ABSTRACT

BELLO LANDER, GONZALO ALEJANDRO. Multi-objective Graph Mining Algorithms for Detecting and Predicting Communities in Complex Dynamic Networks. (Under the direction of Nagiza F. Samatova.)

Mining patterns in real-world systems represented as complex networks is a key task in many scientific domains. Most graph mining algorithms identify patterns that satisfy a single criterion, which is usually related to the structure of the graph. However, complex networks inherently have multiple sources of information. First, there is often a phenomenon of interest related to the underlying system that we want to analyze or predict, such as a particular phenotype in biological networks or a weather event in climate networks. Second, vertices in complex networks often have attributes that characterize them, such as age or gender in social networks or research area in citation networks. Moreover, complex networks are often dynamic in nature; that is, they change over time. Thus, identifying “interesting” patterns in complex networks often requires looking beyond the structure of the graph by incorporating additional sources of information, such as vertex attributes or a response variable of interest, and addressing the dynamic nature of the network.

Therefore, we posit that the emphasis of graph mining research should shift from the traditional structure-focused approaches to a multi-objective approach that incorporates multiple sources of information. In practice, this translates to a move from single-objective graph mining algorithms to multi-objective algorithms that identify patterns by optimizing multiple objective functions simultaneously while also taking into account the dynamic nature of the network. In this dissertation, we illustrate the value of this multi-objective approach in the context of community detection, an essential graph mining task.

First, we propose a multi-objective algorithm to detect communities associated with a response variable of interest. We applied our proposed algorithm to identify communities in climate networks associated with seasonal rainfall variability. The results obtained suggest that our algorithm is able to capture the underlying patterns known to be associated with the phenomenon of interest and to identify communities with greater predictive power for the response variable than state-of-the-art single-objective algorithms.

Second, we propose a multi-objective algorithm to detect communities in dynamic attributed graphs. We evaluated our proposed algorithm on synthetically generated dynamic attributed graphs and on large-scale networks constructed from real-world data sets. The results obtained suggest that our algorithm is able to identify communities with high modularity and attribute similarity and to achieve a better trade-off between efficiency and quality of the communities identified than state-of-the-art algorithms for community detection in static attributed graphs.
Finally, we further address the dynamic nature of complex networks by proposing a methodology to predict the future community structure of the graph using link prediction methods. We evaluated our proposed methodology on synthetically generated dynamic graphs and on real-world networks. The results obtained suggest that our methodology is able to predict future communities in dynamic graphs, but more accurate link prediction methods are needed to improve the accuracy of the prediction in the case of real-world networks.
Multi-objective Graph Mining Algorithms for Detecting
and Predicting Communities in Complex Dynamic Networks

by
Gonzalo Alejandro Bello Lander

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Computer Science

Raleigh, North Carolina
2017

APPROVED BY:

Dennis R. Bahler .........................................................
Rada Y. Chirkova .........................................................

R. Raju Vatsavai .........................................................
Nagiza F. Samatova
Chair of Advisory Committee
DEDICATION

A mis padres, Iraida y Gonzalo,
por su amor y apoyo incondicional.

To my parents, Iraida and Gonzalo,
for their unconditional love and support.
BIOGRAPHY

Gonzalo A. Bello is a Ph.D. candidate in the Department of Computer Science at North Carolina State University advised by Dr. Nagiza F. Samatova. He received an M.S. in Computer Science from North Carolina State University in 2015 and a B.E. in Systems Engineering from Universidad Metropolitana in Caracas, Venezuela in 2010. As a graduate student at North Carolina State University, he has been the instructor of record for undergraduate courses on data structures and theory of computation and received an Outstanding Teaching Assistant Award from the Department of Computer Science in 2014. His primary areas of interest are data mining and algorithms. His work has been published in top conferences in data mining, such as the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD), and his most recent paper received the Best Student Paper Award at the 12th International Conference on Advanced Data Mining and Applications (ADMA). His current research focuses on multi-objective algorithms for community detection in complex networks and their application in social network analysis and climate informatics.
ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor, Dr. Nagiza F. Samatova, for her constant guidance throughout my graduate studies. Her invaluable advice and the numerous opportunities she provided for my professional development have allowed me to grow as an individual, a researcher, and a future professor.

I would also like to thank the faculty at North Carolina State University’s Department of Computer Science, particularly the members of my advisory committee—Dr. Dennis R. Bahler, Dr. Rada Y. Chirkova, and Dr. R. Raju Vatsavai—for their time and insightful suggestions regarding this dissertation, and Dr. George N. Rouskas, Director of Graduate Programs, for his help and assistance.

I am also grateful to my collaborators from the National Science Foundation (NSF) Expeditions in Computing project “Understanding Climate Change: A Data Driven Approach,” particularly Dr. Fredrick H. M. Semazzi at North Carolina State University’s Department of Marine, Earth, and Atmospheric Sciences (MEAS) and Dr. Vipin Kumar at the University of Minnesota. The work I performed as part of this project constitutes an integral part of this dissertation. I am very thankful to Dr. Semazzi’s research group, particularly to Michael Angus and Dr. Pascal F. Waniha, for their useful input regarding the application of my work in the climate science domain.

I would like to thank my fellow students and collaborators at Dr. Samatova’s research group for their many contributions to my work, particularly Steve Harenberg and also Mandar S. Chaudhary, Jitendra K. Harlalka, Stephen Ranshous, Navya Pedemane, Abhishek Agrawal, David “Drew” Boyuka, Lucia Gjeltema, Doel L. Gonzalez, and Kanchana Padmanabhan.

I am especially thankful to my dear friend, Asad Rahman, for constantly pushing me to succeed and achieve my goals.

Finally, I would like to thank my family, particularly my parents, Iraida and Gonzalo, and my grandmother, Consuelo. Without their unconditional love and support, this dissertation would not have been possible.

This dissertation is based upon work supported in part by the Laboratory for Analytic Sciences (LAS), the U.S. Department of Energy (DOE), Office of Science (SC), Advanced Scientific Computing Research (ASCR), and the NSF grant 1029711. Any opinions, findings, conclusions, or recommendations expressed in this dissertation are those of the author and do not necessarily reflect the views of any agency or entity of the United States government.
<table>
<thead>
<tr>
<th>Chapter 1 Introduction</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Community Detection</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Multi-Objective Community Detection</td>
<td>3</td>
</tr>
<tr>
<td>1.3.1 Detecting Communities Associated with a Response Variable</td>
<td>5</td>
</tr>
<tr>
<td>1.3.2 Detecting Communities in Dynamic Attributed Graphs</td>
<td>6</td>
</tr>
<tr>
<td>1.3.3 Predicting Future Communities in Dynamic Graphs</td>
<td>6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 2 Detecting Communities Associated with a Response Variable of Interest</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Introduction</td>
<td>8</td>
</tr>
<tr>
<td>2.2 Problem Statement</td>
<td>10</td>
</tr>
<tr>
<td>2.3 Response-Guided Community Detection</td>
<td>11</td>
</tr>
<tr>
<td>2.3.1 Greedy Algorithm for Response-Guided Community Detection</td>
<td>12</td>
</tr>
<tr>
<td>2.3.2 Simulated Annealing for Response-Guided Community Detection</td>
<td>13</td>
</tr>
<tr>
<td>2.4 Climate Index Discovery</td>
<td>15</td>
</tr>
<tr>
<td>2.5 Response-Guided Network Construction</td>
<td>17</td>
</tr>
<tr>
<td>2.5.1 Selecting the Set of Vertices</td>
<td>17</td>
</tr>
<tr>
<td>2.5.2 Defining the Set of Edges</td>
<td>18</td>
</tr>
<tr>
<td>2.5.3 Incorporating Spatial Neighborhood Information and Multiple Related Response Variables</td>
<td>18</td>
</tr>
<tr>
<td>2.6 Experimental Evaluation</td>
<td>19</td>
</tr>
<tr>
<td>2.6.1 Data Description</td>
<td>20</td>
</tr>
<tr>
<td>2.6.2 Data Preprocessing</td>
<td>20</td>
</tr>
<tr>
<td>2.6.3 Climate Networks Constructed</td>
<td>21</td>
</tr>
<tr>
<td>2.6.4 Climate Indices Discovered</td>
<td>22</td>
</tr>
<tr>
<td>2.6.5 Seasonal Rainfall Prediction</td>
<td>24</td>
</tr>
<tr>
<td>2.6.6 Physical Interpretation of Climate Indices Discovered</td>
<td>27</td>
</tr>
<tr>
<td>2.7 Conclusion</td>
<td>28</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 3 Detecting Communities in Dynamic Attributed Graphs</th>
<th>29</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Introduction</td>
<td>29</td>
</tr>
<tr>
<td>3.2 Related Work</td>
<td>30</td>
</tr>
<tr>
<td>3.3 Problem Statement</td>
<td>31</td>
</tr>
<tr>
<td>3.4 Community Detection in Dynamic Attributed Graphs</td>
<td>31</td>
</tr>
<tr>
<td>3.4.1 Measuring the Quality of the Partition of the Graph</td>
<td>31</td>
</tr>
<tr>
<td>3.4.2 Detecting Communities in Attributed Graphs</td>
<td>33</td>
</tr>
<tr>
<td>3.4.3 Updating Communities in Dynamic Graphs</td>
<td>35</td>
</tr>
<tr>
<td>3.4.4 Tracking Communities in Dynamic Attributed Graphs</td>
<td>36</td>
</tr>
</tbody>
</table>
LIST OF TABLES

Table 2.1 Properties of networks constructed and climate indices discovered for OND rainfall variability in the GHA, using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms: number of networks (Num Nets), average number of vertices and edges per network (Avg Vtxs, Avg Edges), average modularity (Avg Mod), number of indices (Num Idxs), average number of vertices, standard deviation, and internal density per index (Avg Vtxs, Avg Std, Avg Dens), and percentage of indices with a statistically significant ($p < 0.01$) linear correlation with the response variable of interest (% Idxs). Best values are highlighted in bold. .................................................. 23

Table 2.2 Average linear correlation with OND rainfall at each station and at the GHA region, over the training set and the test set, of climate indices discovered for OND rainfall variability in the GHA using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms. Check marks (√) indicate that our proposed methodology performs significantly better according to a two-way ANOVA with a significance level of 0.05. Best values are highlighted in bold. ................. 24

Table 2.3 Linear correlation between true and predicted rainfall (Corr) and RMSE scores for predictions of OND rainfall at each station and at the GHA region from 1998 to 2011 obtained using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms. Check marks (√) indicate that our proposed methodology performs significantly better according to a two-way ANOVA with a significance level of 0.05. Best values are highlighted in bold. ................. 25

Table 3.1 Number of vertices and number of edges per time step of real-world networks 45
Table 3.2 Results obtained (number of communities, modularity, average density, attribute similarity, average entropy or average standard deviation, peak memory in MB, and running time in seconds) using our proposed algorithm and a state-of-the-art community detection algorithm (CODICIL [48] for networks with categorical attributes or I-Louvain [11] for networks with numeric attributes) on large-scale real-world networks. Asterisks (*) and daggers (†) indicate that the implementation of the algorithm did not run in the allotted time of 5 hours or with the allotted memory of 64GB, respectively. Best results are shown in bold and underlined. ................. 47
Table 3.3  Number of vertices (Num Vertices), percentage of vertices from previous time step (% t − 1 Vertices), and top 3 attributes per time step for the largest community in the DBLP network at $t = 0$, illustrating the evolution of this community from $t = 0$ to $t = 7$. Attributes in the DBLP network, such as Computer Science (CS), Math, Systems, Engineering (Eng), and Artificial Intelligence (AI), represent authors’ areas of publication.

