, the best bucket is on top.

The results and contributions are analyzed under 4 parts. First, the overall winners of the experimented algorithms are presented. After that, the contributions of this study to community structure based, communication trend based, and general event detection areas are discussed. Additionally, a data set based analysis and discussion is presented.

#### 5.3.8.1 Overall Winners

Based on Table 5.30 which is sorted from best to worst method, the winner is the proposed CN-NEW method. The second one is an ensemble, NOC-NEW-OR. These two community structure based methods are followed by two communication trend based ones, namely CT-IILR-SD and CT-MOD-SD.

As seen in Table 5.30, communication trend based methods tend to perform better on CDR based data sets, whereas community structure based methods tend to perform better on social media-based data sets.

Community structure based methods have better execution time performance than the communication trend based approaches as seen in Table 5.26.

#### 5.3.8.2 Contributions to Community Structure Based Event Detection

In this work, a new bucket concept is introduced. It is experimented on various scale of data sets. As seen in Table 5.22, using a bucket of the data set improves the accuracy of the community structure based event detection algorithms, particularly for the CDR based data sets. For instance, CN-NEW has 0.49 average precision value for the second quarter (Q2) of RMV dataset, whereas it has 0.35 average precision value for RMV data set. So, it can be stated that bucket concept increased average precision of CN-NEW method from 0.35 to 0.49 in RMV, from 0.62 to 0.78 in TGSM, from 0.26 to 0.38 in RMS data sets. As a summary, it can be stated that the bucket concept improves the accuracy of several methods for CDR based data sets. On the other hand, as it can be seen in Table 5.23, the similar improvement has not been observed for social media based data sets.

Table 5.31 also shows that using buckets H1, Q2, and Q4 produce best results for some of the cases. Regarding the execution time, using buckets considerably improves the execution time of the proposed methods since it decreases the amount of data to be processed.

The ensemble strategies devised in this work give clearer results than [16], since logical and/or combinations of the quarter buckets are used to form the ensembles.

Furthermore, the proposed ensemble methods increase the best average precision values for some cases, such as, from 0.26 to 0.33 for RMS, from 0.35 to 0.46 for RMV, and from 0.62 to 0.66 for CS data sets, as it can be seen in Tables 5.15, 5.17, 5.19, 5.21.

The proposed resolution strategy provides the capability to detect events that come with a delayed communication representation. As seen in Table 5.24, this strategy improves the highest average precision values. For example, for different resolution values, average precision increases for ENR from 0.19 to 0.29, for TGSM from 0.62 to 0.94, for BCS from 0.75 to 0.89, and for CS from 0.62 to 0.92.

In this work, as a contribution to community structure based event detection, multiple centrality score calculation methods such as Betweenness, Katz, Harmonic, and PageRank centrality metrics are tested. As a result of these experiments, the highest average precision values for the CN methods have been obtained with the PageRank centrality metric. Thus, only the results with the PageRank centrality metric have been presented.

#### 5.3.8.3 Contributions to Communication Trend Based Event Detection

The study in [17] uses InfoMap as community detection method, whereas the proposed method, CT-IILR-SD, uses dSLM for community detection. For comparison, CT-IILR-SD with InfoMap is applied on ENR data set. CT-IILR-SD produces 0.16 average precision and outperforms CT-IILR-SD with InfoMap which produces 0.14 average precision on ENR data set. Furthermore, as seen in Table 5.25, when the tracked value and the tracking strategy are changed, it is observed that the event detection performance further improves over [17].

Among the communication trend based methods that are experimented, CT-MOD provides the best execution time performance. For instance, it is about 4 times faster than CT-IILR as seen in Tables 5.26, 5.27.

### 5.3.8.4 Comparison with Graph Feature Based Event Detection

The proposed community structure and communication trend based methods can be executed without any resource problems for big data sets (BBM, BBR, TGSM) whereas the graph feature based event detection methods [1] cannot reach the similar scalability level. For small data sets such as RM, community structure and communication trend based methods execute about 20 times faster than the graph feature based event detection methods. As seen in Table 5.29, the best performing proposed methods for these small data sets produce comparable event detection accuracy values with the graph feature based event detection methods in significantly less execution time.

### 5.3.8.5 Data Set based Analysis and Discussion

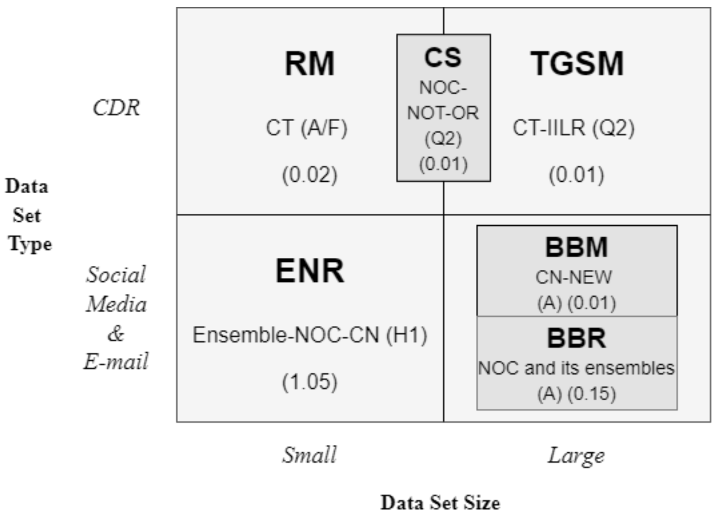

Figure 5.4: Data Matrix to summarize data set based analysis

The best resulting techniques with respect to data sets are summarized in Figure 5.4. In the figure, the data sets are grouped with respect to type (CDR vs. social media, e-mail) and size (small vs. large). For example, RM data set is small in size, and it is a CDR collection. For each data set, the technique giving the best result is also given together with the bucket size and the threshold used in the analysis (given in parenthesis). The denoted threshold value is the one that gives the highest f-measure for that case. Notice that CS data set is medium in size, and hence it is placed separately.

As seen in the figure, the best technique varies with respect to the nature of the data set in terms of size and type. It is observed that ensemble of tracking the change in the number of communities and tracking the change in the central nodes gives the best results for two cases, particularly for ENR and CS data sets.

Additionally, the thresholds used for the algorithms vary with respect to the data set, as well. For example, for CT based method on TGSM data set, a low threshold value of 0.01 gives the best result. On the other hand, for ENR data set, a much higher change threshold value of 1.05 gives the best result. Apparently, the ground truth events in ENR data set lead to a much stronger change in the community structures within communication networks. Another interesting observation is that, for Boston Bombing data set, different threshold values are set for mention (BBM) and retweet (BBR) communication networks. Since higher threshold values are used for BBR data set, we can deduce that retweet communication reflects the events more strongly.

## 5.4 Experiments of Graph Embedding Based Methods with All Data Sets

In these experiments, the area under the Precision-Recall curve is used as the accuracy indicator. The curve is plotted under 100 threshold values and the area under this curve is denoted as Average Precision. [1]

The experiments are conducted on a computer with 32 GB RAM and Intel(R) Core(TM) i7-10750H CPU@2.60GHz processor.

The proposed method as described in Section 4.5 is applied on all the data sets. It produces out of memory errors for the TGSM, BBM, and BBR data sets. Therefore, only graph2vec algorithm is applied on these data sets. The graph2vec algorithm works for TGSM and BBR data sets but gives out of memory error for BBM data set. This way, except BBM, all the data sets have results on graph embedding based event detection methods.

### 5.4.1 Experiments on Big Data Sets: TGSM, BBR

The average precision values of the graph2vec embedding based event detection algorithm with various vector distance computation methods experiment on the TGSM and BBR data sets are demonstrated in Table 5.32. There is also another dimension which is called the reference vector. The vector distances are computed based on a reference vector which is either the mean of the time steps vectors or the vector of the initial network for the given data set. This dimension is also presented as the second header row in Table 5.32. For each column, the cell that holds the highest precision value is marked in bold.

Table 5.32: Evaluation on Graph2Vec Method: TGSM and BBR Data Sets (In Avp.)

| Distance/Dataset | TGSM | | BBR | |
|---|---|---|---|---|
| Reference Vector | Initial | Mean | Initial | Mean |
| Euclidian | 0.5 | **0.59** | 0.06 | 0.08 |
| Cosine | 0.47 | 0.45 | **0.07** | 0.05 |
| Minkowski | 0.5 | 0.55 | 0.06 | 0.08 |
| Bray-Curtis | **0.51** | 0.53 | 0.06 | 0.08 |
| Canberra | 0.43 | 0.5 | **0.07** | 0.08 |
| Chebyshev | 0.48 | 0.5 | 0.05 | **0.12** |
| City Block | 0.5 | 0.55 | 0.06 | 0.08 |
| Correlation | 0.42 | 0.5 | 0.06 | 0.05 |

### 5.4.2 Experiments on Small Data Sets: BCS, CS, ENRON, RMS, RMV

There are four dimensions which are graph embedding method, data set, vector distance computation method, and reference vector. To represent the results effectively, the vector distance computation method is kept as constant, graph embedding method is used as index column, data set is used as the column header having the reference vector values as children. The values in the result tables are all in average precision. For each data set and reference vector pairs, the columns in the result tables, the highest average precision values are marked in bold. This way, the performance of each graph embedding method can be examined by looking at the frequency of bold colors in its row.

So, a table represents the results for each vector distance computation method. The results of the Bray-Curtis distance method can be seen in Table 5.33. The results of the Canberra distance method can be seen in Table 5.34. The results of the Chebyshev distance method can be seen in Table 5.35. The results of the Cityblock distance method can be seen in Table 5.36. The results of the Correlation distance method can be seen in Table 5.37. The results of the Cosine distance method can be seen in Table 5.38. The results of the Euclidian distance method can be seen in Table 5.39. The results of the Minkowski distance method run with $p = 1$ can be seen in Table 5.40.

