Multi-SPMiner: A Deep Learning Framework for Multi-Graph Frequent Pattern Mining with Application to spatiotemporal Graphs

Assaad Oussama Zeghina, Aurelie Leborgne, Florence Le Ber, Antoine Vacavant

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 225 (2023) 1094–1103

Procedia Computer Science

www.elsevier.com/locate/procedia

27th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2023)

Multi-SPMiner: A Deep Learning Framework for Multi-Graph Frequent Pattern Mining with Application to spatiotemporal Graphs

Assaad Zeghina[a,*], Aurélie Leborgne[a], Florence Le Ber[a], Antoine Vacavant[b]

[a] *Université de Strasbourg, CNRS, ENGEES, ICube UMR 7357, F67000 Strasbourg*
[b] *Universite Clermont Auvergne, CNRS, SIGMA Clermont, Institut Pascal*

Abstract

Mining frequent patterns in multigraphs is a challenging task in graph analysis with numerous real-world applications. This paper introduces a novel framework for frequent pattern mining on multi-graphs using the multi-SPMiner method. The approach is inspired by SPMiner, which was the first approach to employ deep learning in graph motif mining tasks. Multi-SPMiner builds on this foundation and focuses on the extraction of frequent motifs in single multi-graphs, specifically spatiotemporal graphs. Multi-SPMiner employs a two-step approach to extract the most frequent motifs in a graph with a high support value. In the first step, it embeds the nodes into an embedding order space, and in the second step, it performs a walk in the space to obtain the frequent motifs by iteratively growing the motif starting from a single node. The results obtained highlight the effectiveness of the proposed approach in identifying frequent motifs in single multigraphs, which is a crucial task in many real-world applications. Moreover, we demonstrate that our method is a generalization of SPMiner by testing it on single connection graphs.

Keywords: Multigraph; Graph Neural Network; Frequent Motif Mining; Spatiotemporal Graphs; Graph Mining

1. Introduction

Graphs are a common representation for data in various domains such as social networks, biology, and transportation systems [18]. With the increasing availability and diversity of graph data, there is a growing need for efficient algorithms that can extract useful patterns and insights from these graphs. Frequent motif, pattern, or subgraph mining (in the remainder of this article, the three terms will be used interchangeably) is a process in graph analysis that focuses on identifying and extracting recurring subgraphs within a large graph or across multiple graphs [9]. These frequent subgraphs represent common structural patterns that offer insights into the underlying relationships, properties, and interactions within the data. Frequent motif mining has broad applications across various fields, such as biology, social

* Corresponding author. Tel.: +33 758 89 09 66
E-mail address: assaad-oussama.zeghina@etu.unistra.fr

1877-0509
10.1016/j.procs.2023.10.097

science, and chemistry [1, 6]. It enables the discovery of hidden structures and relationships that may not be apparent through conventional analysis techniques. However, frequent subgraph mining is highly computationally complex [9]. Traditionally, the approach to motif mining has been to enumerate all possible motifs of size up to k and count their appearances in a given graph. This approach is problematic since the number of motifs increases super-exponentially with their size, and counting the number of occurrences of a single motif in the target graph is an NP-hard problem. Further, the exploration and development of methods specifically designed for frequent motif mining in multigraphs have been relatively limited. The unique characteristics and complexities associated with multigraphs present additional challenges. Consequently, the availability of effective methods tailored for multigraphs remains scarce, further highlighting the need for research and innovation in this area. In recent years, deep learning has witnessed remarkable advancements in graph deep learning [18]. These advancements have also extended to addressing combinatorially hard graph problems [14]. Notably, Ying et al. introduced a groundbreaking approach called SPMiner [17], which represents the first neural method for mining frequent motifs in graphs. SPMiner is a comprehensive framework that leverages graph representation learning techniques to identify frequent motifs within large target graphs. This process involves iteratively expanding a seed node by adding adjacent nodes, randomly sampled, until frequent motifs are discovered.

This paper addresses the limited applicability of SPMiner, which has primarily been used for frequent motif mining in simple single-connection graphs. We aim to extend the utilization of SPMiner to labeled and directed multigraphs, particularly focusing on spatiotemporal graphs. By adapting SPMiner to handle the complexities and unique characteristics of labeled multigraphs, our objective is to enable efficient and effective frequent motif mining in spatiotemporal graph data. The proposed technique has been evaluated on synthetic spatiotemporal graphs represented as multi-graphs, incorporating spatial, spatiotemporal, and filiation relationships among nodes. Results show that our extension enhances the capabilities of SPMiner, opening up new possibilities for new research and applications for mining frequent motifs in diverse graph structures.

The rest of this paper is structured as follows: in Section 2, we establish the context of the work. In section 3 we delve into the theoretical foundations. Section 4 is dedicated to outlining the method and implementation details. Section 5, we present the data generator and the test results, accompanied by a thorough discussion. Finally, Section 6 concludes the paper and offers insights into potential future research directions.

2. Related work

Multigraphs allow for the modeling of complex systems and relationships, capturing diverse interactions between entities. For instance, multigraphs in social network analysis capture diverse relationship types, enabling a more realistic depiction of interpersonal connections. This enhances our understanding of real-world phenomena, making them important for frequent motif mining research.