Table 4.1  Number of vertices, number of edges, mixing parameter ($\mu$), and percentage of new edges ($\delta$) per time step of dynamic graphs constructed from real-world data sets.
LIST OF FIGURES

Figure 1.1 Example of a climate network constructed using monthly sea surface temperature data from 2000 to 2010. Vertices (red circles) correspond to spatial points in a global grid and edges (blue lines) indicate statistically significant correlations between the underlying sea surface temperature data at a pair of spatial points. .................................................. 2

Figure 1.2 Overview of the dissertation. In Chapter 2, we propose a multi-objective algorithm to detect communities associated with a response variable of interest. In Chapter 3, we propose a multi-objective algorithm to detect, update, and track communities in dynamic attributed graphs. In Chapter 4, we propose a methodology to predict future communities in dynamic graphs using link prediction methods. .................................................. 5

Figure 2.1 Overview of the proposed methodology for climate index discovery using response-guided community detection .................................................. 16

Figure 2.2 Map highlighting (in red) the North Eastern Highlands of Tanzania (left) and the four stations located in this region that were used for this study: Arusha, Kilimanjaro, Moshi, and Same (right) .................................................. 20

Figure 2.3 Spatial points selected in the month of October to construct the climate network. Blue triangles indicate spatial points selected only for individual stations and red circles indicate spatial points in the consensus set. .... 21

Figure 2.4 Climate indices discovered using the proposed methodology with the multi-objective algorithms for response-guided community detection based on the Louvain method (left) and simulated annealing (right) and with OND rainfall variability in the GHA as the response variable of interest. Each color represents a different index and diamonds indicate overlaps between indices. To improve visualization, only the top 10 indices with the highest association with the response variable over the training set are shown in each figure. .................................................. 22

Figure 2.5 Average linear correlation between true and predicted rainfall for predictions of OND rainfall at each station in the GHA region over the training set using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms vs. the number of predictors used to build the regression models. The dashed line indicates the number of predictors selected for further analysis. .................................. 25

Figure 2.6 Linear correlation between true and predicted rainfall and RMSE scores for predictions of OND rainfall at each station in the GHA region from 1998 to 2011 obtained using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms ... 26
Figure 2.7  Classification accuracy of the prediction of the OND rainfall season at each station in the GHA region from 1998 to 2011 obtained using the proposed methodology with the multi-objective algorithms for response-guided community detection based on the Louvain method (LM) and simulated annealing (SA), as well as official forecasts issued by the TMA.

Figure 2.8  Time series of the Niño 3.4 index (upper, solid line) and the IOD index (lower, solid line) with climate indices discovered in July (upper, dashed line) and October (lower, dashed line) using the proposed methodology with the multi-objective algorithm for response-guided community detection based on the Louvain method and with OND rainfall variability in the GHA as the response variable of interest. The linear correlation between the time series is shown in the lower left corner of each figure.

Figure 3.1  Average modularity of the partitions obtained at each time step of the benchmark graphs using our proposed algorithm with update parameter $\beta = 1$ and multiple values of the weighting parameter $\alpha$. Benchmark graphs were generated using the following parameters: number of vertices $n = 1000$, mixing parameter $\mu = \{0.1, 0.9\}$, noise parameter $\lambda = \{0.1, 0.9\}$, and change parameter $\delta = 0.1$.

Figure 3.2  Average attribute similarity of the partitions obtained at each time step of the benchmark graphs using our proposed algorithm with update parameter $\beta = 1$ and multiple values of the weighting parameter $\alpha$. Benchmark graphs were generated using the following parameters: number of vertices $n = 1000$, mixing parameter $\mu = \{0.1, 0.9\}$, noise parameter $\lambda = \{0.1, 0.9\}$, and change parameter $\delta = 0.1$.

Figure 3.3  Average running time (in seconds) at each time step of the benchmark graphs of our proposed algorithm with weighting parameter $\alpha = 0.5$ and multiple values of the update parameter $\beta$. Benchmark graphs were generated using the following parameters: number of vertices $n = \{1000, 10000\}$, mixing parameter $\mu = 0.1$, noise parameter $\lambda = 0.1$, and change parameter $\delta = \{0.1, 0.9\}$.

Figure 3.4  Average modularity and attribute similarity of the partitions obtained at each time step of the benchmark graphs using our proposed algorithm with weighting parameter $\alpha = 0.5$ and multiple values of the update parameter $\beta$. Benchmark graphs were generated using the following parameters: number of vertices $n = 1000$, mixing parameter $\mu = 0.1$, noise parameter $\lambda = 0.1$, and change parameter $\delta = \{0.1, 0.9\}$. Comparable results were obtained for $n = 10000$.

Figure 3.5  Box plots of estimated mixing parameter (left) and estimated change parameter (right) for benchmark graphs with $n = 1000$ vertices.
Figure 3.6 Histogram of vertex change scores (left axis) and probability density functions of the two components of a Gaussian mixture model fitted to the vertex change scores using the Expectation Maximization (EM) algorithm (right axis) for benchmark graphs with \( n = 1000 \) vertices and change parameter \( \delta = 0.1 \) (a), 0.5 (b), and 0.9 (c). The corresponding Bayesian information criterion (BIC) values are shown in the upper corner of each figure. Comparable results were obtained for other values of \( \delta \).

Figure 3.7 Average modularity and attribute similarity of the partitions obtained and average running time (in seconds) at each time step of the benchmark graphs using our proposed algorithm with weighting parameter \( \alpha = 0.5 \) and update parameter \( \beta = 0.5 \). Benchmark graphs were generated using the following parameters: number of vertices \( n = 1000 \), mixing parameter \( \mu = 0.1 \), noise parameter \( \lambda = 0.1 \), and change parameter \( \delta = 0.1 \).

Figure 3.8 Running time (left, in seconds) and peak memory (right, in MB) obtained at each time step of the real-world networks using our proposed algorithm, CODICIL [48], and I-Louvain [11]. Line styles and colors are used to denote the algorithms and marker types are used to denote the data sets. Missing results indicate that the implementation of the algorithm did not run in the allotted time of 5 hours or with the allotted memory of 64GB.

Figure 3.9 Word clouds of attributes per time step for the largest community in the DBLP network at \( t = 0 \), illustrating the evolution of this community from \( t = 0 \) to \( t = 7 \). Attributes in the DBLP network, such as Math, Systems, Engineering, and Artificial Intelligence (AI), represent authors' areas of publication. The Computer Science attribute was omitted from the word clouds as it is the top 1 attribute in all communities.

Figure 4.1 Baseline Adjusted Rand Index (ARI) obtained on \( N = 100 \) synthetic dynamic graphs with \( \delta = 0.1 \) and various values of \( \mu \) (left) and with \( \mu = 0.5 \) and various values of \( \delta \) (right), where \( \mu \) is the mixing parameter of the graph and \( \delta \) is the percentage of new edges inserted. Shaded regions indicate plus or minus one standard deviation centered around the mean.

Figure 4.2 Predicted and baseline Adjusted Rand Index (ARI) obtained on \( N = 100 \) synthetic dynamic graphs with \( \mu = 0.5 \) and \( \delta = 0.1 \), where \( \mu \) is the mixing parameter of the graph and \( \delta \) is the percentage of new edges inserted, for various percentages of edges predicted correctly (left) and incorrectly (right). Shaded regions indicate plus or minus one standard deviation centered around the mean.

Figure 4.3 State-of-the-art (SOTA) and baseline Adjusted Rand Index (ARI, left) and percentage of edges predicted correctly (right) obtained on \( N = 10 \) synthetic dynamic graphs with \( \delta = 0.1 \) and various values of \( \mu \), where \( \mu \) is the mixing parameter of the graph and \( \delta \) is the percentage of new edges inserted. Shaded regions indicate plus or minus one standard deviation centered around the mean. Comparable results were obtained for other values of \( \delta \).
Figure 4.4 Adjusted Rand Index (ARI) obtained using the baseline and the proposed community prediction methodology with multiple link prediction methods (Common Neighbors, Jaccard’s Coefficient, Adamic-Adar, Resource Allocation, Katz, and Structural Perturbation Method) on $N = 100$ synthetic dynamic graphs with $\delta = 0.1$ and various values of $\mu$, where $\mu$ is the mixing parameter of the graph and $\delta$ is the percentage of new edges inserted. Red boxes indicate a statistically significant ($p < 0.01$) improvement with respect to the baseline. Comparable results were obtained for other values of $\delta$.  

Figure 4.5 Percentage of edges predicted correctly obtained using multiple link prediction methods (Common Neighbors, Jaccard’s Coefficient, Adamic-Adar, Resource Allocation, Katz, and Structural Perturbation Method) on $N = 100$ synthetic dynamic graphs with $\delta = 0.1$ and various values of $\mu$, where $\mu$ is the mixing parameter of the graph and $\delta$ is the percentage of new edges inserted. Comparable results were obtained for other values of $\delta$.  

Figure 4.6 Adjusted Rand Index (ARI) obtained using the baseline and a modified version of the proposed community prediction methodology that uses predicted future edges instead of edge likelihoods on $N = 100$ synthetic dynamic graphs with $\mu = 0.5$ and $\delta = 0.1$, where $\mu$ is the mixing parameter of the graph and $\delta$ is the percentage of new edges inserted. Red boxes indicate a statistically significant ($p < 0.01$) improvement with respect to the baseline. Comparable results were obtained for other values of $\mu$.  

Figure 4.7 Median number of common neighbors between vertices incident to edges present at the initial time step $t$ (left) and incident to edges inserted at the following time step $t+1$ (right) on $N = 100$ synthetic dynamic graphs with $\delta = 0.1$ and various values of $\mu$, where $\mu$ is the mixing parameter of the graph and $\delta$ is the percentage of new edges inserted. Shaded regions indicate plus or minus one standard deviation centered around the mean.  

Figure 4.8 Predicted and baseline Adjusted Rand Index (ARI) obtained for various percentages of edges predicted correctly (20%, 40%, 60%, and 80%) on dynamic graphs constructed from real-world data sets: DBLP (left, upper row), TripAdvisor (right, upper row), and Yelp (left, lower row). Shaded regions indicate plus or minus one standard deviation centered around the mean for a sample of size $N = 10$.  