Table 5.33: Evaluation on Graph Embedding Methods: Bray-Curtis Distance (In Avp.)

| Method/Dataset | BCS | | CS | | ENRON | | RMS | | RMV | |
|---|---|---|---|---|---|---|---|---|---|---|
| Refer. Vector | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean |
| FEATHER | 0.43 | 0.59 | 0.4 | 0.5 | 0 | 0 | 0.24 | 0.23 | 0.24 | 0.23 |
| GL2Vec | 0.53 | 0.45 | 0.46 | **0.54** | 0.09 | 0.09 | **0.36** | **0.38** | **0.32** | **0.51** |
| Graph2Vec | 0.56 | **0.61** | 0.48 | 0.46 | **0.14** | **0.15** | 0.22 | 0.28 | 0.25 | 0.41 |
| LDP | **0.57** | 0.49 | **0.59** | 0.48 | 0 | 0 | 0.21 | 0.22 | 0.2 | 0.23 |

Table 5.34: Evaluation on Graph Embedding Methods: Canberra Distance (In Avp.)

| Method/Dataset | BCS | | CS | | ENRON | | RMS | | RMV | |
|---|---|---|---|---|---|---|---|---|---|---|
| Refer. Vector | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean |
| FEATHER | 0.46 | 0.58 | 0.47 | 0.51 | 0 | 0 | 0.23 | 0.22 | 0.22 | 0.24 |
| GL2Vec | 0.53 | 0.46 | 0.45 | **0.57** | 0.1 | 0.09 | **0.3** | **0.38** | **0.27** | **0.5** |
| Graph2Vec | 0.54 | **0.64** | 0.4 | 0.47 | **0.15** | **0.19** | 0.22 | 0.28 | 0.23 | 0.43 |
| LDP | **0.6** | 0.48 | **0.49** | 0.49 | 0 | 0 | 0.21 | 0.22 | 0.21 | 0.22 |

Table 5.35: Evaluation on Graph Embedding Methods: Chebyshev Distance (In Avp.)

| Method/Dataset | BCS | | CS | | ENRON | | RMS | | RMV | |
|---|---|---|---|---|---|---|---|---|---|---|
| Refer. Vector | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean |
| FEATHER | 0.39 | 0.62 | 0.39 | 0.49 | 0 | 0 | 0.25 | 0.23 | 0.24 | 0.22 |
| GL2Vec | 0.43 | 0.53 | 0.46 | 0.44 | 0.09 | 0.09 | **0.58** | **0.43** | **0.36** | **0.8** |
| Graph2Vec | 0.4 | **0.66** | **0.59** | **0.52** | **0.15** | **0.15** | 0.22 | 0.3 | 0.35 | 0.5 |
| LDP | **0.57** | 0.49 | **0.59** | 0.47 | 0 | 0 | 0.21 | 0.37 | 0.2 | 0.22 |

Table 5.36: Evaluation on Graph Embedding Methods: Cityblock Distance (In Avp.)

| Method/Dataset | BCS | | CS | | ENRON | | RMS | | RMV | |
|---|---|---|---|---|---|---|---|---|---|---|
| Refer. Vector | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean |
| FEATHER | 0.47 | 0.58 | 0.42 | **0.49** | 0 | 0 | 0.24 | 0.22 | 0.25 | 0.23 |
| GL2Vec | 0.55 | 0.42 | 0.4 | 0.47 | 0.09 | 0.09 | **0.46** | **0.41** | **0.49** | 0.44 |
| Graph2Vec | 0.54 | **0.7** | 0.42 | 0.41 | **0.15** | **0.14** | 0.22 | 0.3 | 0.36 | **0.46** |
| LDP | **0.58** | 0.48 | **0.59** | 0.48 | 0 | 0 | 0.21 | **0.41** | 0.2 | 0.23 |

Table 5.37: Evaluation on Graph Embedding Methods: Correlation Distance (In Avp.)

| Method/Dataset | BCS | | CS | | ENRON | | RMS | | RMV | |
|---|---|---|---|---|---|---|---|---|---|---|
| Refer. Vector | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean |
| FEATHER | 0.41 | 0.56 | 0.4 | **0.65** | 0 | 0 | 0.25 | 0.23 | 0.25 | 0.21 |
| GL2Vec | **0.5** | 0.43 | **0.48** | 0.55 | 0.12 | 0.12 | **0.56** | **0.48** | 0.39 | **0.48** |
| Graph2Vec | 0.49 | 0.5 | 0.45 | 0.47 | **0.16** | **0.16** | 0.22 | 0.22 | **0.42** | 0.45 |
| LDP | 0.49 | **0.72** | 0.39 | 0 | 0 | 0 | 0.24 | 0.21 | 0.23 | 0.21 |

Table 5.38: Evaluation on Graph Embedding Methods: Cosine Distance (In Avp.)

| Method/Dataset | BCS | | CS | | ENRON | | RMS | | RMV | |
|---|---|---|---|---|---|---|---|---|---|---|
| Refer. Vector | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean |
| FEATHER | 0.42 | **0.69** | 0.39 | **0.65** | 0 | 0 | 0.25 | 0.23 | 0.25 | 0.2 |
| GL2Vec | 0.49 | 0.43 | 0.47 | 0.55 | 0.12 | 0.12 | **0.55** | **0.48** | 0.38 | **0.45** |
| Graph2Vec | **0.5** | 0.5 | 0.45 | 0.46 | **0.16** | **0.16** | 0.21 | 0.22 | **0.42** | **0.45** |
| LDP | 0.44 | 0.3 | **0.52** | 0.32 | 0 | 0 | 0.24 | 0.21 | 0.22 | 0.21 |

Table 5.39: Evaluation on Graph Embedding Methods: Euclidian Distance (In Avp.)

| Method/Dataset | BCS | | CS | | ENRON | | RMS | | RMV | |
|---|---|---|---|---|---|---|---|---|---|---|
| Refer. Vector | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean |
| FEATHER | 0.42 | 0.59 | 0.39 | 0.48 | 0 | 0 | 0.25 | 0.23 | 0.25 | 0.22 |
| GL2Vec | 0.55 | 0.46 | 0.41 | **0.52** | 0.09 | 0.09 | **0.48** | **0.43** | **0.45** | 0.43 |
| Graph2Vec | 0.52 | **0.68** | 0.44 | 0.42 | **0.15** | **0.15** | 0.22 | 0.3 | 0.35 | **0.46** |
| LDP | **0.57** | 0.48 | **0.59** | 0.47 | 0 | 0 | 0.21 | 0.39 | 0.2 | 0.23 |

Table 5.40: Evaluation on Graph Embedding Methods: Minkowski Distance (Parameter p is set as 1) (In Avp.)

| Method/Dataset | BCS | | CS | | ENRON | | RMS | | RMV | |
|---|---|---|---|---|---|---|---|---|---|---|
| Refer. Vector | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean | Init. | Mean |
| FEATHER | 0.47 | 0.58 | 0.42 | **0.49** | 0 | 0 | 0.24 | 0.22 | 0.25 | 0.23 |
| GL2Vec | 0.55 | 0.42 | 0.4 | 0.47 | 0.09 | 0.09 | **0.46** | **0.41** | **0.49** | 0.44 |
| Graph2Vec | 0.54 | **0.7** | 0.42 | 0.41 | **0.15** | **0.14** | 0.22 | 0.3 | 0.36 | **0.46** |
| LDP | **0.58** | 0.48 | **0.59** | 0.48 | 0 | 0 | 0.21 | **0.41** | 0.2 | 0.23 |

Table 5.41: Evaluation on Graph Embedding Methods: Run Times (In Seconds)

| Method/Dataset | BCS | CS | ENR | RMS | RMV |
|---|---|---|---|---|---|
| FEATHER | 312.92 | 100.41 | ERR | 0.22 | 1.15 |
| GL2Vec | 4419.31 | 243.95 | 0.75 | 0.51 | 7.29 |
| Graph2Vec | 218.64 | 72.77 | 0.3 | 0.21 | 0.86 |
| LDP | 439.16 | 147.23 | ERR | 0.29 | 1.58 |
| Total Graph Embed. Time | 5390.03 | 564.36 | 1.05 | 1.23 | 10.88 |
| Total Dist. Comp. Time | 45.97 | 7.34 | 0.88 | 0.37 | 0.46 |
| Total Experiment Time | 5436 | 571.7 | 1.93 | 1.6 | 11.34 |

It can be seen from the result tables that FEATHER and LDP embedding algorithms give 0 average precision for ENRON data set since they give error on ENRON data set. The execution times of all these executions are presented in Table 5.41. As it can be seen in Table 5.41, graph2vec is the fastest graph embedding algorithm followed by FEATHER and LDP in order. GL2vec algorithm is the slowest embedding algorithm among the four embedding algorithms. After the embedding step, there is distance computation and event detection step. The execution time for that step is also presented in Table 5.41 and it is much less than the graph embedding execution times. Given these execution times, the total experiment times per data set is also presented in Table 5.41.

### 5.4.3 Overview & Discussion

The results of the experiments in this section can be summarized as follows:

- The graph2vec embedding method is the fastest method and thus can produce results for all data sets except BBM. It produces 0.59 average precision value with Euclidian distance method and mean vector as reference for TGSM data set as seen in Table 5.32. The highest average precision value obtained for TGSM is 0.86 obtained by CT-IILR as seen in Table 5.30. The result of graph2vec embedding method based event detection is less than the best result obtained in previous sections, but it is still a comparable method considering its low execution time thanks to its independence from the community detection step.
- The graph2vec embedding method based event detection produces 0.12 average precision value with Chebyshev distance method and mean vector as reference for BBR data set as seen in Table 5.32. The highest average precision value obtained for BBR is 0.60 obtained by community structure based event detection methods as seen in Table 5.30. Therefore, it can be stated that graph embedding based event detection method could not produce better results for the BBR data set than the studies in previous sections.
- The overall highest precision values obtained by the graph embedding based event detection methods for BCS, CS, ENRON, RMS, RMV data sets are 0.72, 0.65, 0.19, 0.58, 0.80, respectively as seen in tables in Section 5.4.2. The overall highest precision values obtained by the community structure and communication trend based event detection methods for BCS, CS, ENRON, RMS, RMV data sets are 0.75, 0.73, 0.21, 0.39, 0.53, respectively as seen in Table 5.30. There is a considerable accuracy increases in RMS and RMV data sets results by graph embedding based event detection methods, while other data sets best results are very similar.
- The total execution times of the graph embedding based event detection methods for BCS and CS data sets are 5436 and 571 seconds, respectively as seen in Table 5.41. The total execution times of the community structure and com-

munication trend based event detection methods for BCS and CS data sets are 701 and 246 seconds, respectively as seen in Table 5.28. When the total execution times are compared, it is observed that the graph embedding based event detection methods are 2-8 times slower than the community structure and communication trend based methods. The major responsibility on this belongs to GL2vec embedding method which is 3-20 times slower than graph2vec embedding method. Considering graph2vec is also one of the best performed embedding method, if its time can be compared with the community structure and communication trend based methods, it is seen that graph2vec embedding based event detection method is faster.

- Based on the experiments in Section 5.4.2, it is observed that graph2vec and GL2vec generally have more best results than LDP and FEATHER. LDP performs better than FEATHER since it has more best results as seen in tables in Section 5.4.2.
- For ENRON data set, graph2vec is the best performed embedding algorithm for every vector distance computations.
- Mean reference vector perform better than initial reference vector in TGSM, BBR, BCS, RMV data sets, whereas initial reference vector perform better than mean reference vector in RMS data set. In CS and ENRON data sets, there is no clear difference observed.
- As seen in tables in Section 5.4.2 and in Table 5.32, there is no clear advantage of one or more distance methods over others. Only considerable observation can be FEATHER starting having best results on some of the distance methods.

# CHAPTER 6

# CONCLUSION

In this work, event detection methods that track changes in community structures, communication trends, and graph embeddings are proposed. The focus of the study is on temporal graphs corresponding to social interaction and communication among users. The changes occurring in the communication graph is considered to denote an event.

The proposed event detection algorithms are basically in 3 main approaches. The first one tracks the changes in the community structures between consecutive time steps. There are several indicators whose change in the community structure can denote an event, such as number of communities, and central nodes. Therefore, the proposed methods in this group consider the change in the number of communities and central nodes as the event indicators. The change in central nodes is defined in four different ways, which are considering all central nodes, the number of newly introduced ones, the number of central nodes not anymore existing as central nodes, and the sum of the number of newly introduced and the not anymore existing central nodes. The second main approach is to track the changes in communication trends within consecutive time steps. In this group, various indicators including the change in the inter-intra communication ratio and modularity of the graph are used. There are also proposed variations as to how these indicators are measured. Either the difference in values within consecutive time steps or the standard deviation values are examined in order to detect an event. The third main approach is to track the changes in graph structures by computing the distances between consecutive time steps graph embeddings. The time steps whose vectors distance from the previous time steps vector are more than a threshold value are marked as event time steps. In this approach, several graph embedding methods are used in graph embedding phase, and several vector distance computation methods are used in distance computation phase. All graph embedding method and vector distance computation method combinations are experimented and the results are presented.