Adaptation of classic single connection-graph algorithms to multigraphs. Existing algorithms for frequent motif mining, such as SIGRAM, GRAMI, and MuGram, have primarily focused on single relational graphs, with limited consideration for their applicability to multigraphs [11, 5, 8]. However, recent efforts have explored the feasibility of adapting these algorithms to multigraph motif mining. SIGRAM identifies frequent vertices and extends patterns to find larger subgraphs, while GRAMI addresses memory efficiency through minimum image-based metric support. However, adapting these algorithms to multigraphs has posed significant challenges. MuGram was proposed as a solution, utilizing a depth-first search to find frequent subgraphs from frequent edges, but the extraction of motifs remains computationally expensive and memory-intensive. Thus, there is a need for further exploration and development of algorithms specifically tailored for efficient frequent motif mining in multigraphs.

Multigraph frequent motif mining. There are limited methods explicitly designed for frequent motif mining in multigraphs [8]. Nonetheless, several approaches have been proposed for mining tasks in more complex graph structures, such as attributed graphs[2] or heterogeneous information networks [15]. Additionally, techniques for mining networks with heterogeneous information, such as HINMINE [10] while demonstrating a certain degree of effectiveness, still exhibit certain limitations. It may not fully exploit the distinctive characteristics and complexities inherent in multigraphs (depending on the data type), such as the ability to handle multiple edge types and capture intricate relationship patterns. In another instance, meta-graph-based mining [7], can be applicable to multi-graphs depending on

the problem and data specificity. But the method's adaptability to diverse multigraph structures and varying relationship types is limited by its dependency on predefined meta-graphs. Moreover, the method faces scalability challenges when handling large-scale multigraph datasets, which can impact its practical applicability and efficiency in real-world scenarios.

Neural approaches. The primary challenge in graph pattern mining is the exponentially large candidate set that must be examined, which scales poorly with increasing subgraph size due to the combinatorial growth of the sample space. As a result, the performance of these approaches significantly deteriorates when working with larger motifs, which justifies considering deep approaches. Rex Ying et al. [17] presented SPMiner as the first neural approach for frequent motif mining in large target graphs. The framework operates in two main steps: first, it uses a graph neural network to embed node-anchored neighborhoods of input graphs into an order embedding space. It then searches for frequent motifs by analyzing walks in the embedding space.

Despite the demonstrated effectiveness of SPMiner in extracting motifs from simple graphs, its application to other graph formats, particularly multigraphs, remains unexplored. This underscores the significance of this study, which aims to adapt and extend the capabilities of SPMiner to accommodate the mining of motifs in multigraphs.

3. Definitions

3.1. Basics

Definition 1 (Multigraph) *A multigraph $G = (V, E, l_V, l_E, L_V, L_E)$ is a data structure composed of a set V of vertices or nodes, a set of edges $E \subseteq V \times V$, two label setw L_V, L_E, and two labeling functions, $l_V : V \to 2^{L_V}$, that associates a set of labels to vertices, and $l_E : E \to 2^{L_E}$, that associates a set of labels to edges.*

Definition 2 (Subgraph) *A subgraph of a multigraph $G = (V, E, l_V, l_E, L_V, L_E)$ is a multigraph $G' = (V', E', l_{V'}, l_{E'}, L_V, L_E)$ such that $V' \subseteq V$, $E' \subseteq E$, and $\forall v \in V', l_{V'}(v) \subseteq l_V(v)$ and $\forall e \in E', l_{E'}(e) \subseteq l_E(e)$. Two type of subgraphs can be defined:*
G' is a node-induced subgraph if and only if: $V' \subseteq V$, $E' = \{(u, v) \in E \mid u \in V', v \in V'\}$. G' is an edge-induced subgraph if and only if $E' \subseteq E$, and $V' = \{v \in V \mid \exists u \in V \text{ and } (u, v) \in E'\}$.

Definition 3 (Isomorphism) *$G_1 = (V_1, E_1, l_{V_1}, l_{E_1}, L_V, L_E)$ and $G_2 = (V_2, E_2, l_{V_2}, l_{E_2}, L_V, L_E)$ are isomorphic if there exists a bijection $f : V_1 \to V_2$ such that $(u, v) \in E_1$ if and only if $(f(u), f(v)) \in E_2$, $l_{V_1}(u) = l_{V_2}(f(u))$, $l_{V_1}(v) = l_{V_2}(f(v))$ and $l_{E_1}(u, v) = l_{E_2}(f(u), f(v))$. The bijection is a one-to-one correspondence between the nodes of one graph and another, ensuring that all edges and labels are preserved.*