Figure 4.9 Adjusted Rand Index (ARI, left) and percentage of edges predicted correctly (right) obtained using the baseline and the proposed community prediction methodology with local similarity-based link prediction methods (Common Neighbors, Jaccard’s Coefficient, Adamic-Adar, and Resource Allocation) on dynamic graphs constructed from real-world data sets: DBLP (upper row), TripAdvisor (middle row), and Yelp (lower row).
Figure 4.10 Adjusted Rand Index (ARI, left) and percentage of edges predicted correctly (right) obtained using the baseline and the state-of-the-art (SOTA) community prediction methodology [40] on dynamic graphs constructed from real-world data sets: DBLP (upper row) and TripAdvisor (lower row). Due to its high computational complexity, the SOTA methodology did not run on the Yelp network in the time allotted for the experimental evaluation.
Chapter 1

Introduction

1.1 Introduction

Complex systems across multiple scientific domains are commonly represented as networks, where vertices correspond to objects in the system and edges correspond to interactions between these objects. For example, in biology, the metabolic system can be represented as a network of interacting metabolites [22], while, in climate science, the global climate system can be represented as a network of interacting regions [57] (see Figure 1.1). This representation allows researchers to make use of existing graph mining algorithms to identify “interesting” network patterns, which, in turn, can help them better understand the behavior of the system.

Graph mining algorithms usually identify network patterns in the form of subgraphs that satisfy a given criterion, which, in most cases, is related to the structure of the graph. For example, maximal clique enumeration algorithms identify complete subgraphs [50], while frequent subgraph mining algorithms identify frequently occurring isomorphic subgraphs [27]. However, given the complex nature of the systems being represented by these networks, identifying “interesting” patterns often requires looking beyond the structure of the graph.

On one hand, complex networks inherently have multiple sources of information. For example, social networks contain not only vertices and edges representing users and their interactions, but also attributes of heterogeneous types (e.g., age, gender) describing these users. However, most graph mining algorithms do not take these additional sources of information into account [1]. In Chapter 2 and Chapter 3, we show that incorporating additional sources of information, such as vertex attributes or a response variable of interest, allows us to identify more meaningful patterns.

On the other hand, complex networks are often dynamic in nature; that is, they are constantly, and in many cases rapidly, changing. However, most graph mining algorithms do not take this dynamic nature into account, opting instead to identify patterns for a given snapshot
Figure 1.1: Example of a climate network constructed using monthly sea surface temperature data from 2000 to 2010. Vertices (red circles) correspond to spatial points in a global grid and edges (blue lines) indicate statistically significant correlations between the underlying sea surface temperature data at a pair of spatial points.

of the network [1]. In Chapter 3, we show that incorporating the temporal changes of the network allows us to identify patterns more efficiently. Furthermore, in Chapter 4, we show that this also allows us to predict these patterns in future time steps.

For these reasons, we posit that the emphasis of graph mining research should shift from the traditional structure-focused approaches to a multi-objective approach that incorporates multiple sources of information in addition to the structural properties of the graph. In practice, this translates to a move from single-objective graph mining algorithms to multi-objective algorithms that identify patterns by optimizing multiple objective functions simultaneously while also taking into account the dynamic nature of the network. In this dissertation, we illustrate the value of this multi-objective approach in the context of one specific graph mining task: community detection.

1.2 Community Detection

Networks representing complex systems often contain communities; that is, groups of densely connected vertices that are sparsely connected to the rest of the graph [18]. These communities represent groups of related objects that may share common properties or play similar roles within the system. For example, communities may represent groups of friends in social networks, groups of functionally associated genes in biological networks [24], or spatiotemporal climate
patterns in climate networks [57].

Identifying the community structure of a network is essential for understanding the underlying system. Therefore, community detection has become one of the most important tasks in the field of network analysis [18, 23], with applications in many domains. Most existing methods for community detection partition the network into a set of communities $C$ by optimizing a single objective function $f$, which typically measures some structural property of the graph; that is,

$$\min_{C \in \Omega} f(C)$$  \hspace{1cm} (1.1)

where $\Omega$ is the set of feasible solutions.

While no objective function for community detection is universally accepted [18], the goal of most existing methods is to optimize the trade-off between the intra-cluster density and the inter-cluster density of the communities identified. The intra-cluster density and the inter-cluster density of a community $c_i$, $D_{\text{int}}(c_i)$ and $D_{\text{ext}}(c_i)$, respectively, are given by

$$D_{\text{int}}(c_i) = \frac{k_{\text{int},c_i}}{\frac{1}{2} \cdot |c_i|(|c_i| - 1)}$$  \hspace{1cm} (1.2)

$$D_{\text{ext}}(c_i) = \frac{k_{\text{ext},c_i}}{|c_i|(n - |c_i|)}$$  \hspace{1cm} (1.3)

where $k_{\text{int},c_i}$ is the number of edges between vertices in community $c_i$ (i.e., internal edges), $k_{\text{ext},c_i}$ is the number of edges between vertices in community $c_i$ and the rest of the graph (i.e., external edges), and $n$ is the number of vertices in the graph.

However, as is the case with graph mining in general, this single-objective approach to community detection may be insufficient because it limits the communities identified to those that satisfy a particular structural property. This is especially true when the structural information of the network is incomplete or noisy (i.e., missing or incorrect edges). In these cases, incorporating additional sources of information, such as vertex attributes, would allow us to identify the community structure of the network more precisely [64]. For this reason, we posit that a shift towards multi-objective community detection methods that take into account multiple sources of information simultaneously is needed.

### 1.3 Multi-Objective Community Detection

Multi-objective community detection methods partition a network into a set of communities $C$ by optimizing multiple objective functions simultaneously; that is,

$$\min_{C \in \Omega} [f_1(C), f_2(C), ..., f_k(C)]$$  \hspace{1cm} (1.4)
where $\Omega$ is the set of feasible solutions and $f_i(C)$ for $i = 1, 2, ..., k$ are the objective functions. A solution to Equation 1.4 is said to be *Pareto optimal* if it is not possible to modify this solution to improve at least one of the objective functions without degrading any of the rest [51].

Multi-objective community detection has been recently proposed as a means to incorporate multiple structural properties of the graph into the community detection process [52, 51, 63]. For example, Shi et al. [52, 51] and Wu and Pan [63] make use of evolutionary algorithms to identify communities by optimizing two objective functions that measure the intra-cluster density and the inter-cluster density, respectively.

In this dissertation, we further capitalize on the concept of multi-objective community detection by incorporating not only the structural properties of the graph, but also additional sources of information, such as vertex attributes or a response variable of interest, while taking into account the dynamic nature of the network. Our general approach is to optimize multiple objective functions that measure these different sources of information using the well-known *weighted sum* method [41]; that is, we optimize a linear combination of the multiple objective functions given by

$$
\min_{C \in \Omega} \sum_{i=1}^{k} w_i f_i(C) \quad (1.5)
$$

where $\Omega$ is the set of feasible solutions, $f_i(C)$ for $i = 1, 2, ..., k$ are the objective functions, and $w_i$ for $i = 1, 2, ..., k$ are weighting parameters that denote the relative importance of each objective function. A solution to Equation 1.5 is Pareto optimal if $w_i > 0$ for $i = 1, 2, ..., k$ [54]. Moreover, if the graphs are dynamic, we also incorporate the temporal changes of the different sources of information in order to update the communities more efficiently or to predict the communities in future time steps.

It is worth noting that evolutionary algorithms for multi-objective optimization, such as the ones used in previous studies [52, 51, 63], have certain advantages over the weighted sum method: they do not require weighting parameters and they allow us to identify a set of Pareto optimal solutions instead of a single Pareto optimal solution. However, evolutionary algorithms are computationally expensive, and thus they are not suitable for the large-scale networks in most real-world applications.

We adopt this multi-objective approach to tackle three distinct research objectives related to community detection in complex networks:

- Detecting communities associated with a response variable of interest that can be used to analyze or predict this response variable.

- Detecting communities in dynamic attributed graphs.

- Predicting future communities in dynamic graphs.
These objectives, which correspond to the main chapters of this dissertation (see Figure 1.2), are further explained in the following sections.

Figure 1.2: Overview of the dissertation. In Chapter 2, we propose a multi-objective algorithm to detect communities associated with a response variable of interest. In Chapter 3, we propose a multi-objective algorithm to detect, update, and track communities in dynamic attributed graphs. In Chapter 4, we propose a methodology to predict future communities in dynamic graphs using link prediction methods.

1.3.1 Detecting Communities Associated with a Response Variable

Communities in complex networks may be used to analyze or predict a phenomenon of interest related to the system represented by the network. For example, communities in climate networks may be used to predict a weather event [57], while communities in biological networks may be used to analyze a particular disease [24]. However, single-objective community detection algorithms do not take into account the variability of this phenomenon of interest, and thus the communities identified may not necessarily be associated with it.

For this reason, in Chapter 2 [4], we formulate the problem of response-guided community detection; that is, identifying communities in a graph associated with a response variable of interest. We then propose a multi-objective algorithm for response-guided community detection that explicitly incorporates information of the response variable into the community detection process. We apply our proposed algorithm to the discovery of climate indices–time series that
summarize spatiotemporal climate patterns–associated with seasonal rainfall variability, a key task in the climate science domain. The results obtained suggest that our proposed algorithm is able to capture the underlying patterns known to be associated with the phenomenon of interest. Moreover, the communities identified using our proposed algorithm were shown to have a greater predictive power for the response variable than communities identified using single-objective algorithms [57].

1.3.2 Detecting Communities in Dynamic Attributed Graphs

Vertices in complex networks usually have individual properties, or attributes, that characterize them, such as age or gender in social networks or research area in citation networks. Moreover, complex networks are often dynamic in nature; that is, they are constantly changing. While some multi-objective algorithms for community detection in attributed graphs have been recently proposed [11], none of them also identify communities in dynamic graphs, and thus would need to be run from scratch every time the graph changes. Moreover, most of these algorithms are designed to handle only certain attribute types, which hinders their applicability to real-world networks.

For this reason, in Chapter 3 [5], we propose a multi-objective algorithm for community detection in dynamic attributed graphs that incorporates both the structural properties of the graph and the attributes of the vertices while also taking into account the dynamic nature of the network. Our proposed algorithm handles graphs with heterogeneous attribute types, as well as changes to both the structure of the graph and the attribute information of the vertices, which is essential for its applicability to real-world networks. We evaluated our proposed algorithm on a variety of synthetically generated dynamic attributed graphs and on large-scale real-world networks. The results obtained show that our proposed algorithm is able to identify graph partitions with modularity and attribute similarity comparable to that of state-of-the-art algorithms [11, 48]. However, the time and memory requirements of the state-of-the-art algorithms are considerably higher, suggesting that our proposed algorithm is able to achieve a better trade-off between efficiency and quality of the communities identified.