Another novelty proposed by the study is the use of the *buckets*, which denote a set of communities of certain size range. To this aim, 8 different buckets (community size ranges) are defined. Community structure based methods are performed with different buckets (hence considering only communities of certain sizes), and the event detection performance are analyzed with respect to the buckets.

The results of the experiments show that the proposed methods are scalable and can be used on a wide range of social interaction and communication data. According to the results, the proposed methods execute faster than the graph feature based event detection methods of [1]. The proposed methods generally provide better accuracy than the method of Moriano et. al [17] for most of the data sets that they experimented on. Focusing on certain community size ranges improves both speed and accuracy of the proposed community structure based event detection methods. The graph embedding based event detection methods can further improve the best average precision values obtained by the community structure and communication trend based event detection methods for especially CDR data sets. It is observed that communication trend based methods perform better on mobile phone communication data sets while the community structure based methods perform better on social media communication data sets. As it can be inferred from this observation, mobile phone communications react more dynamically to events while the social media communications react more structurally to events.

As future work, the proposed methods can be enhanced in various aspects. As one of the research directions, the effect of using different community detection algorithms, other than dSLM and InfoMap, can be analyzed within community structure based methods. In the study, the change in the modularity is proposed as an event indicator to track in communication trend based methods. Modularity reflects the quality of the community structure of a network and it is hypothesized that a change in the quality of community structure can indicate an event. As a future study, besides modularity, other community structure quality indicators, such as the number of intra-edges, contraction, the number of inter-edges, expansion, conductance [61], modularity density [62], community score, and community fitness [63] can be used as the event indicators to track in the communication trend based event detection methods. In the presented study, it is observed that the performance of the proposed event detection methods may depend on the nature of the data set and the bucket size to be used. The proposed study can be further extended towards a meta learning such that data and the results of the conducted analysis can be used to construct a machine learning model to determine the optimal event detection method and the bucket size, as well as other parameters, for a given data set. As another future work, the proposed methods can be enhanced to work on streaming temporal graphs in near real time. For this, in addition to the use of dynamic community detection methods, such as dSLM, event detection methods should be enhanced to perform on the fly.

# REFERENCES

[1] S. Rayana and L. Akogli, “Less is more: Building selective anomaly ensembles,” *ACM Trans. Knowl. Discov. Data*, vol. 10, pp. 42:1–42:33, May 2016.

[2] O. Ozdikis, P. Karagoz, and H. Oğuztüzün, “Incremental Clustering with Vector Expansion for Online Event Detection in Microblogs,” *Social Network Analysis and Mining*, vol. 7, no. 1, p. 56, 2017.

[3] X. Zhou and L. Chen, “Event Detection over Twitter Social Media Streams,” *The VLDB Journal*, vol. 23, no. 3, pp. 381–400, 2014.

[4] G. Li and J. Jung, “Entropy-based dynamic graph embedding for anomaly detection on multiple climate time series,” *Scientific Reports*, vol. 11, p. 13819, 07 2021.

[5] J. Lv, J. Liang, and Z. Yang, “Hge2med: Heterogeneous graph embedding for multi-domain event detection,” in *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 1036–1043, 2020.

[6] T. Sakaki, M. Okazaki, and Y. Matsuo, “Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors,” in *International Conference on World Wide Web (WWW)*, pp. 851–860, 2010.

[7] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling, “TwitterStand: News in Tweets,” in *ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS)*, pp. 42–51, 2009.

[8] F. Atefeh and W. Khreich, “A Survey of Techniques for Event Detection in Twitter,” *Computational Intelligence*, vol. 31, no. 1, pp. 132–164, 2015.

[9] M. Imran, C. Castillo, F. Diaz, and S. Vieweg, “Processing social media messages in mass emergency: Survey summary,” in *Companion Proceedings of the The Web Conference 2018*, pp. 507–511, 2018.

[10] O. Ozdikis, P. Senkul, and H. Oguztuzun, "Semantic Expansion of Tweet Contents for Enhanced Event Detection in Twitter," in *International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 20–24, 2012.

[11] H. Genc and B. Yilmaz, "Text-based event detection: Deciphering date information using graph embeddings," in *Big Data Analytics and Knowledge Discovery* (C. Ordonez, I.-Y. Song, G. Anderst-Kotsis, A. M. Tjoa, and I. Khalil, eds.), (Cham), pp. 266–278, Springer International Publishing, 2019.

[12] Y. Dong, F. Pinelli, Y. Gkoufas, Z. Nabi, F. Calabrese, and N. V. Chawla, "Inferring unusual crowd events from mobile phone call detail records," *CoRR*, vol. abs/1504.03643, 2015.

[13] I. A. Karatepe and E. Zeydan, "Anomaly detection in cellular network data using big data analytics," in *European Wireless 2014; 20th European Wireless Conference*, pp. 1–5, May 2014.

[14] V. A. Traag, A. Browet, F. Calabrese, and F. Morlot, "Social event detection in massive mobile phone data using probabilistic location inference," in *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pp. 625–628, Oct 2011.

[15] R. Aktunc, I. Toroslu, and P. Karagoz, "Event detection by change tracking on community structure of temporal networks," in *ASONAM*, pp. 928–931, 08 2018.

[16] R. Aktunc, I. H. Toroslu, and P. Karagoz, *Event Detection on Communities: Tracking the Change in Community Structure within Temporal Communication Networks*, pp. 75–96. Cham: Springer International Publishing, 2020.

[17] P. Moriano, J. Finke, and Y.-Y. Ahn, "Community-based event detection in temporal networks," *Scientific Reports*, vol. 9, p. 4358, 03 2019.

[18] L. Waltman and N. J. van Eck, "A smart local moving algorithm for large-scale modularity-based community detection.," *CoRR*, vol. abs/1308.6604, 2013.

[19] R. Aktunc, I. H. Toroslu, M. Ozer, and H. Davulcu, “A dynamic modularity based community detection algorithm for large-scale networks: Dslm,” in *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, ASONAM ’15, pp. 1177–1183, 2015.

[20] V. Blondel, J. Guillaume, R. Lambiotte, and E. Mech, “Fast unfolding of communities in large networks,” *J. Stat. Mech*, p. P10008, 2008.

[21] M. Rosvall and C. T. Bergstrom, “Maps of random walks on complex networks reveal community structure,” *Proceedings of the National Academy of Sciences*, vol. 105, no. 4, pp. 1118–1123, 2008.

[22] R. Shang, W. Zhang, L. Jiao, X. Zhang, and R. Stolkin, “Dynamic immunization node model for complex networks based on community structure and threshold,” *IEEE Transactions on Cybernetics*, vol. 52, no. 3, pp. 1539–1552, 2022.

[23] R. Shang, W. Zhang, J. Zhang, J. Feng, and L. Jiao, “Local community detection based on higher-order structure and edge information,” *Physica A: Statistical Mechanics and its Applications*, vol. 587, p. 126513, 2022.

[24] R. Shang, W. Zhang, J. Zhang, L. Jiao, Y. Li, and R. Stolkin, “Local community detection algorithm based on alternating strategy of strong fusion and weak fusion,” *IEEE Transactions on Cybernetics*, pp. 1–14, 2022.

[25] G. Li, T.-H. Nguyen, and J. J. Jung, “Traffic incident detection based on dynamic graph embedding in vehicular edge computing,” *Applied Sciences*, vol. 11, no. 13, 2021.

[26] F. Cekinel and P. KARAGOZ, “Event prediction from news text using subgraph embedding and graph sequence mining,” *World Wide Web*, 02 2022.

[27] G. K. Orman, V. Labatut, and H. Cherifi, “Comparative evaluation of community detection algorithms: A topological approach,” *CoRR*, vol. abs/1206.4987, 2012.

[28] L. Tang and H. Liu, *Community Detection and Mining in Social Media*. Synthesis Lectures on Data Mining and Knowledge Discovery, Morgan and Claypool Publishers, 2010.

[29] G. Orman and V. Labatut, “A comparison of community detection algorithms on artificial networks,” in *Discovery Science*, pp. 242–256, 2009.

[30] L. Akoglu and C. Faloutsos, “Event detection in time series of mobile communication graphs,” in *Proc. of Army Science Conference*, 2010.

[31] A. Lakhina, M. Crovella, and C. Diot, “Diagnosing network-wide traffic anomalies,” in *ACM SIGCOMM Computer Communication Review*, vol. 34, pp. 219–230, 2004.

[32] S. Papadimitriou, J. Sun, and C. Faloutsos, “Streaming pattern discovery in multiple time-series,” in *Proceedings of the 31st international conference on Very large data bases*, pp. 697–708, VLDB Endowment, 2005.

[33] S. Bommakanti and S. Panda, “Events detection in temporally evolving social networks,” in *2018 IEEE International Conference on Big Knowledge (ICBK)*, pp. 235–242, 2018.

[34] T. Zhu, P. Li, L. Yu, K. Chen, and Y. Chen, “Change point detection in dynamic networks based on community identification,” *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 3, pp. 2067–2077, 2020.

[35] F. Liu, Y. Wen, D. Zhang, X. Jiang, X. Xing, and D. Meng, “Log2vec: A heterogeneous graph embedding based approach for detecting cyber threats within enterprise,” in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS ’19, (New York, NY, USA), p. 1777–1794, Association for Computing Machinery, 2019.

[36] Q. Xiao, J. Liu, Q. Wang, Z. Jiang, X. Wang, and Y. Yao, “Towards network anomaly detection using graph embedding,” in *Computational Science – ICCS 2020* (V. V. Krzhizhanovskaya, G. Závodszky, M. H. Lees, J. J. Dongarra, P. M. A. Sloot, S. Brissos, and J. Teixeira, eds.), (Cham), pp. 156–169, Springer International Publishing, 2020.

[37] A. Modell, J. Larson, M. Turcotte, and A. Bertiger, “A graph embedding approach to user behavior anomaly detection,” in *2021 IEEE International Conference on Big Data (Big Data)*, pp. 2650–2655, 2021.

[38] A. Markovitz, G. Sharir, I. Friedman, L. Zelnik-Manor, and S. Avidan, "Graph embedded pose clustering for anomaly detection," 2019.

[39] A. E. Kiouche, S. Lagraa, K. Amrouche, and H. Seba, "A simple graph embedding for anomaly detection in a stream of heterogeneous labeled graphs," *Pattern Recognition*, vol. 112, p. 107746, 2021.

[40] M. Kosan, A. Silva, S. Medya, B. Uzzi, and A. Singh, "Event detection on dynamic graphs," 2021.

[41] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, pp. 75–174, 2010.

[42] M. E. J. Newman, "Modularity and community structure in networks," in *Proceedings of the National Academy of Sciences*, vol. 103, pp. 8577–8582, 2006.

[43] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," in *COMPUTER NETWORKS AND ISDN SYSTEMS*, pp. 107–117, 1998.

[44] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, aug 2014.

[45] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," 2016.

[46] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo, "struc2vec: Learning node representations from structural identity," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, aug 2017.

[47] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[48] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," 2017.

[49] H. Chen and H. Koga, "Gl2vec: Graph embedding enriched by line graphs with edge features," in *Neural Information Processing* (T. Gedeon, K. W. Wong, and M. Lee, eds.), (Cham), pp. 3–14, Springer International Publishing, 2019.

[50] B. Rozemberczki and R. Sarkar, “Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models,” 2020.