Definition 4 (Subgraph isomorphism) *Subgraph isomorphism is the problem of determining whether a given graph G contains another graph H as a subgraph. Given $G = (V_G, E_G, l_{V_G}, l_{E_G}, L_V, L_E)$ and $H = (V_H, E_H, l_{V_H}, l_{E_H}, L_V, L_E)$, we say that H is isomorphic to a subgraph of G, denoted as $H \subseteq G$, if there exists a one-to-one mapping $\phi : V_H \to V_G$ such that $(u, v) \in E_H$ if and only if $(\phi(u), \phi(v)) \in E_G$, $l_{V_H}(u) \subseteq l_{V_G}(\phi(u))$, $l_{V_H}(v) \subseteq l_{V_G}(\phi(v))$ and $l_{E_H}(u, v) \subseteq l_{E_G}(\phi(u), \phi(v))$.*

3.2. Spatiotemporal Graphs

Spatiotemporal Graphs (st-graphs) consider a set of temporal entities and their relations, including general semantic relations (spatial, correlation, etc., mainly spatial relations in our context) and filiation relations [4]. The model is limited to the restriction that a relation links two entities at the same time instance or at two consecutive time instances (for tracking entities over time), with at most one semantic relation and one filiation relation between two entities. However, the model can be generalized to accept a limited number of semantic relations.

In this work, we follow the definition of [12] where an st-graph is defined on a time domain, $\mathcal{T} = \{t_1, t_2, \dots t_n\}$, where t_i represents a time instance of a given granularity and $t_i < t_{i+1}$ for all $i \in [1, n]$. $\Delta = \{e_1, e_2, \dots e_m\}$ is a set of entities. We also introduce Σ, a set of spatial relations, and Φ, a set of filiation relations.

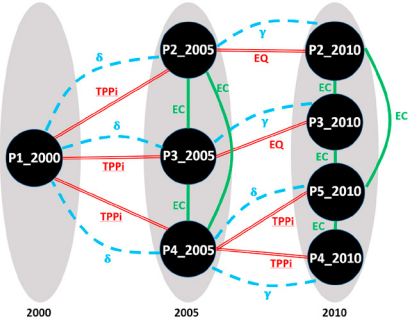

Fig. 1. A representation of a st-graph: red lines represent spatiotemporal relations, green lines represent spatial relations, and blue lines represent filiation relations [12]. In this illustration, each node is labeled by the name of the entity it represents and the time stamp, since the same entity can change properties but keep the same identity.

Definition 5 *A spatiotemporal graph $\mathcal{G}$ is defined as a multigraph $G = (V, E, l_V, l_E, L_V, L_E)$, where the vertex labels are $L_V = \Delta \times \mathcal{T}$ and the edge labels are $L_E = \Sigma \cup \Phi$. with the constraint that the edges only connect vertices that have either identical or successive values in $\mathcal{T}$.*

This model can be seen as the union of three subgraphs (as shown in figure 1) in which entities are grouped according to time instances; each node is labeled with an entity and a time stamp; an edge is labeled with a spatial relation or both a spatiotemporal and a filiation relation:

- The subgraph of spatial relations represents the spatial interactions between entities at a given time t_i. In this work spatial relations come from the Region Connection Calculus (RCC8) theory[3].
- The subgraph of spatiotemporal relations represents the spatiotemporal interactions between entities at two successive time instances. In this work spatiotemporal relations come from the RCC8 theory.
- The subgraph of filiation relations represents the transmission of identity between entities at different times. Two types of filiation relations are considered: continuation γ and derivation δ. A continuation relation between (e_i, t_l) and (e_j, t_{l+1}) means that e_i and e_j have the same identity; a derivation relation means that e_j contains a part of e_i identity.

3.3. Graph convolutional and multiGraph convolution :

Graph convolution networks (GCNs) are neural network architectures specifically designed to operate on graph-structured data, enabling effective learning and representation of node-level and graph-level features through the aggregation of information from neighboring nodes. For a given graph G, $\mathbf{X} \in \mathbb{R}^{n \times d}$ be the node feature matrix, where d is the number of features for n nodes. Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be the adjacency matrix of the graph G. The first-order graph convolutional layer computes the hidden representation $\mathbf{H} \in \mathbb{R}^n \times d'$ (d' is the feature size of the resulting node embedding) as follows:

$$\mathbf{H}^{(l+1)} = \sigma\left(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right), \tag{1}$$

where $\mathbf{H}^{(l)}$ represents the lth layer of the GCN, $\tilde{\mathbf{D}}$ is the degree matrix of $\tilde{\mathbf{A}}$, and $\mathbf{W}^{(l)}$ is the weight matrix of the lth layer. The function σ is typically a non-linear activation function such as ReLU.

Multigraph convolutional networks (MGCN) are a generalization of GCNs to handle multiple graphs. The multi-graph convolutional layer operates on each graph individually, and the results are combined to produce a final output.