1.3.3 Predicting Future Communities in Dynamic Graphs

As previously mentioned, complex networks are often dynamic in nature, and thus their community structure is constantly changing. Several methods have been recently proposed to detect [13, 3] and track [19] communities in dynamic graphs and to characterize the evolution of these communities over time [43]. The problem of link prediction—that is, predicting future edges in dynamic graphs—has also been widely studied. However, little attention has been given to the prediction of communities in future time steps.
For this reason, in Chapter 4, we propose a methodology to predict future communities in dynamic graphs using link prediction methods. Unlike existing community prediction methodologies [40], our proposed approach directly incorporates edge likelihoods obtained using link prediction methods into the community detection process, and thus is more robust to errors in the prediction of future edges. We evaluated our proposed methodology on a variety of synthetically generated dynamic graphs and on real-world networks. The results obtained show statistically significant improvements with respect to the baseline on the synthetic graphs but not on the real-world networks due to the low accuracy of the link prediction methods on these networks. This suggests that, while our proposed methodology can potentially predict future communities in dynamic graphs, more accurate link prediction methods are needed to improve the accuracy of the prediction on real-world networks.
Chapter 2
Detecting Communities Associated with a Response Variable of Interest

2.1 Introduction

Detecting communities in complex networks is a key task in many scientific domains, such as climate science and biology. As discussed in Chapter 1, scientists in these domains are often concerned with finding communities associated with a response variable of interest that can be used to analyze or predict this response variable. For example, in climate science, such communities may represent spatiotemporal climate patterns associated with a particular weather event (e.g., precipitation) [57], while, in biology, they may represent groups of functionally associated genes associated with a particular disease (e.g., Alzheimer’s disease) [24].

However, as explained in Chapter 1, most community detection methods identify communities using single-objective algorithms that do not take into account the variability of the response variable of interest. Therefore, the communities identified may not necessarily be associated with the response variable. Semi-supervised methods that incorporate prior knowledge into the community detection process have been recently proposed [15]. However, these methods require partial information about the community memberships, which is usually not available, and, more importantly, they do not take into account the response variable of interest either. For this reason, we introduce the problem of response-guided community detection—that is, identifying communities in a graph associated with a response variable of interest—and study its application to the discovery of climate indices, an important task in the climate science domain.

Climate indices are time series that summarize spatiotemporal patterns in the global climate system. These patterns are often associated with temperature, pressure, and wind anomalies, which can have a significant impact on regional climate. Consequently, climate indices are frequently used to analyze and predict regional weather events. For example, climate indices
defined for El Niño Southern Oscillation (ENSO)—one of the most widely studied climate patterns, characterized by temperature and pressure anomalies over the Pacific Ocean—are used to forecast Atlantic hurricane activity [20].

Climate indices were traditionally the product of hypothesis-driven research. However, the increasing amount of climate data available has led to the adoption of data-driven approaches to guide and accelerate climate index discovery. The most common of such approaches is the use of Principal Component Analysis (PCA)—known in the climate science domain as Empirical Orthogonal Function (EOF) analysis—to identify major modes of variability in the data. However, the use of eigenvalue analysis techniques, such as PCA, to discover climate indices has important limitations with respect to the physical interpretability of the climate indices discovered and the ability of these techniques to detect weaker patterns [55].

An alternative approach to climate index discovery is the application of clustering techniques, such as Shared Nearest Neighbor (SNN) clustering, to identify regions of homogeneous long-term variability in climate data [55] or, more recently, the application of community detection methods to identify communities in climate networks [57]. The validity of the clusters or communities identified as climate indices has been evaluated in terms of their ability to predict a response variable of interest, such as temperature or precipitation [55, 57]. However, since none of these methods take the response variable into account, the climate indices discovered may not necessarily be good predictors.

For this reason, we propose a methodology to discover climate indices associated with a response variable of interest that makes use of a multi-objective community detection algorithm that explicitly incorporates information of this response variable into the community detection process. We apply this methodology to the discovery of climate indices associated with seasonal rainfall variability in the Greater Horn of Africa (GHA) and validate the climate indices discovered in terms of their predictive power and climatological relevance. Our results show that discovering climate indices associated with a response variable of interest allows us to identify its potential sources of variability. Moreover, using these climate indices as predictors allows us to improve forecasts of the response variable. This is especially important because improving the accuracy of regional climate forecasts, particularly of precipitation, is one of the major current challenges in climate science [49].

The main contributions of this chapter are as follows. First, we formulate the problem of response-guided community detection (Section 2.2) and propose a general multi-objective strategy to identify communities associated with a response variable of interest by explicitly incorporating information of this response variable into the community detection process (Section 2.3). Second, we propose a methodology to discover climate indices associated with a response variable of interest from multivariate spatiotemporal data using response-guided community detection (Section 2.4). As part of this methodology, we introduce a network representation
of multivariate spatiotemporal data that, unlike existing network construction methodologies [56, 57, 61], builds the network in a response-guided manner while also incorporating multiple covariates, spatial neighborhood information, and multiple related response variables into the network construction process (Section 2.5).

The rest of this chapter is organized in the following way. In Section 2.2, we formally define the problem of response-guided community detection. In Section 2.3, we describe a general multi-objective strategy for response-guided community detection and present two examples of community detection algorithms that can be adapted in this way to identify communities associated with a response variable of interest. We present an overview of our proposed methodology for the discovery of climate indices associated with a response variable of interest in Section 2.4, and detail our proposed network construction methodology in Section 2.5. In Section 2.6, we describe the experimental evaluation of our proposed methodology for climate index discovery and report the results obtained. Finally, we present our conclusions in Section 2.7.

2.2 Problem Statement

Let $X = \{x_{t,d,f} \in \mathbb{R} | t \in T, d \in D, f \in F\}$ be a multivariate spatiotemporal data set and $Y = \{y_t \in \mathbb{R} | t \in T\}$ be a response variable, where $T$ is a set of time steps, $D$ is a set of spatial points, and $F$ is a set of covariates. For our motivating application of climate index discovery, $X$ may be a global climate data set for a given month, $Y$ may be the total rainfall at a target region for a given season, $T$ may be a set of years, $D$ may be a set of global coordinates, and $F$ may be a set of climate variables (e.g., temperature, pressure, humidity).

Let data set $X$ be represented as a graph $G = (V,E)$, where $V \subseteq D$ is the set of vertices, $E$ is the set of edges, and each edge $(d_1,d_2) \in E$ is defined based on a domain-specific relationship between the data at spatial points $d_1$ and $d_2$ for all covariates $f \in F$ and over all time steps $t \in T$. For our motivating application of climate index discovery, an edge $(d_1,d_2)$ may represent a statistically significant correlation between the data at spatial points $d_1$ and $d_2$.

Informally, we define response-guided community detection as the task of partitioning graph $G$ into a set of communities $C$, such that every community $c \in C$ is strongly associated with the response variable $Y$. To quantify this association, we construct an index for each community.

**Definition 2.1 (Index).** Given a community $c$, the index constructed for $c$ using covariate $f \in F$, $I_{c,f}$, is given by

$$I_{c,f}(t) = \frac{1}{|c|} \sum_{d \in c} x_{t,d,f} \quad \forall t \in T \quad (2.1)$$

where $x_{t,d,f}$ is the value of covariate $f$ for spatial point $d$ at time step $t$. 

10
Definition 2.2 (Association). Given a community $c_i$, the association of $c_i$ with the response variable $Y$, $\phi(c_i)$, is given by

$$\phi(c_i) = \max_{f \in F} |r_{I_i,f,Y}|$$  \hspace{1cm} (2.2)

where $r_{I_i,f,Y}$ is the Pearson’s linear correlation coefficient between index $I_i,f$ and the response variable $Y$ over all time steps $t \in T$.

Definition 2.3 (Average Association). Given a set of communities $C$, the average association of $C$ with the response variable $Y$, $\bar{\phi}(C)$, is given by

$$\bar{\phi}(C) = \frac{1}{|C|} \sum_{c_i \in C} \phi(c_i)$$  \hspace{1cm} (2.3)

where $\phi(c_i)$ the association of community $c_i$ with the response variable $Y$.

Finally, we formally define the problem of response-guided community detection as follows. Given a graph $G = (V,E)$ and a response variable $Y$, partition $G$ into a set of communities $C = \{c_1, c_2, ..., c_|C|\}$, where $\bigcup_{i=1}^{|C|} c_i = V$ and $c_i \cap c_j = \emptyset$ for all $c_i, c_j \in C$ with $i \neq j$, such that the average association of $C$ with the response variable $Y$, $\bar{\phi}(C)$, is maximized.

2.3 Response-Guided Community Detection

A common approach to community detection is to find the set of communities that maximizes a given quality function that measures the “goodness” of the partition of the graph. For traditional community detection, a “good” partition of the graph is generally such that there are many edges within the communities but few edges among them. However, for response-guided community detection, the goal is to identify communities strongly associated with a response variable of interest. Therefore, we must maximize not only the “goodness” of the partition of the graph, but also the association of the communities in the partition with this variable.

To this end, we introduce a multi-objective function, $\mathcal{F}$, given by

$$\mathcal{F}(C) = \alpha \cdot q(C) + (1 - \alpha) \cdot \bar{\phi}(C)$$  \hspace{1cm} (2.4)

where $C$ is a set of communities, $q(C)$ is a function of the “goodness” of $C$, $\bar{\phi}(C)$ is the average association of the communities in $C$ with the response variable of interest, and $\alpha \in (0, 1]$ is a weighting parameter to balance the trade-off between the “goodness” of $C$ and the association of the communities with the response variable.

The “goodness” function is typically a metric that quantifies some structural property of the partition of the graph. Here, we choose modularity—“by far the most used and best known
quality function” for community detection [18]—as the “goodness” function. The modularity of a given partition of a graph is defined as the difference between the number of edges within the communities and the expected number of such edges in a random graph with the same degree distribution [39].

**Definition 2.4 (Modularity).** Let $G = (V, E)$ be an unweighted, undirected graph partitioned into a set of disjoint communities $C = \{c_1, c_2, ..., c_K\}$. The **modularity** of the partition of $G$ [38] is given by

$$Q(C) = \frac{1}{2m} \sum_{v,w \in V} \left[ A_{vw} - \frac{k_v k_w}{2m} \right] \delta(c_v, c_w)$$

(2.5)

where $A$ is the adjacency matrix of $G$ (i.e., $A_{vw} = 1$ if vertices $v$ and $w$ are adjacent and $A_{vw} = 0$ otherwise), $m = \frac{1}{2} \Sigma_{vw} A_{vw}$ is the number of edges in $G$, $k_v = \Sigma_w A_{vw}$ is the degree of vertex $v$, $c_v$ is the community of vertex $v$, and $\delta(c_v, c_w)$ is the Kronecker delta function (i.e., $\delta(c_v, c_w) = 1$ if $c_v = c_w$ and $\delta(c_v, c_w) = 0$ otherwise).