[51] C. Cai and Y. Wang, “A simple yet effective baseline for non-attributed graph classification,” 2018.

[52] Q. V. Le and T. Mikolov, “Distributed representations of sentences and documents,” 2014.

[53] F. Harary and R. Z. Norman, “Some properties of line digraphs,” *Rendiconti del Circolo Matematico di Palermo*, vol. 9, pp. 161–168, 1960.

[54] J. R. Bray and J. T. Curtis, “An ordination of the upland forest communities of southern wisconsin,” *Ecological Monographs*, vol. 27, no. 4, pp. 325–349, 1957.

[55] A. Sanfeliu and K.-S. Fu, “A distance measure between attributed relational graphs for pattern recognition,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, no. 3, pp. 353–362, 1983.

[56] Z. Zeng, A. K. H. Tung, J. Wang, J. Feng, and L. Zhou, “Comparing stars: On approximating graph edit distance,” *Proc. VLDB Endow.*, vol. 2, p. 25–36, aug 2009.

[57] J. Diesner, T. Frantz, and K. Carley, “Communication networks from the enron email corpus “it’s always about the people. enron is no different”,” *Computational & Mathematical Organization Theory*, vol. 11, pp. 201–228, 10 2005.

[58] R. Darst, C. Granell, A. Arenas, S. Gomez, J. Saramäki, and S. Fortunato, “Detection of timescales in evolving complex systems,” *Scientific Reports*, vol. 6, 04 2016.

[59] N. Eagle, A. S. Pentland, and D. Lazer, “Inferring friendship network structure by using mobile phone data,” *Proceedings of the National Academy of Sciences*, vol. 106, no. 36, pp. 15274–15278, 2009.

[60] S. Rayana and L. Akoglu, “Less is more: Building selective anomaly ensembles with application to event detection in temporal graphs,” in *SDM*, 2015.

[61] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," 2012.

[62] M. Chen, T. Nguyen, and B. K. Szymanski, "On measuring the quality of a network community structure.," in *SocialCom*, pp. 122–127, IEEE Computer Society, 2013.

[63] S. Kaur, S. Singh, S. Kaushal, and A. Sangaiah, "Comparative analysis of quality metrics for community detection in social networks using genetic algorithm," *Neural Network World*, vol. 26, pp. 625–641, 01 2016.

**Appendix A**

# DETAILED EXPERIMENTS EXPLANATION FOCUSING PER DATA SET

Each proposed method is run with each bucket parameter for each data set and the best performed 25 results are represented in the Tables in this chapter.

Ratio column represent the ratio of the number of nodes in the bucket given in that row to the number of nodes in the A bucket. This data gives an insight on what portion of the data the processed bucket corresponds to. Each bucket contains the communities whose size are greater than or equal to a minimum threshold and less than or equal to a maximum threshold. The size range column represents these minimum and maximum thresholds for the bucket in that row. This information gives an insight on the sizes of the communities that are targeted in that execution of the method. As seen in the tables, the initial A target has 0 as minimum and 100,000 as maximum threshold values. The values for the other buckets vary depending on the size of the data set since they are computed based on F and its proportions.

For each data set, there is a section with the corresponding experiment results in it. For Enron data set, there are 7 resolutions experimented such that each resolution corresponds to number of weeks an the results for each resolution are represented in Appendix B. Similarly, 3 resolution steps with days as the time window are experimented for the TGSM data sets and the results are presented in Appendix E. The results for Boston Bombing data sets are presented in Appendix C and for Reality Mining data sets are presented in Appendix D with 1 resolution.

The aggregated overall tested methods name abbreviations and their meanings can be seen in Table A.1.

Table A.1: Abbreviations for all the methods that are experimented

| Abbreviation | Method Description |
|---|---|
| CN-NEW | Track the number of new central nodes |
| CN-SIZE | Track the number of central nodes |
| CN-NOT | Track the number of not anymore central nodes |
| CN-NOT-NEW | Track the number of not anymore + new central nodes |
| NOC | Track the number of communities |
| NOC-qx..qy-OR | OR ensemble of buckets x..y in NOC |
| NOC-qx..qy-AND | AND ensemble of buckets x..y in NOC |
| NOC-NEW-OR | OR ensemble of NOC and CN-NEW |
| NOC-NEW-AND | AND ensemble of NOC and CN-NEW |
| NOC-NOT-OR | OR ensemble of NOC and CN-NOT |
| NOC-NOT-AND | AND ensemble of NOC and CN-NOT |
| NOC-NOT-NEW-OR | OR ensemble of NOC and CN-NOT-NEW |
| NOC-NOT-NEW-AND | AND ensemble of NOC and CN-NOT-NEW |
| NOC-SIZE-OR | OR ensemble of NOC and CN-SIZE |
| NOC-SIZE-AND | AND ensemble of NOC and CN-SIZE |
| CT-IILR | Track the IILR by comparing with prev time |
| CT-IILR-BaseInfomap | Initial network communities detected by InfoMap |
| CT-IILR-Base10 | Initial network is constructed by merging initial 10 time steps |
| CT-IILR-BaseAll | Initial network is constructed by merging all time steps |
| CT-IILR-SD | Track the IILR with respect to standard deviation |
| CT-IILR-BaseInfomap-SD | Initial network communities detected by InfoMap |
| CT-IILR-Base10-SD | Initial network is constructed by merging initial 10 time steps |
| CT-IILR-BaseAll-SD | Initial network is constructed by merging all time steps |
| CT-MOD | Track the Modularity by comparing with prev time |
| CT-MOD-SD | Track the Modularity with respect to standard deviation |

# Appendix B

## DETAILED EXPERIMENTS ON ENRON DATA SETS

Table B.1: Evaluation on Enron Data Sets (Res: 1 Week)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| NOC-SIZE-AND | H1 | 0.45 | 3-10 | 0.21 |
| NOC-NEW-OR | H1 | 0.45 | 3-10 | 0.20 |
| NOC-NOT-NEW-AND | H1 | 0.45 | 3-10 | 0.20 |
| CT-MOD | A | 1.00 | 0-100000 | 0.19 |
| CN-SIZE | H1 | 0.45 | 3-10 | 0.18 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.18 |
| NOC-q1q2q4-OR | A | 1.00 | 0-100000 | 0.18 |
| NOC-q2q3q4-OR | A | 1.00 | 0-100000 | 0.18 |
| NOC-q1q3q4-OR | A | 1.00 | 0-100000 | 0.18 |
| NOC-NOT-OR | H1 | 0.45 | 3-10 | 0.18 |
| NOC | H1 | 0.45 | 3-10 | 0.17 |
| CT-IILR-SD | A | 1.00 | 0-100000 | 0.16 |
| NOC-SIZE-OR | H1 | 0.45 | 3-10 | 0.16 |
| NOC-NEW-AND | H1 | 0.45 | 3-10 | 0.15 |
| CN-NEW | H1 | 0.45 | 3-10 | 0.14 |
| CT-IILR-BaseInfomap | A | 1.00 | 0-100000 | 0.14 |
| CT-IILR-BaseInfomap-SD | A | 1.00 | 0-100000 | 0.14 |
| NOC-SIZE-OR | A | 1.00 | 0-100000 | 0.14 |
| NOC-SIZE-OR | Q1 | 0.22 | 3-6 | 0.14 |
| NOC | Q1 | 0.22 | 3-6 | 0.13 |
| CN-SIZE | A | 1.00 | 0-100000 | 0.13 |
| CN-NOT-NEW | A | 1.00 | 0-100000 | 0.13 |
| NOC-NOT-AND | H1 | 0.45 | 3-10 | 0.13 |
| CN-NOT | H1 | 0.45 | 3-10 | 0.12 |
| NOC-NOT-NEW-OR | A | 1.00 | 0-100000 | 0.12 |

Table B.2: Evaluation on Enron Data Sets (Res: 2 Weeks)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-MOD-SD | A | 1.00 | 0-100000 | 0.45 |
| CN-NOT-NEW | H2 | 0.39 | 13-27 | 0.38 |
| CT-IILR-BaseInfomap-SD | A | 1.00 | 0-100000 | 0.38 |
| CN-NOT-NEW | H1 | 0.47 | 3-12 | 0.36 |
| CT-IILR-BaseAll-SD | A | 1.00 | 0-100000 | 0.33 |
| CN-NOT | F | 0.86 | 3-27 | 0.32 |
| CN-NOT-NEW | F | 0.86 | 3-27 | 0.28 |
| CN-NOT-NEW | Q3 | 0.23 | 13-19 | 0.27 |
| CN-NOT | H1 | 0.47 | 3-12 | 0.26 |
| CN-NOT-NEW | A | 1.00 | 0-100000 | 0.26 |
| CT-IILR-SD | A | 1.00 | 0-100000 | 0.26 |
| CT-IILR-BaseInfomap | A | 1.00 | 0-100000 | 0.26 |
| NOC-NOT-NEW-OR | H1 | 0.47 | 3-12 | 0.26 |
| NOC-NOT-NEW-OR | H2 | 0.39 | 13-27 | 0.25 |
| NOC-NEW-OR | Q4 | 0.16 | 20-27 | 0.24 |
| CN-SIZE | H2 | 0.39 | 13-27 | 0.22 |
| CN-NOT | H2 | 0.39 | 13-27 | 0.22 |
| CT-MOD | A | 1.00 | 0-100000 | 0.22 |
| NOC-NOT-NEW-OR | F | 0.86 | 3-27 | 0.22 |
| NOC-NOT-NEW-OR | Q3 | 0.23 | 13-19 | 0.21 |
| CN-NOT-NEW | Q2 | 0.22 | 9-12 | 0.19 |
| NOC-q1q2q3q4-OR | A | 1.00 | 0-100000 | 0.19 |
| NOC-NOT-AND | A | 1.00 | 0-100000 | 0.19 |
| NOC-SIZE-OR | F | 0.86 | 3-27 | 0.19 |
| NOC-NOT-OR | F | 0.86 | 3-27 | 0.19 |

Table B.3: Evaluation on Enron Data Sets (Res: 3 Weeks)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-IILR-BaseInfomap-SD | A | 1.00 | 0-100000 | 0.64 |
| CT-IILR-SD | A | 1.00 | 0-100000 | 0.49 |
| CT-IILR-BaseAll-SD | A | 1.00 | 0-100000 | 0.47 |
| CN-NOT-NEW | H1 | 0.42 | 3-12 | 0.36 |
| NOC-NOT-NEW-OR | F | 0.83 | 3-24 | 0.32 |
| CN-NOT-NEW | F | 0.83 | 3-24 | 0.31 |
| CT-IILR-BaseAll | A | 1.00 | 0-100000 | 0.31 |
| NOC-q1q2-OR | A | 1.00 | 0-100000 | 0.30 |
| NOC-q1q3-OR | A | 1.00 | 0-100000 | 0.30 |
| NOC-q2q3-OR | A | 1.00 | 0-100000 | 0.30 |
| NOC-q1q4-OR | A | 1.00 | 0-100000 | 0.30 |
| NOC-q2q4-OR | A | 1.00 | 0-100000 | 0.30 |
| NOC-q3q4-OR | A | 1.00 | 0-100000 | 0.30 |
| NOC-NOT-OR | F | 0.83 | 3-24 | 0.30 |
| NOC-NOT-NEW-OR | H1 | 0.42 | 3-12 | 0.30 |
| CT-IILR | A | 1.00 | 0-100000 | 0.29 |
| CN-NOT | F | 0.83 | 3-24 | 0.28 |
| CN-NOT | H1 | 0.42 | 3-12 | 0.28 |
| NOC-NOT-AND | F | 0.83 | 3-24 | 0.28 |
| CN-NOT-NEW | Q2 | 0.23 | 10-13 | 0.27 |
| NOC-NOT-NEW-OR | A | 1.00 | 0-100000 | 0.27 |
| NOC-SIZE-OR | F | 0.83 | 3-24 | 0.27 |
| NOC | F | 0.83 | 3-24 | 0.26 |
| CT-IILR-BaseInfomap | A | 1.00 | 0-100000 | 0.26 |
| NOC-NEW-OR | A | 1.00 | 0-100000 | 0.26 |