4. Proposed Method

In this section, we present our approach for frequent motif mining in multigraphs, Multi-SPMiner. This approach builds upon the foundations of the SPMiner method, which is mainly used for frequent motif mining for undirected

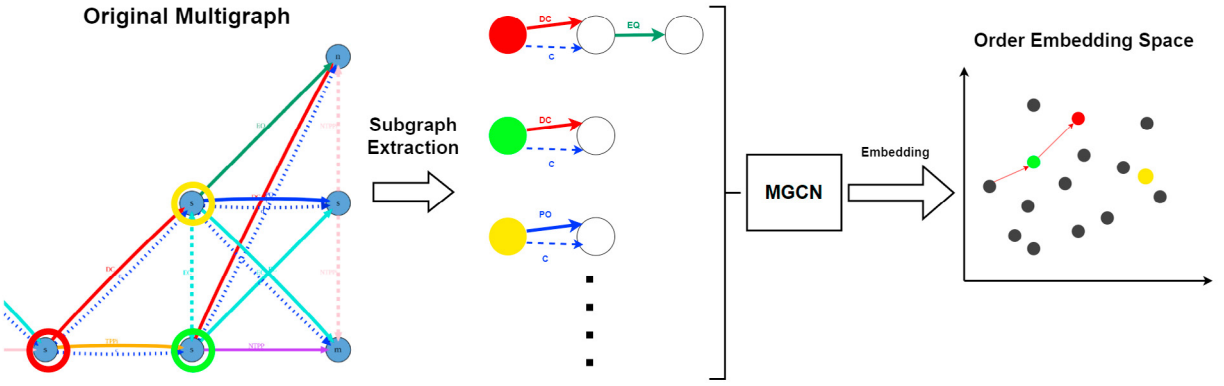

Fig. 2. The Multi-SPMiner framework differntlmy than the base SPMiner model uses a MGCN to laverage the the multigraph relations to calulate the neighborhoods, while relaying on the same order embedding principle.

unlabeled graphs. In contrast to the base method, where the authors focus on identifying frequent subgraph patterns that appear most frequently in a given dataset, we aim to extract the frequent motifs in single multigraphs. We focus on the case of finding all the node-anchored motifs with a support (or number of frequencies) higher than a value s. Similar to SPMiner, Multi-SPMiner decomposes the graph or the graph dataset into many node-anchored neighborhoods, then utilizes a deep learning model specifically designed for multigraphs (MGCN) to effectively map each neighborhood into a point in the embedding space. The MGCN ensures the preservation of the order embedding property: if neighborhood A is a subgraph of neighborhood B then A is embedded to the lower left of B. In the second step, Multi-SPMiner starts with an empty graph and iteratively adds nodes and edges to it to find frequent motifs as demonstrated in figure 2. In the next subsection, a detailed description of both steps is given.

4.1. The embedding phase

In this phase, the graph is first decomposed into K-hop neighborhoods $\mathcal{G}$ anchored at each node v, where K is set to a value superior to the max size of the motif that can be found in the graph (for the case of a motif node size equal to 10, K is set to 12), as demonstrated in figure 2.

The neighborhoods are then encoded using a multigraph convolutional network into an order embedding space. We rely on the definition presented in [16] and [17] of order embedding, which is a technique for representation learning that utilizes the geometric relations of embeddings to model a partial ordering structure. It aims to preserve the relative order or arrangement of the elements, enhancing the learning and understanding of structured or sequential data. In our specific context, it is well-suited for modeling subgraph relations.

Given the transitive property of the order embedding, we define a partial order $\leq$ on the set of all encoded graphs $\mathcal{G}$. Given two graphs $A, B \in \mathcal{G}$, we say $A \leq B$ if graph A is isomorphic to a subgraph of B. In order to enforce the order embedding constraint, we assume the existence of an embedding function $\phi : \mathcal{G} \mapsto \mathbb{R}^n$ that maps graphs to vectors such that $A \leq B$ if and only if $\phi(A) \leq \phi(B)$ elementwise. This suggests that in the 2D plane example, the embedding $\phi(A)$ is positioned to the "lower left" of the embedding $\phi(B)$ (following our definition of the loss function), as illustrated in Figure 2 by the green and red node-anchored neighborhoods.

The embedding function ϕ is learned using a multigraph convolution network that aims to preserve the order in the embedding space. The training is performed by creating a set of positive P (the pairs (A, B) where A is a subgraph of B) and negative N (the pairs (A', B') where A' is not a subgraph of B'), and then by optimizing for the max-margin loss defined as follow:

$$\sum_{(A,B)\in P} E(A, B) + \sum_{(A',B')\in N} \max\left(0, \alpha - E\left(A', B'\right)\right) \tag{2}$$

Where α is a margin hyperparameter and E is the order embedding penalty defined as:

$$E(A, B) = \| \max(0, \phi(A) - \phi(B))\|^2. \tag{3}$$

The network is trained by generating positive and negative training instances and minimizing the loss function; a random subgraph is sampled as the target neighborhood, and positive examples are generated by sampling a smaller subgraph from the target neighborhood. Negative examples are generated by randomly sampling a different subgraph.

Our GNN is constructed with multiple layers of multigraph convolutional operations with multiple skip layers to extract structural attributes of neighborhoods of varying sizes. MGCN is designed to operate on graph-structured data with multiple edges between nodes. Each node in the multigraph is represented as a feature vector, and multiple filters (*i.e.*, convolutional kernels) are applied to these features to extract relevant information. These filters operate on different types of edges (*e.g.*, spatial, temporal, filiation), allowing the model to capture and leverage the different relationships present in the graph. The embeddings at layer l capture the structural properties of the neighborhood up to a certain distance, which is determined by the number of layers in the graph convolutional network. In the context of subgraph isomorphism tasks, this property is particularly useful as it allows the model to capture and compare local structural similarities between subgraphs of varying sizes and topologies.