Maximizing modularity is a computationally hard problem; in fact, the decision version of the modularity maximization problem has been shown to be NP-complete [9]. However, several heuristic algorithms have been proposed to efficiently identify graph partitions with high modularity [18]. A general strategy for response-guided community detection is to adapt these modularity optimization algorithms by replacing modularity with the multi-objective function $F$ as the optimization criterion. To illustrate this strategy, we present two algorithms that can be adapted in this way to identify communities strongly associated with a response variable of interest: the Louvain method, a greedy algorithm for modularity optimization, and simulated annealing, a probabilistic optimization technique.

### 2.3.1 Greedy Algorithm for Response-Guided Community Detection

Greedy algorithms for modularity optimization identify communities by iteratively merging vertices or communities that result in the largest increase in the modularity of the graph partition [7, 10]. Here, we make use of the Louvain method [7], a well-known, single-objective greedy algorithm for modularity optimization that has been shown to outperform other community detection algorithms in empirical comparative studies [31]. We adapt the Louvain method for response-guided community detection by using the multi-objective function $F$ as the optimization criterion. The adapted algorithm (see Algorithm 2.1) proceeds as follows.

In the first phase of the algorithm, the vertices are initially assigned to singleton communities. Then, each vertex is iteratively and sequentially reassigned to the community that yields the highest positive gain in the multi-objective function $F$ until no further improvement can be achieved. The gain in the multi-objective function $F$ resulting from assigning a vertex $z$ to a
community $c_i$, $\Delta F(z, c_i)$, is given by

$$\Delta F(z, c_i) = \alpha \cdot \Delta Q(z, c_i) + (1 - \alpha) \cdot \Delta \phi(z, c_i) \quad (2.6)$$

where $\Delta Q(z, c_i)$ and $\Delta \phi(z, c_i)$ are the gain in modularity and the gain in average association with the response variable of interest, respectively.

The gain in modularity resulting from assigning vertex $z$ to community $c_i$, $\Delta Q(z, c_i)$, can be efficiently computed as follows [7].

$$\Delta Q(z, c_i) = \left[ \frac{k'_{c_i} + k_{z,c_i}}{2m} - \left( \frac{k_{c_i} + k_z}{2m} \right)^2 \right] - \left[ \frac{k'_{c_i}}{2m} - \left( \frac{k_{c_i}}{2m} \right)^2 - \left( \frac{k_z}{2m} \right)^2 \right] \quad (2.7)$$

where $k_z$ is the degree of vertex $z$, $k_{z,c_i}$ is the number of edges between vertex $z$ and vertices in community $c_i$, $k_{c_i}$ is the number of edges incident to vertices in community $c_i$, $k'_{c_i}$ is the number of edges between vertices in community $c_i$, and $m$ is the number of edges in the graph.

In the second phase of the algorithm, a new graph is constructed by aggregating the vertices in each community into a single vertex. The number of edges between two vertices in this new graph is given by the sum of the edges between vertices in the two corresponding communities.

The first phase of the algorithm is then reapplied on this new graph.

These two phases are repeated iteratively until no further changes to the community structure can be made. Then, the partition that yields the highest value of the multi-objective function $F$ for the original graph is returned. Note, however, that there is no guarantee of the optimality of the partition. Furthermore, the output of the algorithm depends on the order in which the vertices are iterated over, although empirical analysis indicates that this order does not generally have a significant impact on the value obtained for the objective function [7].

### 2.3.2 Simulated Annealing for Response-Guided Community Detection

Another strategy that has been employed for modularity optimization is simulated annealing [30], an optimization technique that avoids local optima by incorporating stochastic noise into the search procedure. The level of noise is defined by a computational temperature, $T$, which decreases after each iteration. Here, we adapt the single-objective simulated annealing algorithm proposed by Guimerà et al. [22] for response-guided community detection by using the multi-objective function $F$ as the optimization criterion. The adapted algorithm (see Algorithm 2.2) proceeds as follows.

Initially, each vertex is assigned to a singleton community. At each temperature $T$, the algorithm performs (typically) $n^2$ local moves and $n$ global moves. Local moves consist of...
moving a randomly selected vertex to a randomly selected community. Global moves consist of merging two randomly selected communities and splitting a randomly selected community.

Each local and global move is accepted with probability

\[
p = \begin{cases} 
1, & \text{if } \Delta \mathcal{F} \geq 0 \\
\exp \left( \frac{\Delta \mathcal{F}}{T} \right), & \text{if } \Delta \mathcal{F} < 0 
\end{cases}
\]  

(2.8)

where \( \Delta \mathcal{F} \) is the gain in the multi-objective function \( \mathcal{F} \) resulting from the local or global move; that is, the gain from moving a vertex \( z \) to a community \( c_i \), \( \Delta \mathcal{F}(z,c_i) \), the gain from merging two communities \( c_i \) and \( c_j \), \( \Delta \mathcal{F}(c_i,c_j) \), or the gain from splitting a community \( c_i \) in two, \( \Delta \mathcal{F}(c_i) \).

After all local and global moves have been evaluated, the current temperature \( T \) is decreased to \( T' = c \cdot T \), where \( c \in (0, 1) \) is a cooling parameter (typically between 0.990 and 0.999). The algorithm stops when a minimum temperature is reached or when there is no change in the multi-objective function \( \mathcal{F} \) for a given number of consecutive iterations.

Algorithm 2.1: Greedy algorithm for response-guided community detection

<table>
<thead>
<tr>
<th>Input</th>
<th>graph, ( G = (V,E) ); weighting parameter, ( \alpha )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>vector of community memberships, ( C )</td>
</tr>
<tr>
<td>1</td>
<td>( G_0 \leftarrow G )</td>
</tr>
<tr>
<td>2</td>
<td>do</td>
</tr>
<tr>
<td></td>
<td>/* Phase one: */</td>
</tr>
<tr>
<td>3</td>
<td>for each ( v \in V ) do</td>
</tr>
<tr>
<td>4</td>
<td>( C[v] \leftarrow v )</td>
</tr>
<tr>
<td>5</td>
<td>do</td>
</tr>
<tr>
<td>6</td>
<td>( \text{num_moves} \leftarrow 0 )</td>
</tr>
<tr>
<td>7</td>
<td>for each ( c_i \in C ) connected to ( v ) do</td>
</tr>
<tr>
<td>8</td>
<td>( \Delta \mathcal{F}(v,c_i) \leftarrow \alpha \cdot \Delta \mathcal{Q}(v,c_i) + (1 - \alpha) \cdot \Delta \phi(v,c_i) )</td>
</tr>
<tr>
<td>9</td>
<td>if ( \exists c_i \in C ) such that ( \Delta \mathcal{F}(v,c_i) &gt; 0 ) then</td>
</tr>
<tr>
<td>10</td>
<td>( C[v] \leftarrow \arg \max_{c_i \in C} \Delta \mathcal{F}(v,c_i) )</td>
</tr>
<tr>
<td>11</td>
<td>( \text{num_moves} \leftarrow \text{num_moves} + 1 )</td>
</tr>
<tr>
<td>12</td>
<td>while ( \text{num_moves} &gt; 0 )</td>
</tr>
<tr>
<td>13</td>
<td>/* Phase two: */</td>
</tr>
<tr>
<td>14</td>
<td>Compute the value of the multi-objective function, ( \mathcal{F}(C) ), for ( G_0 )</td>
</tr>
<tr>
<td>15</td>
<td>Build a new graph ( G = (V,E) ) by aggregating each ( c_i \in C ) into one vertex</td>
</tr>
<tr>
<td>16</td>
<td>while at least one vertex is moved during phase one</td>
</tr>
<tr>
<td>17</td>
<td>return ( C ) with the highest value of the multi-objective function, ( \mathcal{F}(C) ), for ( G_0 )</td>
</tr>
</tbody>
</table>
### Algorithm 2.2: Simulated annealing algorithm for response-guided community detection

**Input:** graph, $G = (V,E)$; weighting parameter, $\alpha$; cooling parameter, $c$; maximum number of iterations, $\gamma$

**Output:** vector of community memberships, $C$

1. $T \leftarrow \frac{2}{|V|}$
2. $num\_moves \leftarrow 0$
3. for each $v \in V$ do
   4. $C[v] \leftarrow v$
4. while $T > 0$ and $num\_moves < \gamma$ do
   5. /* Local moves: */
   6.   for $i \leftarrow 1$ to $|V|^2$ do
   7.       Randomly select a vertex $v \in V$ and a community $c_i \in C$
   8.       $\Delta F(v, c_i) \leftarrow \alpha \cdot \Delta Q(v, c_i) + (1 - \alpha) \cdot \Delta \phi(v, c_i)$
   9.       if $\Delta F(v, c_i) \geq 0$ or random(0,1) $\leq \exp\left(\frac{\Delta F(v, c_i)}{T}\right)$ then
     10.       Move $v$ to $c_i$
   11. /* Global moves: */
   12.   for $i \leftarrow 1$ to $|V|$ do
   13.       Randomly select two communities $c_i, c_j \in C$
   14.       $\Delta F(c_i, c_j) \leftarrow \alpha \cdot \Delta Q(c_i, c_j) + (1 - \alpha) \cdot \Delta \phi(c_i, c_j)$
   15.       if $\Delta F(c_i, c_j) \geq 0$ or random(0,1) $\leq \exp\left(\frac{\Delta F(c_i, c_j)}{T}\right)$ then
     16.       Merge $c_i$ and $c_j$
     17.       Randomly select a community $c_i \in C$
     18.       $\Delta F(c_i) \leftarrow \alpha \cdot \Delta Q(c_i) + (1 - \alpha) \cdot \Delta \phi(c_i)$
     19.       if $\Delta F(c_i) \geq 0$ or random(0,1) $\leq \exp\left(\frac{\Delta F(c_i)}{T}\right)$ then
         20.         Split $c_i$ in two communities
   21. Compute the value of the multi-objective function, $F(C)$, for $G$
   22. $T \leftarrow c \cdot T$
   23. if no move is accepted then
   24.     $num\_moves \leftarrow num\_moves + 1$
   25. else
     26.     $num\_moves \leftarrow 0$
27. return $C$ with the highest value of the multi-objective function, $F(C)$, for $G$

### 2.4 Climate Index Discovery

Response-guided community detection can be applied in the climate science domain to discover climate indices associated with a response variable of interest from multivariate spatiotemporal climate data. Our proposed methodology for climate index discovery is comprised of two main
steps (see Figure 2.1). We are given as input a spatiotemporal climate data set and a response variable of interest. First, we represent the multivariate spatiotemporal climate data as a graph using our proposed response-guided network construction methodology (Section 2.5). Second, we identify communities in this graph using one of our adapted algorithms for response-guided community detection (see Section 2.3). For each community $c_i$ identified, we return as output an index $I_{i,f_i^*}$ potentially associated with the response variable of interest, where $f_i^*$ is the representative covariate of the community for the response variable, defined as

$$f_i^* = \arg \max_{f \in F} |r_{I_{i,f}, Y}|$$

(2.9)

where $r_{I_{i,f}, Y}$ is the Pearson’s linear correlation coefficient between index $I_{i,f}$ and the response variable $Y$ over all time steps $t \in T$. 