Table B.4: Evaluation on Enron Data Sets (Res: 4 Weeks)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-IILR-BaseInfomap-SD | A | 1.00 | 0-100000 | 0.65 |
| CT-IILR-BaseAll | A | 1.00 | 0-100000 | 0.54 |
| CT-IILR-BaseAll-SD | A | 1.00 | 0-100000 | 0.54 |
| CN-NOT-NEW | Q1 | 0.20 | 3-8 | 0.52 |
| CN-NOT-NEW | F | 0.80 | 3-28 | 0.45 |
| CT-IILR-BaseInfomap | A | 1.00 | 0-100000 | 0.43 |
| NOC-NOT-OR | A | 1.00 | 0-100000 | 0.43 |
| NOC-NEW-OR | A | 1.00 | 0-100000 | 0.42 |
| CN-NOT | Q1 | 0.20 | 3-8 | 0.41 |
| NOC-NOT-NEW-OR | A | 1.00 | 0-100000 | 0.40 |
| CN-NOT-NEW | A | 1.00 | 0-100000 | 0.39 |
| CN-NOT | A | 1.00 | 0-100000 | 0.38 |
| CT-IILR-SD | A | 1.00 | 0-100000 | 0.36 |
| NOC-NOT-NEW-OR | Q1 | 0.20 | 3-8 | 0.36 |
| NOC | A | 1.00 | 0-100000 | 0.35 |
| CN-NOT | Q4 | 0.16 | 21-28 | 0.35 |
| NOC-SIZE-OR | A | 1.00 | 0-100000 | 0.35 |
| NOC-NOT-OR | Q4 | 0.16 | 21-28 | 0.35 |
| CN-NEW | A | 1.00 | 0-100000 | 0.33 |
| CT-IILR | A | 1.00 | 0-100000 | 0.33 |
| CN-SIZE | F | 0.80 | 3-28 | 0.32 |
| CN-NOT | F | 0.80 | 3-28 | 0.32 |
| CN-NOT-NEW | Q4 | 0.16 | 21-28 | 0.32 |
| NOC-NEW-OR | F | 0.80 | 3-28 | 0.32 |
| NOC-NOT-NEW-OR | F | 0.80 | 3-28 | 0.32 |

Table B.5: Evaluation on Enron Data Sets (Res: 5 Weeks)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CN-NOT | A | 1.00 | 0-100000 | 0.58 |
| CN-NOT-NEW | A | 1.00 | 0-100000 | 0.57 |
| CT-IILR-SD | A | 1.00 | 0-100000 | 0.56 |
| CT-MOD | A | 1.00 | 0-100000 | 0.52 |
| CT-IILR-BaseInfomap | A | 1.00 | 0-100000 | 0.50 |
| NOC-NOT-NEW-OR | A | 1.00 | 0-100000 | 0.48 |
| NOC-NOT-OR | Q4 | 0.16 | 22-37 | 0.48 |
| CT-IILR | A | 1.00 | 0-100000 | 0.46 |
| CN-NOT-NEW | Q2 | 0.22 | 11-14 | 0.44 |
| CT-IILR-BaseInfomap-SD | A | 1.00 | 0-100000 | 0.42 |
| NOC-NOT-OR | A | 1.00 | 0-100000 | 0.42 |
| NOC-NEW-OR | Q2 | 0.22 | 11-14 | 0.42 |
| CN-NOT-NEW | Q3 | 0.25 | 15-20 | 0.40 |
| CT-IILR-BaseAll-SD | A | 1.00 | 0-100000 | 0.40 |
| NOC-SIZE-OR | F | 0.87 | 3-37 | 0.40 |
| NOC-NOT-OR | Q2 | 0.22 | 11-14 | 0.40 |
| NOC-NOT-NEW-OR | Q2 | 0.22 | 11-14 | 0.40 |
| CN-NOT | H2 | 0.45 | 14-37 | 0.39 |
| CN-NOT-NEW | H2 | 0.45 | 14-37 | 0.38 |
| CN-SIZE | Q2 | 0.22 | 11-14 | 0.37 |
| CN-NOT | F | 0.87 | 3-37 | 0.37 |
| NOC-NEW-OR | F | 0.87 | 3-37 | 0.37 |
| NOC-SIZE-OR | Q2 | 0.22 | 11-14 | 0.37 |
| CN-NOT-NEW | F | 0.87 | 3-37 | 0.36 |
| NOC-SIZE-OR | A | 1.00 | 0-100000 | 0.36 |

Table B.6: Evaluation on Enron Data Sets (Res: 6 Weeks)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-IILR-BaseInfomap-SD | A | 1.00 | 0-100000 | 0.85 |
| CN-NOT-NEW | Q2 | 0.21 | 11-15 | 0.81 |
| CN-NOT-NEW | F | 0.73 | 3-24 | 0.78 |
| CT-IILR-BaseAll | A | 1.00 | 0-100000 | 0.77 |
| CN-NOT | H2 | 0.35 | 15-24 | 0.76 |
| CN-NOT | F | 0.73 | 3-24 | 0.74 |
| CN-NOT-NEW | H2 | 0.35 | 15-24 | 0.71 |
| CT-IILR-SD | A | 1.00 | 0-100000 | 0.68 |
| CN-NOT-NEW | Q1 | 0.21 | 3-10 | 0.66 |
| NOC-NOT-OR | Q4 | 0.10 | 22-24 | 0.64 |
| CN-NOT | Q2 | 0.21 | 11-15 | 0.63 |
| NOC-NOT-NEW-OR | Q2 | 0.21 | 11-15 | 0.63 |
| CN-NOT | Q1 | 0.21 | 3-10 | 0.61 |
| CN-NOT-NEW | Q4 | 0.10 | 22-24 | 0.61 |
| CT-IILR-BaseAll-SD | A | 1.00 | 0-100000 | 0.61 |
| NOC-NOT-AND | A | 1.00 | 0-100000 | 0.61 |
| NOC-SIZE-OR | F | 0.73 | 3-24 | 0.60 |
| CN-NOT | A | 1.00 | 0-100000 | 0.59 |
| CN-NOT-NEW | A | 1.00 | 0-100000 | 0.59 |
| NOC-NOT-NEW-OR | F | 0.73 | 3-24 | 0.59 |
| CN-SIZE | Q1 | 0.21 | 3-10 | 0.57 |
| CT-MOD | A | 1.00 | 0-100000 | 0.57 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.57 |
| NOC-q1q2q4-OR | A | 1.00 | 0-100000 | 0.57 |
| NOC-q2q3q4-OR | A | 1.00 | 0-100000 | 0.57 |

Table B.7: Evaluation on Enron Data Sets (Res: 7 Weeks)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-IILR | A | 1.00 | 0-100000 | 0.83 |
| CT-IILR-BaseAll-SD | A | 1.00 | 0-100000 | 0.82 |
| CT-IILR-BaseInfomap | A | 1.00 | 0-100000 | 0.80 |
| CT-IILR-BaseInfomap-SD | A | 1.00 | 0-100000 | 0.80 |
| CN-NOT | F | 0.81 | 3-26 | 0.76 |
| CN-NOT-NEW | F | 0.81 | 3-26 | 0.75 |
| CN-NOT-NEW | A | 1.00 | 0-100000 | 0.73 |
| CN-NOT-NEW | H2 | 0.41 | 16-26 | 0.69 |
| CN-NOT-NEW | Q3 | 0.24 | 17-22 | 0.68 |
| NOC-NEW-OR | A | 1.00 | 0-100000 | 0.65 |
| NOC-NOT-NEW-OR | A | 1.00 | 0-100000 | 0.65 |
| CN-NOT | H1 | 0.41 | 3-15 | 0.64 |
| NOC-NOT-OR | A | 1.00 | 0-100000 | 0.64 |
| NOC-NOT-AND | F | 0.81 | 3-26 | 0.64 |
| CN-NOT | Q3 | 0.24 | 17-22 | 0.63 |
| CT-IILR-BaseAll | A | 1.00 | 0-100000 | 0.62 |
| NOC-SIZE-OR | A | 1.00 | 0-100000 | 0.60 |
| NOC-NOT-OR | Q3 | 0.24 | 17-22 | 0.60 |
| CN-NOT-NEW | H1 | 0.41 | 3-15 | 0.59 |
| NOC-NEW-OR | Q3 | 0.24 | 17-22 | 0.59 |
| CT-IILR-SD | A | 1.00 | 0-100000 | 0.58 |
| NOC-NOT-NEW-OR | F | 0.81 | 3-26 | 0.58 |
| NOC-NOT-OR | F | 0.81 | 3-26 | 0.57 |
| CN-NOT | H2 | 0.41 | 16-26 | 0.56 |
| NOC-NOT-NEW-OR | Q3 | 0.24 | 17-22 | 0.56 |

# Appendix C

## DETAILED EXPERIMENTS ON BBM/BBR DATA SETS

Table C.1: Evaluation on Boston Bombing Mention Data Set

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CN-NEW | A | 0.98 | 0-100000 | 0.58 |
| NOC-NEW-OR | A | 0.98 | 0-100000 | 0.58 |
| CN-NEW | F | 0.20 | 3-16 | 0.52 |
| NOC-NEW-OR | F | 0.20 | 3-16 | 0.52 |
| CT-MOD | A | 0.98 | 0-100000 | 0.35 |
| CN-NOT | F | 0.20 | 3-16 | 0.27 |
| NOC-SIZE-OR | A | 0.98 | 0-100000 | 0.27 |
| NOC-NOT-OR | F | 0.20 | 3-16 | 0.27 |
| NOC-SIZE-AND | Q3 | 0.05 | 6-14 | 0.25 |
| NOC-SIZE-AND | H2 | 0.08 | 5-16 | 0.21 |
| NOC-SIZE-AND | Q2 | 0.06 | 4-5 | 0.21 |
| CT-MOD-SD | A | 0.98 | 0-100000 | 0.19 |
| NOC-q1q2-AND | A | 0.98 | 0-100000 | 0.19 |
| NOC-q1q3-AND | A | 0.98 | 0-100000 | 0.19 |
| NOC-q2q3-AND | A | 0.98 | 0-100000 | 0.19 |
| NOC-q1q4-AND | A | 0.98 | 0-100000 | 0.19 |
| NOC-q2q4-AND | A | 0.98 | 0-100000 | 0.19 |
| NOC-q3q4-AND | A | 0.98 | 0-100000 | 0.19 |
| NOC | A | 0.98 | 0-100000 | 0.18 |
| NOC-NEW-AND | A | 0.98 | 0-100000 | 0.18 |
| NOC-NOT-AND | A | 0.98 | 0-100000 | 0.18 |
| NOC-NOT-NEW-AND | A | 0.98 | 0-100000 | 0.18 |
| CT-IILR | A | 0.98 | 0-100000 | 0.17 |
| NOC-q1q2q3q4-AND | A | 0.98 | 0-100000 | 0.17 |
| NOC-q1q2q3-AND | A | 0.98 | 0-100000 | 0.16 |