4.2. The search phase

The approach aims to find frequently occurring node-anchored motifs in a given graph. This is done using the embeddings obtained by the MGCN in the previous step that maps node-anchored subgraphs in order. The search procedure involves iteratively adding nodes to generate frequent motifs.

In order to identify frequently appearing motifs in a given graph, a simple search procedure to directly find frequent motifs is difficult due to the exponential number of possible motifs. Thus, an iterative search procedure that grows the motif is recommended, starting from a trivial seed graph of size 1 by randomly sampling a seed node from the dataset. The next graph is generated by adding an adjacent node in the graph and its corresponding edges. Multiple seed nodes are sampled to attain a robust estimate, and the resulting motifs of a given size are selected based on the number of times they were encountered during the walk. The added node at each iteration is selected so that the resulting new graph G_{k-1} of size k minimize the total margin $m\,(G_{k-1})$ where m is defined as:

$$m(G) = \sum_{N \in \mathcal{N}} \| \max\,(0, \phi(G) - \phi(N))\,\|^2 \tag{4}$$

where : $\mathcal{N}$ is the set of all neighborhood graphs of all nodes $v \in G$ and ϕ is the order embedding function obtained by the MGCN in the previous step.

In contrast to the base model, SPMiner, which imposes constraints on the search space to manage the sizes of the graph sets, our approach considers all nodes during the addition process, given that the embedding space is limited to one graph at a time. For G' chosen by adding 1 adjacent node to G_i (the current subgraph), the greedy approximation is a step-wise minimization. Multi-SPMiner iteratively grows the motif by adding an adjacent node following a monotonic walk in the order embedding space, indicated by the red arrows in figure 2. During the testing phase, a range of values for k is examined to extract motifs of different sizes according to specific requirements. The Multi-SPminer approach is summarized in Algorithm 1.

5. Experimentation

5.1. Data generation

In order to train the multigraph convolutional network and test the performance of the framework, we make use of a synthetic spatiotemporal graph generator [12]. The generator exhibits patterns similar to those found in real data. These motifs, or st-subgraphs, should appear frequently above a specified threshold specified during the generation process. To achieve this, a st-graph G is randomly generated using the Poisson distribution, and a set of source patterns is used as a basis. These patterns are modified and embedded in G, which itself is a st-graph. Additionally, the nodes in G are randomly allocated to successive time instances. A total of 22 experiments were conducted, utilizing the data

Algorithm 1: Multi-SPMiner

Input: Graph G, Max motif size K, Margin hyperparameter α
Output: Frequent motifs

`EmbeddingPhase(`G, K, α`);`

`SearchPhase(`G, K, s`);`

Function `EmbeddingPhase(`*Graph G, Max motif size K, Margin hyperparameter* α`):`

Decompose G into K-hop neighborhoods anchored at each node;
Encode neighborhoods using a multigraph convolutional network into an order embedding space by learning the embedding function ϕ with max-margin loss;

Function `SearchPhase(`*Graph G, Max motif size K, Support s*`):`

Initialize an empty set of frequent motifs;
for k *in* 1 *to* K **do**
  for i *in* 1 *to* N **do**
    Randomly sample a seed node v from G;
    Initialize a subgraph G_i with v;
    while *subgraph* G_i *size is less than* k **do**
      Add an adjacent node that minimizes the total margin to G_i to generate G_{i+1};
    end
    Add G_{i+1} to the set of frequent motifs if the number of times encountered during the walk $\geq s$;
  end
end
return *Set of frequent motifs*;

generator for each experiment. The features employed to generate the data for each experiment are outlined in the subsequent section. The graph generator and the Multi-SPMiner source code are available in [1].

5.2. Results

In this section, we present the testing protocol and results obtained for the proposed framework, Multi-SPMiner. The baseline model, SPMiner, focused on extracting frequent motifs in a set of graphs. However, in this work, we focus on the identification of frequent motifs within single multigraphs, with a particular emphasis on spatiotemporal graphs as defined in Section 3.2. The change in the data on which we base our approach also alters the testing method. While the baseline model focuses on the accuracy of extracting the top-N most frequent motifs in a graph dataset, we emphasize the accuracy of the extracted frequent motifs integrated into the generation process of a single graph. In this section, the test results will be presented as the mean accuracy obtained for each test, and each test was repeated 10 times.