![Figure 2.1: Overview of the proposed methodology for climate index discovery using response-guided community detection](image-url)
2.5 Response-Guided Network Construction

Spatiotemporal data can be represented as a graph, where each vertex is a spatial point and each edge indicates a significant relationship between a pair of spatial points. This type of representation has been adopted to model climate data because it captures the dynamical behavior of the climate system [56, 57, 61]. Furthermore, communities in climate networks often have a higher association with the response variable of interest than clusters in climate data obtained using spectral clustering and the $k$-means clustering algorithm [57]. This shows the potential of community detection methods over traditional clustering techniques as a means for discovering climate indices.

In this chapter, we propose a methodology for the construction of climate networks associated with a response variable of interest from multivariate spatiotemporal data. The key features of this methodology are as follows. First, we construct the network in a response-guided manner. Existing methodologies for climate network construction consider all the spatial points in the data set as vertices and build the network by computing the correlation between every pair of vertices [57, 61]. Given the high-dimensional nature of the data, this can be computationally expensive. In contrast, we only consider as vertices the spatial points associated with the response variable of interest. Second, we incorporate multiple covariates into the network construction process. Some existing methodologies have incorporated multiple covariates by defining a cross correlation function to weight the edges of the network [56]. Here, instead, we use the information of multiple covariates to assess the statistical significance of each edge in the network. Third, we incorporate spatial neighborhood information and multiple related response variables into the network construction process to increase its robustness in the case of data sets with small sample size.

2.5.1 Selecting the Set of Vertices

The set of vertices $V$ of the network is selected based on the statistical significance of the relationship between each spatial point in the data set and the response variable of interest for multiple covariates. To assess this statistical significance, we first calculate the Spearman’s rank correlation coefficients between the time series for each covariate at each spatial point and the response variable. Spearman’s rank correlation is used to capture nonlinear relationships known to exist in climate data [25].

For each spatial point $d$, the $p$-values of the Spearman’s rank correlation coefficients computed for each covariate are combined using Fisher’s $X^2$ test [17]; that is, by calculating the
\( p \)-value of the test statistic given by

\[
-2 \sum_{f \in F} \ln(p_{X_{d,f},Y})
\]  

(2.10)

where \( p_{X_{d,f},Y} \) is the \( p \)-value of the Spearman’s rank correlation coefficient between the time series for covariate \( f \) at spatial point \( d \), \( X_{d,f} \), and the response variable \( Y \) over all time steps \( t \in T \). The use of this combined probability test allows us to capture relationships between multiple covariates and the response variable. Finally, the set \( S \) of spatial points with a statistically significant combined \( p \)-value (\( p < 0.01 \)) is selected as the set of vertices \( V \) of the network. These vertices represent spatial points potentially associated with the response variable of interest.

### 2.5.2 Defining the Set of Edges

The set of edges \( E \) of the network is defined based on the statistical significance of the relationship between each pair of spatial points in \( V \) for multiple covariates. To assess this statistical significance, we first calculate the Pearson’s linear correlation coefficients between the time series for each covariate at each pair of spatial points. Despite the presence of nonlinear relationships in climate data, climate networks constructed using Pearson’s linear correlation coefficient have been shown to be highly similar to those constructed using nonlinear measures, such as mutual information [14].

For each pair of spatial points \( d_1, d_2 \in V \), the \( p \)-values of the Pearson’s linear correlation coefficients computed for each covariate are combined using Fisher’s \( X^2 \) test [17]; that is, by calculating the \( p \)-value of the test statistic given by

\[
-2 \sum_{f \in F} \ln(p_{X_{d_1,f},X_{d_2,f}})
\]  

(2.11)

where \( p_{X_{d_1,f},X_{d_2,f}} \) is the \( p \)-value of the Pearson’s linear correlation coefficient between the time series for covariate \( f \) at spatial point \( d_1 \), \( X_{d_1,f} \), and at spatial point \( d_2 \), \( X_{d_2,f} \), over all time steps \( t \in T \). Finally, an edge \( (d_1, d_2) \in E \) is defined for every pair of spatial points \( d_1, d_2 \in V \) with a statistically significant combined \( p \)-value (\( p < 10^{-10} \), as defined in previous studies [57]).

### 2.5.3 Incorporating Spatial Neighborhood Information and Multiple Related Response Variables

Data sets with small sample size, such as the ones used in this study, can often lead to the selection of spatial points with spurious associations with the response variable of interest as vertices. To increase the robustness of the vertex selection in these cases, we make use of the spatial structure of the data and the information of multiple related (i.e., highly correlated)
response variables (e.g., seasonal rainfall at multiple stations in the same region) by finding a consensus set of spatial points, $S^*$, given by

$$S^* = \bigcap_{j=1}^{h} S_j \cup \{N(d) \mid d \in S_j\}$$  \hspace{1cm} (2.12)

where $h$ is the number of response variables, $S_j$ is the set of spatial points potentially associated with the $j^{th}$ response variable and $N(d)$ indicates the spatial points spatially adjacent to spatial point $d$. We incorporate spatial neighborhood information because, given the strong spatial autocorrelations present in spatiotemporal data, it is likely that if a spatial point is associated with the response variable of interest, then its spatially adjacent points will also be associated with the response variable.

We then construct a climate network for the multiple related response variables using the previously described methodology with the consensus set of spatial points $S^*$ as the set of vertices $V$ of the network. Note that the rest of our proposed methodology for climate index discovery, including the response-guided community detection algorithms, can also be extended to incorporate multiple related response variables. In this case, the association of a community $c_i$ with the multiple related response variables, $\phi_{c_i}$, is defined as the average association of $c_i$ over all response variables $Y_j$ for $j = 1, 2, ..., h$; that is,

$$\phi(c_i) = \frac{1}{h} \sum_{j=1}^{h} \max_{f \in F} |r_{I_i,f,Y_j}|$$  \hspace{1cm} (2.13)

where $r_{I_i,f,Y_j}$ is the Pearson’s linear correlation coefficient between index $I_i,f$ and the $j^{th}$ response variable over all time steps $t \in T$.

### 2.6 Experimental Evaluation

We applied our proposed methodology to the discovery of climate indices associated with October to December (OND) rainfall variability in the GHA, using data from four (4) stations with highly correlated rainfall patterns located in the North Eastern Highlands of Tanzania (Arusha, Kilimanjaro, Moshi, and Same, see Figure 2.2). Rainfall variability has a profound socio-economic impact on the GHA because of its severe effect on agriculture production and hydro-electric power generation. Discovering climate indices strongly associated with rainfall variability in the GHA could potentially lead to more accurate forecasts of seasonal rainfall that would greatly benefit logistical planning efforts in the region by allowing policymakers to anticipate potential food and power shortages.
Figure 2.2: Map highlighting (in red) the North Eastern Highlands of Tanzania (left) and the four stations located in this region that were used for this study: Arusha, Kilimanjaro, Moshi, and Same (right)

2.6.1 Data Description

We used monthly gridded ocean data for the following climate variables: Sea Surface Temperature (SST), obtained from the NOAA Extended Reconstructed Sea Surface Temperature version 3 (ERSST V3) data set (data available from 1854 to present at 2° latitude-longitude resolution) [53], and Sea Level Pressure (SLP), Geopotential Height at 500 mb (GH), Relative Humidity at 850 mb (RH) and Precipitable Water (PW), obtained from the NCEP/NCAR Reanalysis 1 data set (data available from 1948 to present at 2.5° latitude-longitude resolution) [28]. Historically, SST, SLP, and GH have been the most frequently used variables in identifying global climate patterns. Here, we also include RH and PW as secondary variables for the temperature and water vapor content of the atmosphere to provide a basic relation to both the overall atmospheric temperature pattern [60] and to dynamical processes such as diabatic heating [45].

Monthly rainfall data (52 years, from 1960 to 2011) and seasonal rainfall forecasts (14 years, from 1998 to 2011) for stations in Tanzania were provided by the Tanzania Meteorological Agency (TMA). Data was divided into a training set (38 years, from 1960 to 1997) and a test set (14 years, from 1998 to 2011). Note that only the training set was used to construct the climate networks and discover the climate indices presented in Section 2.6.3 and Section 2.6.4, respectively.

2.6.2 Data Preprocessing

Climate data exhibits complex characteristics, such as seasonal trends and strong spatial and temporal autocorrelations [29]. To remove seasonality and minimize autocorrelations, we nor-
malized the data using monthly z-scores transformations by subtracting the mean and dividing by the standard deviation of the data over the training set [57]. Since the focus of this study is on interannual variability, we also linearly detrended the data. Furthermore, all experiments were performed using a spatial resolution of 10° latitude-longitude for the gridded ocean data.

### 2.6.3 Climate Networks Constructed

Climate networks were constructed using our proposed network construction methodology with OND rainfall variability in the GHA as the response variable of interest (see Section 2.5). To capture time-lagged relationships, which are often present in climate data, five (5) climate networks were constructed, one for each month, starting four (4) months before the season (June) until the first month of the season (October). It is worth noting that when constructing a climate network for the month of May, no spatial points were selected as potentially associated with the response variable, suggesting that this month may be too early before the season to yield significant climate indices.

Each climate network was constructed using information from four (4) related stations in the North Eastern Highlands of Tanzania. Since these stations are located in the same climatological region and exhibit highly correlated rainfall patterns, they are expected to be associated with the same global climate patterns\(^1\). Hence, the use of the consensus set allows us to filter out spatial points with potentially spurious associations with the response variable. For example, the spatial points in the Arctic and the Antarctic in Figure 2.3–two regions with no known association with rainfall variability in the GHA–are filtered out using the consensus set.

![Figure 2.3](image.png)

**Figure 2.3:** Spatial points selected in the month of October to construct the climate network. Blue triangles indicate spatial points selected only for individual stations and red circles indicate spatial points in the consensus set.

\(^1\)Interstation rainfall variability due to local factors is out of the scope of this dissertation.
2.6.4 Climate Indices Discovered

Communities associated with OND rainfall variability in the GHA were identified in the climate networks constructed using the multi-objective algorithms for response-guided community detection based on the Louvain method and simulated annealing described in Section 2.3. We set the value of the weighting parameter $\alpha$ to the multiple of 0.05 in the interval $[0.75, 1]$ that yields the set of communities with the highest average association with the response variable over the training set. Lower values of $\alpha$ were not considered to ensure that the network partition also has a high modularity value. As explained in Section 2.4, a climate index was constructed for each community identified by computing the spatial average over the community of its representative climate variable (see Figure 2.4).

We compare our climate indices with those discovered using a baseline methodology and a state-of-the-art methodology for climate index discovery [57]. For the baseline methodology, communities were identified in multivariate climate networks (i.e., one network was constructed for all covariates via a combined probability test as described in Section 2.5) using the original Louvain method [7] and the original simulated annealing algorithm for community detection [22]. For the state of the art [57], communities were identified in univariate climate networks (i.e., one network was constructed for each covariate) using Walktrap, a single-objective community detection algorithm that identifies communities by optimizing a distance metric based on random walks [46]. In both cases, the community detection and the network construction processes were performed without taking into account the response variable of interest.