Table C.2: Evaluation on Boston Bombing Retweet Data Set

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| NOC | A | 0.91 | 0-100000 | 0.60 |
| NOC-q1q2-AND | A | 0.91 | 0-100000 | 0.60 |
| NOC-q1q3-AND | A | 0.91 | 0-100000 | 0.60 |
| NOC-q2q3-AND | A | 0.91 | 0-100000 | 0.60 |
| NOC-q1q2q3-AND | A | 0.91 | 0-100000 | 0.60 |
| NOC-SIZE-AND | A | 0.91 | 0-100000 | 0.60 |
| NOC-NEW-AND | A | 0.91 | 0-100000 | 0.60 |
| NOC-NOT-AND | A | 0.91 | 0-100000 | 0.60 |
| NOC-NOT-NEW-AND | A | 0.91 | 0-100000 | 0.60 |
| NOC-SIZE-AND | Q2 | 0.03 | 4-5 | 0.57 |
| NOC | Q2 | 0.03 | 4-5 | 0.56 |
| NOC-SIZE-AND | F | 0.11 | 3-12 | 0.56 |
| NOC-SIZE-AND | H1 | 0.06 | 3-3 | 0.56 |
| NOC-SIZE-AND | Q1 | 0.06 | 3-3 | 0.56 |
| NOC-NEW-AND | Q2 | 0.03 | 4-5 | 0.56 |
| NOC-NOT-AND | Q2 | 0.03 | 4-5 | 0.56 |
| NOC-NOT-NEW-AND | Q2 | 0.03 | 4-5 | 0.56 |
| NOC | F | 0.11 | 3-12 | 0.55 |
| NOC | H1 | 0.06 | 3-3 | 0.55 |
| NOC | H2 | 0.05 | 4-12 | 0.55 |
| NOC | Q1 | 0.06 | 3-3 | 0.55 |
| NOC-q1q2-OR | A | 0.91 | 0-100000 | 0.55 |
| NOC-q1q3-OR | A | 0.91 | 0-100000 | 0.55 |
| NOC-q2q3-OR | A | 0.91 | 0-100000 | 0.55 |
| NOC-q1q2q3-OR | A | 0.91 | 0-100000 | 0.55 |

# Appendix D

## DETAILED EXPERIMENTS ON RM DATA SETS

Table D.1: Evaluation on Reality Mining SMS Directed Weighted Data Set

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-IILR-SD | F | 0.70 | 3-50 | 0.60 |
| CT-IILR | Q3 | 0.18 | 12-26 | 0.43 |
| CN-NEW | H1 | 0.32 | 3-9 | 0.37 |
| CN-NEW | Q1 | 0.19 | 3-6 | 0.36 |
| CN-NOT | Q4 | 0.13 | 29-50 | 0.35 |
| NOC-NOT-OR | Q4 | 0.13 | 29-50 | 0.35 |
| CN-NEW | A | 1.00 | 0-100000 | 0.34 |
| NOC-NOT-OR | Q3 | 0.18 | 12-26 | 0.32 |
| NOC-NEW-OR | H1 | 0.32 | 3-9 | 0.31 |
| NOC-q1q2q3q4-OR | A | 1.00 | 0-100000 | 0.30 |
| NOC-SIZE-OR | Q3 | 0.18 | 12-26 | 0.30 |
| NOC-NEW-OR | A | 1.00 | 0-100000 | 0.29 |
| NOC-NEW-OR | H2 | 0.38 | 10-50 | 0.29 |
| NOC-NOT-OR | H2 | 0.38 | 10-50 | 0.29 |
| NOC-NEW-OR | Q1 | 0.19 | 3-6 | 0.29 |
| CN-NEW | Q2 | 0.19 | 7-11 | 0.28 |
| NOC-NEW-AND | A | 1.00 | 0-100000 | 0.28 |
| NOC-NOT-NEW-OR | A | 1.00 | 0-100000 | 0.28 |
| CN-SIZE | A | 1.00 | 0-100000 | 0.27 |
| CN-SIZE | Q4 | 0.13 | 29-50 | 0.27 |
| CN-NOT | Q1 | 0.19 | 3-6 | 0.27 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.27 |
| NOC-q1q2q4-OR | A | 1.00 | 0-100000 | 0.27 |
| NOC-q2q3q4-OR | A | 1.00 | 0-100000 | 0.27 |
| NOC-q1q3q4-OR | A | 1.00 | 0-100000 | 0.27 |

Table D.2: Evaluation on Reality Mining SMS Directed Unweighted Data Set

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-IILR-SD | A | 1.00 | 0-100000 | 0.39 |
| CN-NEW | H1 | 0.37 | 3-9 | 0.38 |
| NOC-NEW-OR | H2 | 0.34 | 10-34 | 0.38 |
| NOC-SIZE-OR | Q3 | 0.18 | 10-15 | 0.37 |
| NOC-NOT-AND | Q3 | 0.18 | 10-15 | 0.37 |
| NOC-SIZE-OR | H2 | 0.34 | 10-34 | 0.36 |
| NOC-NEW-OR | Q3 | 0.18 | 10-15 | 0.36 |
| CN-NEW | F | 0.70 | 3-34 | 0.35 |
| NOC-NEW-OR | H1 | 0.37 | 3-9 | 0.35 |
| NOC-SIZE-OR | Q1 | 0.18 | 3-5 | 0.35 |
| CT-MOD-SD | A | 1.00 | 0-100000 | 0.34 |
| CN-SIZE | Q3 | 0.18 | 10-15 | 0.33 |
| CN-NEW | H2 | 0.34 | 10-34 | 0.33 |
| CN-NEW | Q3 | 0.18 | 10-15 | 0.33 |
| NOC-q1q2q3q4-OR | A | 1.00 | 0-100000 | 0.33 |
| NOC-SIZE-OR | H1 | 0.37 | 3-9 | 0.33 |
| NOC-NOT-OR | H1 | 0.37 | 3-9 | 0.33 |
| NOC | Q3 | 0.18 | 10-15 | 0.32 |
| NOC-NEW-OR | F | 0.70 | 3-34 | 0.32 |
| NOC-NEW-OR | Q1 | 0.18 | 3-5 | 0.31 |
| NOC-NOT-OR | Q1 | 0.18 | 3-5 | 0.31 |
| NOC-NOT-NEW-AND | Q3 | 0.18 | 10-15 | 0.31 |
| CN-SIZE | H1 | 0.37 | 3-9 | 0.30 |
| CN-NOT | H1 | 0.37 | 3-9 | 0.30 |
| NOC-SIZE-OR | F | 0.70 | 3-34 | 0.30 |

Table D.3: Evaluation on Reality Mining SMS Undirected Weighted Data Set

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-IILR-SD | F | 0.74 | 3-25 | 0.55 |
| CN-NEW | A | 1.00 | 0-100000 | 0.40 |
| NOC-NEW-OR | H1 | 0.37 | 3-7 | 0.40 |
| CN-NOT | Q3 | 0.20 | 9-14 | 0.39 |
| CN-NEW | Q3 | 0.20 | 9-14 | 0.39 |
| NOC-NOT-OR | H1 | 0.37 | 3-7 | 0.38 |
| CN-NEW | H1 | 0.37 | 3-7 | 0.37 |
| CN-SIZE | A | 1.00 | 0-100000 | 0.36 |
| CT-MOD-SD | A | 1.00 | 0-100000 | 0.36 |
| NOC-NEW-OR | A | 1.00 | 0-100000 | 0.36 |
| NOC-SIZE-OR | H1 | 0.37 | 3-7 | 0.35 |
| CN-NOT | H1 | 0.37 | 3-7 | 0.34 |
| NOC-SIZE-OR | A | 1.00 | 0-100000 | 0.34 |
| CN-NOT | A | 1.00 | 0-100000 | 0.33 |
| CT-IILR-SD | A | 1.00 | 0-100000 | 0.33 |
| NOC-NOT-OR | Q3 | 0.20 | 9-14 | 0.33 |
| NOC-NOT-OR | A | 1.00 | 0-100000 | 0.32 |
| CN-SIZE | F | 0.74 | 3-25 | 0.31 |
| CN-NEW | F | 0.74 | 3-25 | 0.31 |
| NOC-q1q2-OR | A | 1.00 | 0-100000 | 0.31 |
| NOC-q1q3-OR | A | 1.00 | 0-100000 | 0.31 |
| NOC-q2q3-OR | A | 1.00 | 0-100000 | 0.31 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.31 |
| NOC-q1q4-OR | A | 1.00 | 0-100000 | 0.31 |
| NOC-q2q4-OR | A | 1.00 | 0-100000 | 0.31 |

Table D.4: Evaluation on Reality Mining Voice Directed Weighted Data Set

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-MOD-SD | A | 1.00 | 0-100000 | 0.52 |
| CN-NOT | Q4 | 0.19 | 40-77 | 0.46 |
| NOC-NEW-AND | Q3 | 0.20 | 27-39 | 0.46 |
| CN-NEW | Q3 | 0.20 | 27-39 | 0.45 |
| CN-NEW | Q4 | 0.19 | 40-77 | 0.45 |
| CN-NEW | H1 | 0.40 | 3-25 | 0.44 |
| CN-NOT | H1 | 0.40 | 3-25 | 0.43 |
| NOC-NEW-OR | H1 | 0.40 | 3-25 | 0.42 |
| NOC-NOT-OR | H1 | 0.40 | 3-25 | 0.39 |
| NOC-NEW-OR | Q3 | 0.20 | 27-39 | 0.39 |
| NOC-NEW-OR | Q4 | 0.19 | 40-77 | 0.39 |
| NOC-NOT-OR | Q4 | 0.19 | 40-77 | 0.39 |
| NOC-NOT-OR | Q3 | 0.20 | 27-39 | 0.38 |
| NOC | Q3 | 0.20 | 27-39 | 0.36 |
| NOC-NOT-OR | F | 0.81 | 3-77 | 0.36 |
| NOC-SIZE-AND | Q3 | 0.20 | 27-39 | 0.36 |
| NOC-NOT-NEW-AND | Q3 | 0.20 | 27-39 | 0.36 |
| CN-NEW | Q2 | 0.21 | 16-26 | 0.35 |
| NOC-SIZE-OR | Q4 | 0.19 | 40-77 | 0.35 |
| CN-SIZE | Q4 | 0.19 | 40-77 | 0.34 |
| NOC | A | 1.00 | 0-100000 | 0.33 |
| CN-NOT | H2 | 0.41 | 26-77 | 0.33 |
| CN-NEW | F | 0.81 | 3-77 | 0.33 |
| NOC-NEW-OR | F | 0.81 | 3-77 | 0.33 |
| NOC-NOT-NEW-OR | F | 0.81 | 3-77 | 0.33 |