Table 1 presents all the different experiments performed, where in each we generated 200 graphs with the same properties for training the MGCN and 10 graphs for testing. In the first test, as exemplified by the first row of the table, spatiotemporal graphs were generated with 200 nodes, consisting of 40 nodes per temporality. The distribution of spatial, spatiotemporal, and filiation relations followed mean values of 5, 5, and 2 per node, respectively. Within each graph, a total of 16 motifs of size 5 and 13 motifs of size 6 were inserted. The tests focused on small to moderate-sized frequent motifs ranging from size 4 to 10. The remaining experiments depicted in the table adhere to the identical

[1] https://aurelieleborgne.github.io/webpage_MoS-T/

Table 1. Obtained Test result for our framework Multi-SPMiner with different parameters. **Number of graph nodes**: the number of nodes in a single graph, **Number of nodes by temporality**: the maximum number of node for each timestamp in the graph, **Number of relations**: a list defining the maximum number of links (spatial, spatiotemporal, filliation) in the motif

Number of graph nodes	Number of nodes by temporality	Number of relations	Number of motifs	Ratio of obtained motifs	Precision	Recall
200	40	[5,5,2]	16 of size 5, 13 of size 6	0.632 ± 0.03	0.59 ± 0.03	0.68 ± 0.02
1000	40	[5,5,2]	60 of size 5, 50 of size 6	0.651 ± 0.02	0.63 ± 0.03	0.67 ± 0.01
1000	40	[5,5,2]	37 of size 8, 30 of size 10	0.664 ± 0.03	0.63 ± 0.01	0.70 ± 0.02
1000	40	[5,5,2]	90 of size 5, 75 of size 6	0.612 ± 0.01	0.59 ± 0.02	0.63 ± 0.01
1000	40	[5,5,2]	56 of size 8, 45 of size 10	0.643 ± 0.02	0.62 ± 0.01	0.64 ± 0.02
1000	40	[5,5,2]	120 of size 5, 100 of size 6	0.604 ± 0.02	0.59 ± 0.01	0.60 ± 0.02
1000	40	[5,5,2]	75 of size 8, 60 of size 10	0.631 ± 0.03	0.61 ± 0.02	0.64 ± 0.02
1000	40	[5,5,2]	150 of size 5, 125 of size 6	0.592 ± 0.02	0.58 ± 0.02	0.61 ± 0.01
1000	40	[5,5,2]	93 of size 8, 75 of size 10	0.664 ± 0.01	0.64 ± 0.02	0.67 ± 0.01
1500	40	[5,5,2]	120 of size 5, 100 of size 6	0.623 ± 0.01	0.62 ± 0.01	0.62 ± 0.01
1000	40	[2,2,2]	150 of size 4, 100 of size 6	0.634 ± 0.01	0.62 ± 0.01	0.64 ± 0.01
1000	40	[2,2,2]	188 of size 4, 125 of size 6	0.661 ± 0.01	0.66 ± 0.01	0.67 ± 0.01
1000	40	[5,5,2]	40 of size 5	0.683 ± 0.04	0.66 ± 0.03	0.68 ± 0.02
2000	40	[5,5,2]	10 motifs of 4,5,6,8,10 each	0.647 ± 0.03	0.62 ± 0.04	0.64 ± 0.03

Table 2. Comparison of Results for Different Motif Sizes in Training and Testing Experiments.

Number of nodes	Trainig graphs motif sizes	Testing graph motif sizes	Ratio
1000	40 of size 8	40 of size 6	0.653
1000	40 of size 10	40 of size 6	0.642
1000	40 of size 4	40 of size 7	0.662
1000	40 of size 6	40 of size 9	0.648

testing protocol as described earlier. To manage the exponential number of combinations, 14 tests were conducted, varying parameters such as graph size, motif size, and the number of motifs. This allowed us to examine the influence of these parameters on the ratio of obtained motifs.

According to the results presented in Table 1, we can observe that the proposed framework, Multi-SPMiner, achieved stable performance in extracting frequent motifs from spatiotemporal graphs. The accuracy ranged from 60% to 68%, which can be considered good results given the complexity of the task. The highest accuracy of 68.3% was achieved when testing on a single size motif inserted into the spatiotemporal graph. In contrast, the lowest accuracy was obtained with graphs that had a higher motif density. Nevertheless, the other parameters tested do not seem to influence the accuracy of the framework.

When comparing the experiments, we observe variations in the ratio of obtained motifs, precision, and recall. For instance, in the experiments with a fixed number of graph nodes and a consistent distribution of relations, the ratio of obtained motifs ranges from 0.592 to 0.664. Similarly, precision and recall values display slight variations across experiments, indicating the consistency of the model. Furthermore, experiments involving different temporalities and relation distributions highlight the robustness of the framework. For instance, the experiment with the distribution of relations [2,2,2], and motifs of size 4 and 6, the framework achieves a ratio of obtained motifs of 0.661. This result suggests that Multi-SPMiner can effectively capture motifs even in scenarios with fewer temporalities and more balanced relation distributions. To the best of our knowledge, this is the first attempt to apply neural methods for searching frequent patterns in ST-graphs and multigraphs in general. Consequently, we do not have any existing methods to compare our results with.