![Climate indices discovered using the proposed methodology with the multi-objective algorithms for response-guided community detection based on the Louvain method (left) and simulated annealing (right) and with OND rainfall variability in the GHA as the response variable of interest. Each color represents a different index and diamonds indicate overlaps between indices. To improve visualization, only the top 10 indices with the highest association with the response variable over the training set are shown in each figure.](image)
Table 2.1 summarizes the properties of the climate networks constructed and the climate indices discovered using each methodology. Given that our response-guided community detection algorithms do not exclusively optimize the “goodness” of the network partitions, our climate networks exhibit a lower modularity than those constructed using single-objective methodologies (0.34 vs. 0.74, 0.75, and 0.59). However, our communities have a higher internal density (0.62 vs. 0.29, 0.28, and 0.47) and a lower internal variability (0.63 and 0.62 vs. 0.77, 0.78, and 0.74), indicating a well-defined structure.

We also observe that, unlike most of the climate indices discovered using the baseline and the state of the art, the majority of our climate indices (66.67%) have a statistically significant linear correlation ($p < 0.01$) with the response variable of interest over the training set. Moreover, our proposed methodology performs significantly ($p < 0.05$) better than the baseline and the state of the art across all stations in terms of the average linear correlation between the climate indices and the response variable of interest over the training set and the test set (see Table 2.2). This shows that, as expected, our proposed methodology is able to discover climate indices more strongly associated with the response variable of interest than those discovered with methodologies that make use of single-objective community detection algorithms.

Table 2.1: Properties of networks constructed and climate indices discovered for OND rainfall variability in the GHA, using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms: number of networks (Num Nets), average number of vertices and edges per network (Avg Vtxs, Avg Edges), average modularity (Avg Mod), number of indices (Num Idxs), average number of vertices, standard deviation, and internal density per index (Avg Vtxs, Avg Std, Avg Dens), and percentage of indices with a statistically significant ($p < 0.01$) linear correlation with the response variable of interest (% Idxs). Best values are highlighted in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>Algorithm</th>
<th>Networks</th>
<th>Indices</th>
<th>Significant Indices</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Num</td>
<td>Avg Vtxs</td>
<td>Avg Edges</td>
</tr>
<tr>
<td>Proposed</td>
<td>Adapted LM</td>
<td>5</td>
<td>40.80</td>
<td>169.20</td>
</tr>
<tr>
<td></td>
<td>Adapted SA</td>
<td>5</td>
<td>40.80</td>
<td>169.20</td>
</tr>
<tr>
<td>Baseline</td>
<td>Original LM</td>
<td>5</td>
<td>446.00</td>
<td>2614.60</td>
</tr>
<tr>
<td></td>
<td>Original SA</td>
<td>5</td>
<td>446.00</td>
<td>2614.60</td>
</tr>
<tr>
<td>SOTA</td>
<td>Walktrap</td>
<td>25</td>
<td>444.80</td>
<td>7493.80</td>
</tr>
</tbody>
</table>
Table 2.2: Average linear correlation with OND rainfall at each station and at the GHA region, over the training set and the test set, of climate indices discovered for OND rainfall variability in the GHA using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms. Check marks (✓) indicate that our proposed methodology performs significantly better according to a two-way ANOVA with a significance level of 0.05. Best values are highlighted in bold.

<table>
<thead>
<tr>
<th>Station</th>
<th>Proposed</th>
<th>Baseline</th>
<th>SOTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Adapted LM</td>
<td>Adapted SA</td>
<td>Original LM</td>
</tr>
<tr>
<td></td>
<td>Train</td>
<td>Test</td>
<td>Train</td>
</tr>
<tr>
<td>Arusha</td>
<td>0.4436</td>
<td>0.2999</td>
<td>0.4431</td>
</tr>
<tr>
<td>Kilimanjaro</td>
<td>0.4103</td>
<td><strong>0.3752</strong></td>
<td>0.4300</td>
</tr>
<tr>
<td>Moshi</td>
<td>0.3629</td>
<td><strong>0.2980</strong></td>
<td><strong>0.3764</strong></td>
</tr>
<tr>
<td>Same</td>
<td>0.4292</td>
<td><strong>0.3403</strong></td>
<td><strong>0.4341</strong></td>
</tr>
<tr>
<td>GHA</td>
<td>0.4502</td>
<td><strong>0.3478</strong></td>
<td><strong>0.4614</strong></td>
</tr>
</tbody>
</table>

Two-way ANOVA (α = 0.05) ✓ ✓ ✓ ✓ ✓ ✓

2.6.5 Seasonal Rainfall Prediction

We validate the climate indices discovered using our proposed methodology by assessing their predictive power for OND rainfall in the GHA. To this end, we trained linear regression models to predict rainfall at each station and average rainfall at the region using our climate indices as predictors. As specified in Section 2.6.1, data from 1960 to 1997 was used for training and data from 1998 to 2011 was used for testing. For comparison purposes, linear regression models were also built using the climate indices discovered with the baseline and state-of-the-art [57] methodologies introduced in Section 2.6.4.

In order to avoid overfitting, given the small sample size of the data sets, only the top six (6) climate indices with the highest average correlation with OND rainfall in the GHA over the training set were used to build the models. This number of predictors was selected because it yielded relatively stable performance over the training set across all methodologies (see Figure 2.5). Furthermore, to evaluate the ability of the models to make predictions before the start of the OND rainfall season, all experiments were performed using data up to the month of August (one-month lead time). Climate indices discovered for the months of September and October were reconstructed using August data.

The correlations between predicted and true rainfall and the root mean squared errors (RMSE) obtained for each methodology are shown in Table 2.3. We observe that the models built using our climate indices yield a significantly (p < 0.05) higher correlation and lower RMSE...
than those built using climate indices discovered with methodologies that make use of single-objective community detection algorithms (see Figure 2.6). This suggests that climate indices more strongly associated with the response variable of interest, such as the ones discovered using our proposed methodology, have greater predictive power.

Figure 2.5: Average linear correlation between true and predicted rainfall for predictions of OND rainfall at each station in the GHA region over the training set using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms vs. the number of predictors used to build the regression models. The dashed line indicates the number of predictors selected for further analysis.

Table 2.3: Linear correlation between true and predicted rainfall (Corr) and RMSE scores for predictions of OND rainfall at each station and at the GHA region from 1998 to 2011 obtained using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms. Check marks (√) indicate that our proposed methodology performs significantly better according to a two-way ANOVA with a significance level of 0.05. Best values are highlighted in bold.

<table>
<thead>
<tr>
<th>Station</th>
<th>Proposed</th>
<th>Baseline</th>
<th>SOTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Adapted LM</td>
<td>Adapted SA</td>
<td>Original LM</td>
</tr>
<tr>
<td></td>
<td>Corr</td>
<td>RMSE</td>
<td>Corr</td>
</tr>
<tr>
<td>Arusha</td>
<td>0.7143</td>
<td>0.5017</td>
<td>0.5869</td>
</tr>
<tr>
<td>Kilimanjaro</td>
<td>0.7629</td>
<td>0.5034</td>
<td>0.6736</td>
</tr>
<tr>
<td>Moshi</td>
<td>0.6561</td>
<td>0.4719</td>
<td><strong>0.6564</strong></td>
</tr>
<tr>
<td>Same</td>
<td><strong>0.7237</strong></td>
<td>0.4779</td>
<td>0.6896</td>
</tr>
<tr>
<td>GHA</td>
<td><strong>0.7722</strong></td>
<td>0.4133</td>
<td>0.7425</td>
</tr>
</tbody>
</table>

Two-way ANOVA ($\alpha = 0.05$) | ✓ | ✓ | ✓ | ✓ | ✓ | ✓
Figure 2.6: Linear correlation between true and predicted rainfall and RMSE scores for predictions of OND rainfall at each station in the GHA region from 1998 to 2011 obtained using the proposed, baseline, and state-of-the-art (SOTA) [57] methodologies with the Louvain method (LM), simulated annealing (SA) and Walktrap as the community detection algorithms.

We further assess the predictive power of our climate indices by comparing our predictions with the official forecasts of the OND rainfall season issued by the TMA every year on September. To this end, the rainfall season for each year was categorized according to the guidelines of the TMA as below normal, normal, or above normal (rainfall below 75%, between 75% and 125%, or above 125% of long-term averages, respectively). Long-term averages were computed using the training set. Decision trees to classify the OND rainfall season at each station were trained using data up to the month of August and considering only the top six (6) climate indices discovered using our proposed methodology as predictors. The decision trees were built using the Gini index as the split criterion and were pruned to avoid overfitting.

The classification accuracies obtained are shown in Figure 2.7. We observe that the accuracy of the decision trees built using our climate indices is higher than that of the official forecasts for three (3) out of four (4) stations. This suggests that the use of the climate indices discovered using our proposed methodology can potentially improve forecasts of the response variable of interest.
2.6.6 Physical Interpretation of Climate Indices Discovered

Finally, we discuss the climate indices discovered in terms of their climatological relevance. Rainfall variability in the GHA is known to be mainly associated with ENSO in the equatorial Pacific Ocean [42] and the Indian Ocean Dipole (IOD) in the tropical Indo-Pacific Ocean [6].

Climate indices significantly correlated \( (p < 0.01) \) with ENSO, in particular with the Niño 3.4 index, were discovered in June, July, August, September, and October using both proposed multi-objective algorithms for response-guided community detection (for example, see Figure 2.8). The representative climate variable selected for these climate indices is mostly either SST or PW. SST is the representative climate variable of ENSO, while PW is a close proxy of SST in the equatorial Pacific Ocean in the NCAR/NCEP Reanalysis 1 data set [59]. The relationship between ENSO and rainfall variability in the GHA is well established in climate literature [42]. Higher SSTs in the equatorial Pacific Ocean are associated with a suppression of East African rainfall, by modulating the strength of the global upper level wind flow.

Climate indices significantly correlated \( (p < 0.01) \) with the IOD were discovered in July, August, September and October using both proposed multi-objective algorithms for response-guided community detection (for example, see Figure 2.8). These climate indices were generally discovered closer to the onset of the OND rainfall season than the ones in the equatorial Pacific Ocean, as the IOD exerts its influence on East African rainfall on a shorter timescale through local wind anomalies [6].
2.7 Conclusion

In this chapter, we introduced the problem of response-guided community detection through its application to the task of climate index discovery. We proposed a methodology for the discovery of climate indices associated with a response variable of interest from multivariate spatiotemporal data, the contribution of which is twofold. First, we proposed a general multi-objective strategy for response-guided community detection, and second, we introduced a network representation of the data that incorporates information from multiple variables.