Table D.5: Evaluation on Reality Mining Voice Directed Unweighted Data Set

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| NOC-NOT-AND | Q4 | 0.20 | 36-63 | 0.53 |
| CT-MOD-SD | A | 1.00 | 0-100000 | 0.50 |
| CN-NEW | Q2 | 0.20 | 17-25 | 0.49 |
| NOC-NEW-OR | Q4 | 0.20 | 36-63 | 0.49 |
| CN-NEW | H1 | 0.42 | 3-25 | 0.48 |
| CN-SIZE | Q4 | 0.20 | 36-63 | 0.46 |
| CN-NOT | Q2 | 0.20 | 17-25 | 0.46 |
| NOC-SIZE-AND | A | 1.00 | 0-100000 | 0.46 |
| NOC-SIZE-OR | Q4 | 0.20 | 36-63 | 0.46 |
| NOC-NEW-OR | H1 | 0.42 | 3-25 | 0.45 |
| CN-NOT | Q4 | 0.20 | 36-63 | 0.43 |
| NOC-NEW-AND | A | 1.00 | 0-100000 | 0.43 |
| NOC-NOT-AND | Q2 | 0.20 | 17-25 | 0.43 |
| CN-NEW | Q4 | 0.20 | 36-63 | 0.42 |
| NOC-NEW-OR | Q2 | 0.20 | 17-25 | 0.42 |
| NOC-NOT-NEW-AND | Q4 | 0.20 | 36-63 | 0.42 |
| CN-SIZE | Q2 | 0.20 | 17-25 | 0.41 |
| CN-NEW | H2 | 0.40 | 26-63 | 0.41 |
| NOC-NOT-NEW-AND | A | 1.00 | 0-100000 | 0.41 |
| NOC-SIZE-AND | Q2 | 0.20 | 17-25 | 0.41 |
| NOC-NEW-AND | Q2 | 0.20 | 17-25 | 0.41 |
| NOC-NOT-OR | Q4 | 0.20 | 36-63 | 0.41 |
| NOC | Q4 | 0.20 | 36-63 | 0.40 |
| CN-NEW | F | 0.82 | 3-63 | 0.40 |
| CN-NEW | Q1 | 0.21 | 3-16 | 0.40 |

Table D.6: Evaluation on Reality Mining Voice Undirected Weighted Data Set

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-MOD-SD | A | 1.00 | 0-100000 | 0.46 |
| CN-SIZE | F | 0.96 | 3-60 | 0.39 |
| CN-SIZE | Q4 | 0.18 | 37-60 | 0.39 |
| CN-NEW | H1 | 0.49 | 3-23 | 0.39 |
| CN-NOT | Q3 | 0.25 | 25-36 | 0.38 |
| NOC-SIZE-OR | Q4 | 0.18 | 37-60 | 0.38 |
| NOC-NOT-NEW-AND | Q4 | 0.18 | 37-60 | 0.37 |
| NOC-NEW-AND | Q4 | 0.18 | 37-60 | 0.36 |
| CN-NOT | H2 | 0.46 | 24-60 | 0.35 |
| CN-NEW | F | 0.96 | 3-60 | 0.35 |
| CN-NEW | Q2 | 0.25 | 16-24 | 0.35 |
| NOC-SIZE-OR | F | 0.96 | 3-60 | 0.35 |
| NOC-NEW-OR | F | 0.96 | 3-60 | 0.35 |
| NOC-NEW-OR | H1 | 0.49 | 3-23 | 0.35 |
| NOC-NOT-AND | Q4 | 0.18 | 37-60 | 0.35 |
| CN-NEW | Q3 | 0.25 | 25-36 | 0.34 |
| NOC-NEW-OR | Q2 | 0.25 | 16-24 | 0.34 |
| NOC-SIZE-AND | Q4 | 0.18 | 37-60 | 0.34 |
| NOC | Q4 | 0.18 | 37-60 | 0.33 |
| CN-SIZE | A | 1.00 | 0-100000 | 0.33 |
| CN-NEW | H2 | 0.46 | 24-60 | 0.33 |
| NOC-NOT-OR | F | 0.96 | 3-60 | 0.33 |
| NOC-NEW-OR | A | 1.00 | 0-100000 | 0.32 |
| NOC-NOT-OR | H1 | 0.49 | 3-23 | 0.32 |
| NOC-NEW-OR | H2 | 0.46 | 24-60 | 0.32 |

# Appendix E

## DETAILED EXPERIMENTS ON TGSM DATA SETS

Table E.1: Evaluation on TGSM Data Set (Res: 1 Day)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-IILR | Q2 | 0.04 | 4-5 | 0.86 |
| CN-NEW | Q1 | 0.04 | 3-3 | 0.78 |
| NOC-NEW-OR | Q1 | 0.04 | 3-3 | 0.78 |
| CN-SIZE | F | 0.12 | 3-14 | 0.71 |
| CN-SIZE | Q3 | 0.03 | 6-10 | 0.71 |
| NOC-SIZE-OR | F | 0.12 | 3-14 | 0.71 |
| NOC-SIZE-OR | Q3 | 0.03 | 6-10 | 0.71 |
| NOC-SIZE-OR | H2 | 0.05 | 5-14 | 0.70 |
| NOC-NOT-OR | Q1 | 0.04 | 3-3 | 0.70 |
| NOC-SIZE-OR | H1 | 0.06 | 3-4 | 0.68 |
| NOC-SIZE-OR | Q1 | 0.04 | 3-3 | 0.68 |
| CN-SIZE | H1 | 0.06 | 3-4 | 0.67 |
| CN-SIZE | H2 | 0.05 | 5-14 | 0.67 |
| CN-NOT | Q1 | 0.04 | 3-3 | 0.66 |
| CN-SIZE | Q1 | 0.04 | 3-3 | 0.64 |
| CN-SIZE | Q4 | 0.01 | 11-14 | 0.63 |
| CN-NOT | H1 | 0.06 | 3-4 | 0.63 |
| NOC-NOT-OR | H1 | 0.06 | 3-4 | 0.63 |
| NOC-SIZE-OR | Q4 | 0.01 | 11-14 | 0.63 |
| CN-NOT | A | 1.00 | 0-100000 | 0.62 |
| CN-NEW | Q2 | 0.04 | 4-5 | 0.62 |
| NOC-NOT-OR | A | 1.00 | 0-100000 | 0.62 |
| NOC-NEW-OR | Q2 | 0.04 | 4-5 | 0.62 |
| CN-NOT | F | 0.12 | 3-14 | 0.61 |
| CN-NEW | F | 0.12 | 3-14 | 0.61 |

Table E.2: Evaluation on TGSM Data Set (Res: 2 Days)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CN-SIZE | Q1 | 0.02 | 3-3 | 0.94 |
| CT-MOD | A | 1.00 | 0-100000 | 0.94 |
| NOC-SIZE-OR | Q1 | 0.02 | 3-3 | 0.94 |
| NOC | Q3 | 0.01 | 6-9 | 0.93 |
| CN-SIZE | F | 0.05 | 3-10 | 0.93 |
| CN-SIZE | H1 | 0.03 | 3-4 | 0.93 |
| NOC-SIZE-OR | F | 0.05 | 3-10 | 0.93 |
| NOC-SIZE-OR | H1 | 0.03 | 3-4 | 0.93 |
| NOC-SIZE-OR | H2 | 0.02 | 5-10 | 0.93 |
| NOC-NEW-AND | Q3 | 0.01 | 6-9 | 0.93 |
| NOC-NOT-AND | Q3 | 0.01 | 6-9 | 0.93 |
| NOC-NOT-NEW-AND | Q3 | 0.01 | 6-9 | 0.93 |
| NOC | F | 0.05 | 3-10 | 0.92 |
| NOC | Q1 | 0.02 | 3-3 | 0.92 |
| NOC | Q2 | 0.02 | 4-5 | 0.92 |
| NOC-q1q2-OR | A | 1.00 | 0-100000 | 0.92 |
| NOC-q1q3-OR | A | 1.00 | 0-100000 | 0.92 |
| NOC-q2q3-OR | A | 1.00 | 0-100000 | 0.92 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.92 |
| NOC-q1q4-OR | A | 1.00 | 0-100000 | 0.92 |
| NOC-q2q4-OR | A | 1.00 | 0-100000 | 0.92 |
| NOC-q3q4-OR | A | 1.00 | 0-100000 | 0.92 |
| NOC-q1q2q4-OR | A | 1.00 | 0-100000 | 0.92 |
| NOC-q2q3q4-OR | A | 1.00 | 0-100000 | 0.92 |
| NOC-q1q3q4-OR | A | 1.00 | 0-100000 | 0.92 |

Table E.3: Evaluation on TGSM Data Set (Res: 3 Days)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CN-SIZE | F | 0.03 | 3-9 | 0.90 |
| CN-SIZE | Q2 | 0.01 | 4-5 | 0.90 |
| CT-MOD | A | 1.00 | 0-100000 | 0.90 |
| NOC-SIZE-OR | F | 0.03 | 3-9 | 0.90 |
| NOC-SIZE-AND | F | 0.03 | 3-9 | 0.90 |
| NOC-SIZE-AND | Q1 | 0.01 | 3-3 | 0.90 |
| NOC-SIZE-OR | Q2 | 0.01 | 4-5 | 0.90 |
| NOC | F | 0.03 | 3-9 | 0.89 |
| NOC | H1 | 0.02 | 3-4 | 0.89 |
| NOC | Q1 | 0.01 | 3-3 | 0.89 |
| NOC | Q2 | 0.01 | 4-5 | 0.89 |
| NOC-q1q2-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q1q3-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q2q3-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-NEW-AND | F | 0.03 | 3-9 | 0.89 |
| NOC-NOT-AND | F | 0.03 | 3-9 | 0.89 |
| NOC-NOT-NEW-AND | F | 0.03 | 3-9 | 0.89 |
| NOC-SIZE-AND | H1 | 0.02 | 3-4 | 0.89 |
| NOC-NEW-AND | H1 | 0.02 | 3-4 | 0.89 |
| NOC-NOT-AND | H1 | 0.02 | 3-4 | 0.89 |
| NOC-NOT-NEW-AND | H1 | 0.02 | 3-4 | 0.89 |
| NOC-NEW-AND | Q1 | 0.01 | 3-3 | 0.89 |
| NOC-NOT-NEW-AND | Q1 | 0.01 | 3-3 | 0.89 |
| NOC-SIZE-AND | Q2 | 0.01 | 4-5 | 0.89 |

Table E.4: Evaluation on BCS Data Set (Res: 1 Day)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-MOD | A | 1.00 | 0-100000 | 0.75 |
| CN-NEW | H1 | 0.20 | 3-4 | 0.67 |
| NOC-NEW-OR | H1 | 0.20 | 3-4 | 0.67 |
| NOC-SIZE-OR | Q2 | 0.11 | 4-5 | 0.67 |
| CN-NOT | Q1 | 0.13 | 3-3 | 0.66 |
| NOC-NOT-OR | Q1 | 0.13 | 3-3 | 0.66 |
| CN-NOT | H1 | 0.20 | 3-4 | 0.64 |
| CN-NEW | Q1 | 0.13 | 3-3 | 0.64 |
| NOC-NOT-OR | H1 | 0.20 | 3-4 | 0.64 |
| NOC-NEW-OR | Q1 | 0.13 | 3-3 | 0.64 |
| NOC | H2 | 0.19 | 5-27 | 0.62 |
| CN-NEW | A | 1.00 | 0-100000 | 0.62 |
| NOC-q1q2q3q4-OR | A | 1.00 | 0-100000 | 0.62 |
| NOC-NEW-OR | A | 1.00 | 0-100000 | 0.62 |
| NOC-NEW-AND | H2 | 0.19 | 5-27 | 0.62 |
| NOC-NOT-AND | H2 | 0.19 | 5-27 | 0.62 |
| NOC-NOT-NEW-AND | H2 | 0.19 | 5-27 | 0.62 |
| NOC | Q4 | 0.04 | 12-27 | 0.61 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.61 |
| NOC-q1q2q4-OR | A | 1.00 | 0-100000 | 0.61 |
| NOC-q2q3q4-OR | A | 1.00 | 0-100000 | 0.61 |
| NOC-q1q3q4-OR | A | 1.00 | 0-100000 | 0.61 |
| NOC-NEW-AND | Q4 | 0.04 | 12-27 | 0.61 |
| NOC-NOT-AND | Q4 | 0.04 | 12-27 | 0.61 |
| NOC-NOT-NEW-AND | Q4 | 0.04 | 12-27 | 0.61 |