In order to investigate the generalization capability of our model and its ability to learn structural relationships between nodes that facilitate motif discovery, we conducted a series of experiments where training and testing were performed on graphs with varying motif sizes. The objective was to assess whether the model could effectively transfer its learned knowledge to graphs with different motif sizes. Table 2 presents the results of these experiments. In the first experiment, we trained the model on graphs containing motifs of size 8, and subsequently tested it on graphs with motif sizes of 6. Similarly, in the second experiment, the model was trained on motifs of size 10 and tested on graphs containing motifs of size 6. These experiments aimed to examine the model's ability to handle different motif sizes during testing while maintaining consistent performance. Furthermore, we conducted additional experiments where we trained the model on motifs with smaller sizes (4 and 6) and evaluated its performance on graphs with larger motif sizes (7 and 9). This change in testing approach allowed us to investigate the model's capability to generalize from smaller motifs to larger ones. Remarkably, the obtained testing results presented in Table 2 closely

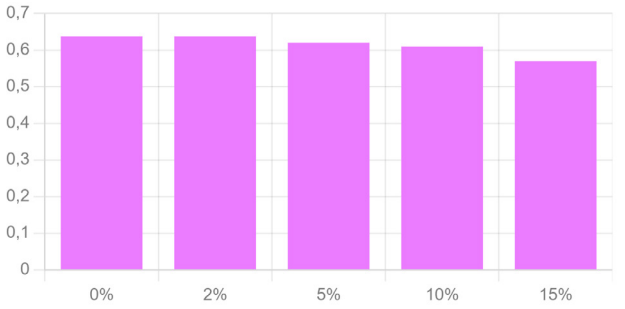

Fig. 3. Effect of Node Removal on Motif Discovery in the Graph: The percentages indicated below the bars represent the proportion of removed nodes, while the values on the left represent the corresponding ratio of discovered motifs.

aligned with the previously obtained results in Table 1. This indicates that our Multi-Graph Convolutional Network (MGCN) successfully learned the relationships between graphs and subgraphs, regardless of the specific motif sizes involved. It demonstrated the model's ability to effectively handle graphs with varying motif sizes during training and testing, highlighting its robustness and generalization capacity. These experiments provide valuable insights into the versatility of our proposed model and its capacity to discover motifs in graphs with diverse motif size distributions. Such flexibility enhances its applicability in real-world scenarios where motifs may exhibit significant variations in size. The demonstrated capability of our approach to effectively learn and exploit these relationships paves the way for broader and more accurate motif discovery tasks in various domains and the possibility to train and test on different data types.

To assess the robustness of our approach against missing nodes in the multigraph, we conducted an experiment involving the random removal of nodes from the multigraph. We replicated the fifth experiment from Table 1 to ensure consistency in our evaluation. In figure 3, we present the results obtained after removing 2%, 5%, 10%, and 15% of the nodes in the multigraph. The experimental results reveal that the accuracy of our model remains largely unaffected when 2% and 5% of the nodes are removed. However, a slight decrease in accuracy is observed when 10% and 15% of the nodes are removed. This decrease can be attributed to the impact of removed nodes on the frequency of certain motifs. Consequently, some motifs may no longer satisfy the support threshold s, resulting in their exclusion from consideration as frequent motifs by the algorithm. Nevertheless, it is important to note that our approach exhibits no significant decline in accuracy despite the removal of nodes. This finding underscores the resilience of our method against node removal and highlights its ability to maintain robust performance even in the presence of perturbations or missing data.

5.3. Comparision with SPMiner

Multi-SPMiner serves a different purpose compared to SPMiner, as it aims at extracting motifs of various sizes by iterating through different motif sizes, while SPMiner focuses on extracting top-N motifs of a fixed size. Our approach can be seen as a generalization of SPMiner, whereby considering only a single connection and ignoring the rest, Multi-SpMiner converges to the base model, SPMiner, resulting in an identical outcome. To examine our hypothesis, we conducted an experiment using the spatiotemporal graph generator, specifically focusing on graphs with only spatiotemporal connections. By disregarding node and edge labels, we obtained single-link graphs and inserted motifs of size 6. To compare our approach, Multi-SPMiner, with the pre-trained SPMiner model, we trained our model from scratch (using the same training protocol as in the previous experiments.). The results revealed comparable performance, with SPMiner achieving an accuracy of 78.7% and Multi-SPMiner achieving 74.2%. The slight discrepancy can be attributed to the differences in training data size (thousands of graphs for SPMiner versus 200 graphs for our model) and the incorporation of orientations and labels in our model, which SPMiner does not consider. Overall, Multi-SPMiner showcased promising results, showcasing its potential for multigraph analysis as initially intended.

The memory cost analysis of Multi-SPMiner, similar to SPMiner, is influenced by the number of neighborhoods and embedding dimension. The estimated memory cost follows an expression of $O((N+K)d+Kn^2)$, accounting for the number of neighborhoods (N), embedding dimension (d), number of decoder iterations (K), and desired motif size (n).

The runtime analysis considers factors such as neighborhoods, motif size, nodes, and embedding dimension, with a complexity of $O(Nb^2 + Kn(n^2 + Nnd))$. Compared to SPMiner, Multi-SPMiner introduces additional complexities due to the adoption of a multigraph convolutional GNN and the consideration of labeled directed multigraphs. Nonetheless, Multi-SPMiner exhibits efficient runtime and memory usage without significantly introducing complexity when compared to the base model.

6. Conclusion and future work

This paper introduces Multi-SPMiner, a novel approach for multi-graph pattern mining based on the multi-graph convolutional network. Building upon the limitations of SPMiner, which was originally designed for simple graphs, our approach extends the capabilities of deep learning-based graph motif mining. The proposed method aligns with the fundamental principles of SPMiner, encompassing two key phases: node embedding and motif extraction using a walk-based approach. However, it incorporates adaptations to effectively encode and analyze multirelational graphs. Experimental results have demonstrated the effectiveness of Multi-SPMiner in identifying frequent motifs in single multigraphs, and demonstrates its capability to handle data with motif sizes that were not specifically trained for. Additionally, by testing the approach on single connection graphs, we have showcased its potential as a generalized version of SPMiner, which enables the potential application of similar modifications to adapt the baseline approach for other types of graphs, such as knowledge graphs. Moving forward, In future endeavors, our focus will revolve around augmenting the precision of the framework and broadening its applicability to intricate and extensive datasets. This will be accomplished by integrating advanced learning techniques and sophisticated graph architectures, such as graph transformers. Furthermore, we have planned to conduct real-world data testing, particularly in the biomedical domain [13], to obtain valuable insights and validate the effectiveness of our approach using real-world datasets.

References

[1] C. C. Aggarwal. *Applications of Frequent Pattern Mining*, pages 443–467. Springer International Publishing, Cham, 2014.

[2] M. Atzmueller, H. Soldano, G. Santini, and D. Bouthinon. Minerlsd: Efficient local pattern mining on attributed graphs. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 219–228, 2018.

[3] A.G. Cohn, B. Bennett, J. Gooday, and N.M. Gotts. Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. *Geoinformatica*, 1(3):275–316, 1997.

[4] G. Del Mondo, M.A. Rodríguez, C. Claramunt, L. Bravo, and R. Thibaud. Modeling consistency of spatio-temporal graphs. *Data & Knowledge Engineering*, 84:59–80, 2013.

[5] M. Elseidy, E. Abdelhamid, S. Skiadopoulos, and P. Kalnis. Grami: Frequent subgraph and pattern mining in a single large graph. *Proceedings of the VLDB Endowment*, 7(7):517–528, 2014.

[6] P. Fournier-Viger, G. He, C. Cheng, J. Li, M. Zhou, J.C.W. Lin, and U. Yun. A survey of pattern mining in dynamic graphs. *WIREs Data Mining and Knowledge Discovery*, 10(6), May 2020.

[7] D. Gaur, A. Shastri, and R. Biswas. Metagraph-based substructure pattern mining. In *2008 International Conference on Advanced Computer Theory and Engineering*, pages 865–869, 2008.

[8] V. Ingalalli, D. Ienco, and P. Poncelet. Mining frequent subgraphs in multigraphs. *Information Sciences*, 451:50–66, 2018.

[9] C. Jiang, F. Coenen, and M. Zito. A survey of frequent subgraph mining algorithms. *The Knowledge Engineering Review*, 28(1):75–105, 2013.

[10] J. Kralj, M. Robnik-Šikonja, and N. Lavrač. HINMINE: heterogeneous information network mining with information retrieval heuristics. *Journal of Intelligent Information Systems*, 50(1):29–61, January 2017.

[11] M. Kuramochi and G. Karypis. Finding frequent patterns in a large sparse graph. *Data mining and knowledge discovery*, 11(3):243–271, 2005.

[12] A. Leborgne, M. Kirandjiska, and F. Le Ber. Random generation of a locally consistent spatio-temporal graph. In *Graph-Based Representation and Reasoning: 26th Int. Conference on Conceptual Structures, ICCS 2021*, LNCS 12879, pages 155–169. Springer, 2021. Online Event.

[13] A. Leborgne, F. Le Ber, L. Degiorgis, L. Harsan, S. Marc-Zwecker, and V. Noblet. Analysis of brain functional connectivity by frequent pattern mining in graphs. application to the characterization of murine models. In *2021 IEEE 18th Int. Symposium on Biomedical Imaging*, 2021.

[14] M. Prates, P.H.C. Avelar, H. Lemos, L.C. Lamb, and M.Y. Vardi. Learning to solve NP-complete problems: A graph neural network for decision TSP. In *Proceedings of the 33th AAAI Conference on Artificial Intelligence*, pages 4731–4738, 2019.

[15] C. Shi, Y. Li, J. Zhang, Y. Sun, and S.Y. Philip. A survey of heterogeneous information network analysis. *IEEE Transactions on Knowledge and Data Engineering*, 29(1):17–37, 2016.

[16] I. Vendrov, R. Kiros, S. Fidler, and R. Urtasun. Order-embeddings of images and language. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[17] R. Ying, A. Wang, J. You, and J. Leskovec. Frequent subgraph mining by walking in order embedding space. *The International Conference on Machine Learning (ICML)*, 2020.

[18] Z. Zhang, P. Cui, and W. Zhu. Deep learning on graphs: A survey. *IEEE Trans. on Knowledge and Data Engineering*, 34(1):249–270, 2020.