Table E.5: Evaluation on BCS Data Set (Res: 2 Days)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-MOD | A | 1.00 | 0-100000 | 0.94 |
| NOC-SIZE-OR | F | 0.37 | 3-32 | 0.93 |
| CN-SIZE | F | 0.37 | 3-32 | 0.92 |
| NOC-q1q2-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-q1q3-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-q2q3-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-q1q4-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-q2q4-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-q3q4-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-q1q2q4-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-q2q3q4-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-q1q3q4-OR | A | 1.00 | 0-100000 | 0.91 |
| NOC-SIZE-OR | H1 | 0.21 | 3-5 | 0.91 |
| NOC-SIZE-OR | Q2 | 0.10 | 4-5 | 0.91 |
| NOC-SIZE-OR | A | 1.00 | 0-100000 | 0.90 |
| NOC-SIZE-OR | Q4 | 0.06 | 12-32 | 0.90 |
| CN-NOT | Q3 | 0.10 | 6-11 | 0.88 |
| NOC-q1q2q3q4-OR | A | 1.00 | 0-100000 | 0.88 |
| NOC-NOT-OR | Q3 | 0.10 | 6-11 | 0.88 |
| CN-NEW | Q2 | 0.10 | 4-5 | 0.87 |
| NOC-NEW-OR | Q2 | 0.10 | 4-5 | 0.87 |
| CN-NOT | F | 0.37 | 3-32 | 0.86 |
| NOC-NOT-OR | F | 0.37 | 3-32 | 0.86 |
| CN-NOT | H2 | 0.16 | 6-32 | 0.85 |

Table E.6: Evaluation on BCS Data Set (Res: 3 Days)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-MOD | A | 1.00 | 0-100000 | 0.89 |
| CN-SIZE | H2 | 0.14 | 6-29 | 0.86 |
| CN-NEW | A | 1.00 | 0-100000 | 0.86 |
| NOC-NEW-OR | A | 1.00 | 0-100000 | 0.86 |
| NOC-SIZE-OR | H2 | 0.14 | 6-29 | 0.86 |
| CN-SIZE | Q2 | 0.09 | 4-5 | 0.83 |
| CN-SIZE | Q3 | 0.09 | 6-11 | 0.83 |
| CN-NEW | H1 | 0.19 | 3-5 | 0.83 |
| NOC-NEW-OR | H1 | 0.19 | 3-5 | 0.83 |
| NOC-SIZE-OR | Q2 | 0.09 | 4-5 | 0.83 |
| NOC-SIZE-OR | Q3 | 0.09 | 6-11 | 0.83 |
| CN-SIZE | Q4 | 0.05 | 12-29 | 0.79 |
| NOC-SIZE-OR | Q4 | 0.05 | 12-29 | 0.79 |
| CN-NEW | F | 0.33 | 3-29 | 0.76 |
| NOC-NEW-OR | F | 0.33 | 3-29 | 0.76 |
| NOC-SIZE-AND | H1 | 0.19 | 3-5 | 0.75 |
| CN-NOT | A | 1.00 | 0-100000 | 0.73 |
| NOC-q1q2-OR | A | 1.00 | 0-100000 | 0.73 |
| NOC-q1q3-OR | A | 1.00 | 0-100000 | 0.73 |
| NOC-q2q3-OR | A | 1.00 | 0-100000 | 0.73 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.73 |
| NOC-q1q4-OR | A | 1.00 | 0-100000 | 0.73 |
| NOC-q2q4-OR | A | 1.00 | 0-100000 | 0.73 |
| NOC-q3q4-OR | A | 1.00 | 0-100000 | 0.73 |
| NOC-q1q2q4-OR | A | 1.00 | 0-100000 | 0.73 |

Table E.7: Evaluation on CS Data Set (Res: 1 Day)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| NOC-NOT-OR | Q2 | 0.09 | 4-5 | 0.73 |
| NOC | Q4 | 0.02 | 14-23 | 0.70 |
| CN-NOT | Q2 | 0.09 | 4-5 | 0.69 |
| CT-IILR | H1 | 0.19 | 3-4 | 0.69 |
| NOC | Q2 | 0.09 | 4-5 | 0.68 |
| NOC-NEW-OR | Q2 | 0.09 | 4-5 | 0.68 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.66 |
| NOC-q1q2q4-OR | A | 1.00 | 0-100000 | 0.66 |
| NOC-q2q3q4-OR | A | 1.00 | 0-100000 | 0.66 |
| NOC-q1q3q4-OR | A | 1.00 | 0-100000 | 0.66 |
| NOC | F | 0.32 | 3-23 | 0.65 |
| CN-NEW | Q2 | 0.09 | 4-5 | 0.65 |
| NOC-q1q2-OR | A | 1.00 | 0-100000 | 0.65 |
| NOC-q1q3-OR | A | 1.00 | 0-100000 | 0.65 |
| NOC-q2q3-OR | A | 1.00 | 0-100000 | 0.65 |
| NOC-q1q4-OR | A | 1.00 | 0-100000 | 0.65 |
| NOC-q2q4-OR | A | 1.00 | 0-100000 | 0.65 |
| NOC-q3q4-OR | A | 1.00 | 0-100000 | 0.65 |
| NOC-q1q2q3q4-OR | A | 1.00 | 0-100000 | 0.65 |
| NOC-NEW-AND | F | 0.32 | 3-23 | 0.65 |
| NOC-NOT-AND | F | 0.32 | 3-23 | 0.65 |
| NOC-NOT-NEW-AND | F | 0.32 | 3-23 | 0.65 |
| NOC | H2 | 0.13 | 5-23 | 0.64 |
| CN-NEW | H1 | 0.19 | 3-4 | 0.64 |
| NOC-NEW-OR | H1 | 0.19 | 3-4 | 0.64 |

Table E.8: Evaluation on CS Data Set (Res: 2 Days)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| NOC-SIZE-OR | F | 0.32 | 3-28 | 0.93 |
| CT-MOD | A | 1.00 | 0-100000 | 0.92 |
| NOC-SIZE-OR | H1 | 0.17 | 3-4 | 0.92 |
| NOC | H1 | 0.17 | 3-4 | 0.89 |
| NOC | Q3 | 0.08 | 6-13 | 0.89 |
| NOC-q1q2-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q1q3-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q2q3-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q1q2q3-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q1q4-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q2q4-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q3q4-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q1q2q4-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q2q3q4-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q1q3q4-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-q1q2q3q4-OR | A | 1.00 | 0-100000 | 0.89 |
| NOC-NEW-AND | Q2 | 0.09 | 4-5 | 0.89 |
| NOC-NOT-AND | Q2 | 0.09 | 4-5 | 0.89 |
| NOC-NOT-NEW-AND | Q2 | 0.09 | 4-5 | 0.89 |
| NOC-NEW-AND | Q3 | 0.08 | 6-13 | 0.89 |
| NOC-NOT-AND | Q3 | 0.08 | 6-13 | 0.89 |
| NOC-NOT-NEW-AND | Q3 | 0.08 | 6-13 | 0.89 |
| NOC | F | 0.32 | 3-28 | 0.88 |
| NOC | Q1 | 0.12 | 3-3 | 0.88 |
| CN-SIZE | F | 0.32 | 3-28 | 0.88 |

Table E.9: Evaluation on CS Data Set (Res: 3 Days)

| Event Detection Method | Bucket | Ratio | Size Range | Av. Prec. |
|---|---|---|---|---|
| CT-MOD | A | 1.00 | 0-100000 | 0.90 |
| CN-NEW | Q1 | 0.12 | 3-3 | 0.88 |
| NOC-NEW-OR | Q1 | 0.12 | 3-3 | 0.88 |
| CN-NEW | Q3 | 0.08 | 6-13 | 0.86 |
| NOC-NEW-OR | Q3 | 0.08 | 6-13 | 0.86 |
| NOC-q1q2q3q4-AND | A | 1.00 | 0-100000 | 0.85 |
| NOC-q1q2-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-q1q3-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-q2q3-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-q1q2q3-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-q1q4-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-q2q4-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-q3q4-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-q1q2q4-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-q2q3q4-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-q1q3q4-AND | A | 1.00 | 0-100000 | 0.80 |
| NOC-SIZE-AND | Q1 | 0.12 | 3-3 | 0.80 |
| NOC | F | 0.31 | 3-29 | 0.79 |
| NOC | H1 | 0.17 | 3-4 | 0.79 |
| NOC | Q1 | 0.12 | 3-3 | 0.79 |
| CN-NEW | H2 | 0.14 | 5-29 | 0.79 |
| CN-NEW | Q2 | 0.09 | 4-5 | 0.79 |
| CN-NEW | Q4 | 0.03 | 14-29 | 0.79 |
| NOC-q1q2-OR | A | 1.00 | 0-100000 | 0.79 |
| NOC-q1q3-OR | A | 1.00 | 0-100000 | 0.79 |

# CURRICULUM VITAE

## PERSONAL INFORMATION

**Surname, Name:** Aktunc, Riza
**Nationality:** Turkish (TC)
**Date and Place of Birth:** 1989, Kayseri

## EDUCATION

| Degree | Institution | Year of Graduation |
|---|---|---|
| M.S. | METU Computer Engineering | 2015 |
| B.S. | Bilkent Computer Science | 2011 |
| High School | Sivas Science High School | 2006 |

## PROFESSIONAL EXPERIENCE

| Year | Place | Enrollment |
|---|---|---|
| 2022-Current | Microsoft | Senior Software Engineer |
| 2020-2022 | Tork (Microsoft Vendor) | Senior Software Engineer |
| 2018-2020 | Bayzat | Senior Software Engineer |
| 2011-2018 | TUBITAK | Software Engineer |

## PUBLICATIONS

1. R. Aktunc, P. Karagoz and I. H. Toroslu, "Event Detection via Tracking the Change in Community Structure and Communication Trends," in IEEE Access, vol. 10, pp. 109712-109728, 2022.

2. R. Aktunc, I. H. Toroslu, and P. Karagoz, "Event Detection on Communities: Tracking the Change in Community Structure within Temporal Communication Networks", pp. 75–96. Cham: Springer International Publishing, 2020.
3. R. Aktunc, I. Toroslu, and P. Karagoz, "Event detection by change tracking on community structure of temporal networks" in ASONAM, pp. 928–931, 08 2018.
4. R. Aktunc, I. H. Toroslu, M. Ozer, and H. Davulcu, "A dynamic modularity based community detection algorithm for large-scale networks: Dslm" in Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, ASONAM '15, pp. 1177–1183, 2015.
5. Selma Suloglu, Riza Aktunc, and Mustafa Yucefaydalı, "Verification of variable service orchestrations using model checking" in Proceedings of the 2013 International Workshop on Quality Assurance for Service-based Applications (QASBA 2013). Association for Computing Machinery, New York, NY, USA, 5–8, 2013.