We applied our proposed methodology to the discovery of climate indices associated with seasonal rainfall variability in the GHA. The climatological relevance of the climate indices discovered is supported by domain knowledge, as evidenced by their association with traditional climate indices known to be related to seasonal rainfall in the region. Furthermore, our results show that our methodology is able to improve the forecast skill for this response variable with respect to existing methodologies for climate index discovery that make use of single-objective community detection algorithms, as well as official forecasts.
Chapter 3

Detecting Communities in Dynamic Attributed Graphs

3.1 Introduction

Communities are traditionally defined as sets of vertices more densely connected among each other than to the rest of the graph [18, 23]. Therefore, as explained in Chapter 1 and Chapter 2, most community detection methods identify communities using single-objective algorithms that only take into account the structural properties of the graph. However, real-world systems represented as complex networks inherently have multiple sources of information.

On one hand, objects in complex networks often have individual properties, or attributes, that characterize them. By describing these objects as unattributed vertices we are potentially losing valuable information that may allow us to identify more meaningful communities [8, 64]. For this reason, an active line of research in community detection focuses on identifying communities in attributed graphs by incorporating both the structural properties of the graph and the attribute information of the vertices [11, 12, 16, 48, 64, 65].

Moreover, complex networks are constantly evolving. Hence, another active line of research in community detection is concerned with identifying communities in dynamic graphs [3, 13]. However, to the best of our knowledge, none of the methods proposed for community detection in dynamic graphs also incorporate attribute information, nor do they address the additional challenges inherent to the evolution of attributed graphs, such as the presence of changes in the attributes of the vertices. For this reason, in this chapter, we propose a multi-objective algorithm for community detection in dynamic attributed graphs that identifies communities by taking into account both the structural properties of the graph and the attribute information of the vertices and then updates and tracks these communities based on changes to both the structure and the attribute information at each time step.
The main contributions of this chapter are as follows. First, we propose an efficient multi-objective algorithm for community detection in dynamic attributed graphs (Section 3.4). Unlike existing community detection methods for attributed [11, 48] or dynamic [3, 13] graphs, our proposed algorithm handles graphs with heterogeneous attribute types, as well as changes to both the structure and the attribute information, which is essential for its applicability to real-world networks. Second, we introduce a methodology to generate benchmark dynamic attributed graphs for testing community detection algorithms (Section 3.4.5).

The rest of this chapter is organized in the following way. In Section 3.2, we present an overview of existing methods for community detection in attributed graphs and highlight their limitations. In Section 3.3, we formally define the problem of community detection in dynamic attributed graphs. In Section 3.4, we describe our proposed multi-objective algorithm for community detection in dynamic attributed graphs and introduce a methodology to generate benchmark dynamic attributed graphs. In Section 3.5, we discuss the results obtained using our proposed algorithm on a variety of synthetically generated benchmark graphs and large-scale real-world networks. Finally, we present our conclusions in Section 3.6.

3.2 Related Work

Several methods have been proposed to identify communities in attributed graphs [11, 12, 16, 48, 64, 65]. One approach is to augment the structure of the graph with the attribute information of the vertices and then apply a single-objective community detection method on the augmented graph. For example, the Inc-Cluster algorithm [65] adds new vertices representing attribute values and then clusters the augmented graph using the \( k \)-medoids clustering algorithm with a distance metric based on random walks. Similarly, the CODICIL algorithm [48] adds new edges between vertices with similar attribute values and then applies any given community detection method on a simplified version of the augmented graph. Another approach is to extend single-objective community detection methods to take into account both the structure of the graph and the attribute information of the vertices. For example, the I-Louvain algorithm [11] extends the Louvain method [7] by introducing an inertia-based modularity function that measures the similarity between vertices based on the Euclidean distance between their attribute vectors.

However, these existing methods have important limitations. First, none of these methods explicitly identify communities in dynamic graphs, and thus would need to be run from scratch every time the graph changes. Second, some of these methods require computing pairwise similarities between all vertices in the graph [11, 48], which leads to high time and space complexities that may not be suitable for large-scale networks. Third, most of these methods are designed to handle only certain types of attributes, whether binary, categorical [48, 65] or numeric [11], which hinders their applicability to real-world networks with heterogeneous attribute types. We
address each of these limitations in Section 3.4 with our proposed multi-objective algorithm for community detection in dynamic attributed graphs. Finally, it is worth noting that the focus of this dissertation is on detection of disjoint communities; therefore, existing methods to identify overlapping communities in attributed graphs [64] were not considered.

3.3 Problem Statement

Let $G = (V, E, X)$ be an attributed graph, where $V$ is the set of vertices, $E$ is the set of edges, and $X = \{x_1, x_2, ..., x_D\}$ is the set of attributes associated with the vertices in $V$. We define a dynamic attributed graph as an ordered set $\mathcal{G} = \{G_t\}_{t=1}^T$, where $T$ is the number of time steps and $G_t$ is the attributed graph at time step $t$. Note that we assume that corresponding vertices are labeled consistently across different time steps.

The problem of community detection in dynamic attributed graphs is defined as follows. Given a dynamic attributed graph $\mathcal{G}$, partition each attributed graph $G_t \in \mathcal{G}$ into a set of disjoint communities $C_t = \{c_{1t}, c_{2t}, ..., c_{K_t}\}$, where $\bigcup_{i=1}^{K_t} c_{it} = V_t$ and $c_{it} \cap c_{jt} = \emptyset$ for all $c_{it}, c_{jt} \in C_t$ with $i \neq j$, such that vertices in the same community are more densely connected between each other and have more similar attribute values than those in different communities.

3.4 Community Detection in Dynamic Attributed Graphs

In this section, we describe our proposed multi-objective algorithm for community detection in dynamic attributed graphs and introduce a methodology to generate benchmark dynamic attributed graphs.

3.4.1 Measuring the Quality of the Partition of the Graph

As explained in Chapter 2, a common approach to community detection is to partition the graph into communities by optimizing a given quality function that measures the “goodness” of the partition of the graph. A “good” partition of an attributed graph is one that maximizes both the structural quality of the partition and the similarity between the attributes of the vertices in each community. As in Chapter 2, we use the well-known modularity function (see Definition 2.4) [39] to measure the structural quality of the partition of the graph. To measure the similarity between the attributes of the vertices in each community, we define the attribute similarity function as follows.

**Definition 3.1 (Attribute Similarity).** Let $G = (V, E, X)$ be an attributed graph partitioned into a set of disjoint communities $C = \{c_1, c_2, ..., c_K\}$. The attribute similarity of the parti-
The function \( S(C) \) is given by
\[
S(C) = \frac{1}{K \cdot D} \left[ \sum_{c_i \in C} \sum_{v,w \in c_i} \sum_{j=1}^{D} \frac{\text{sim}(x_{jv}, x_{jw})}{|c_i|^2} \right]
\] (3.1)
where \( K = |C| \) is the number of communities, \( D = |X| \) is the number of attributes in \( G \), \( x_{jv} \) is the value of the \( j \)-th attribute for vertex \( v \), and \( \text{sim}(x_{jv}, x_{jw}) \) is a function of the similarity between \( x_{jv} \) and \( x_{jw} \).

For binary attributes, \( \text{sim}(x_{jv}, x_{jw}) \) is given by the simple matching coefficient between \( x_{jv} \) and \( x_{jw} \); that is,
\[
\text{sim}(x_{jv}, x_{jw}) = \frac{\sum_{k=1}^{d} (x_{jkv} \land x_{jkw}) \lor (\neg x_{jkv} \land \neg x_{jkw})}{d}
\] (3.2)

For categorical attributes, \( \text{sim}(x_{jv}, x_{jw}) \) is given by the Jaccard similarity index between the “1-of-N” binary encodings of \( x_{jv} \) and \( x_{jw} \); that is,
\[
\text{sim}(x_{jv}, x_{jw}) = \frac{\sum_{k=1}^{d} x_{jkv} \land x_{jkw}}{\sum_{k=1}^{d} x_{jkv} \lor x_{jkw}}
\] (3.3)

For numeric attributes, \( \text{sim}(x_{jv}, x_{jw}) \) is given by the inverse of one plus the Euclidean distance between \( x_{jv} \) and \( x_{jw} \); that is,
\[
\text{sim}(x_{jv}, x_{jw}) = \frac{1}{1 + \sqrt{\sum_{k=1}^{d} (x_{jkv} - x_{jkw})^2}}
\] (3.4)

where \( d \) is the number of dimensions of the \( j \)-th attribute, \( x_{jkv} \) is the value of the \( k \)-th coordinate of the \( j \)-th attribute for vertex \( v \), and \( \neg, \land \) and \( \lor \) are the logical NOT, AND, and OR operators, respectively.

Our proposed attribute similarity function differs from other similarity measures for community detection in attributed graphs in two important ways. First, unlike the similarity measure by Dang and Viennet [12], our proposed attribute similarity function is normalized by the number and size of the communities so that its range is comparable to that of modularity. Second, unlike the similarity measure by Combe et al. [11], which is only suitable for numeric attributes, our proposed attribute similarity function allows us to combine attributes of different types, which is essential given the heterogeneous nature of many real-world networks.
3.4.2 Detecting Communities in Attributed Graphs

The goal of our proposed multi-objective algorithm for community detection in attributed graphs is to maximize both the modularity and the attribute similarity of the partition of the graph. As discussed in Chapter 2, maximizing modularity is a computationally hard problem \([9]\), but several heuristic algorithms have been proposed to efficiently identify graph partitions with high modularity \([18]\). Here, we adapt the Louvain method introduced in Chapter 2 \([7]\), originally a single-objective modularity optimization algorithm, to detect communities in attributed graphs by optimizing a multi-objective function, \(\mathcal{F}\), given by

\[
\mathcal{F}(C) = \alpha \cdot Q(C) + (1 - \alpha) \cdot S(C)
\]  

(3.5)

where \(C\) is a set of communities, \(Q\) is the modularity function, \(S\) is the attribute similarity function, and \(\alpha \in (0, 1]\) is a weighting parameter to balance the trade-off between modularity and attribute similarity. Note that when \(\alpha = 1\), the adapted algorithm reduces to the original single-objective modularity optimization algorithm.

The adapted algorithm for detecting communities in attributed graphs (see Algorithm 3.1) proceeds as follows. In the first phase of the algorithm, the vertices are initially assigned to singleton communities. Then, each vertex is iteratively and sequentially reassigned to the community that yields the highest positive gain in the multi-objective function \(\mathcal{F}\) until no further improvement can be achieved. The gain in the multi-objective function \(\mathcal{F}\) resulting from assigning a vertex \(z\) to a community \(c_i\), \(\Delta \mathcal{F}(z, c_i)\), is given by

\[
\Delta \mathcal{F}(z, c_i) = \alpha \cdot \Delta Q(z, c_i) + (1 - \alpha) \cdot \Delta S(z, c_i)
\]  

(3.6)

where \(\Delta Q(z, c_i)\) and \(\Delta S(z, c_i)\) are the gain in modularity and the gain in attribute similarity resulting from assigning a vertex \(z\) to a community \(c_i\), respectively. The gain in modularity, \(\Delta Q(z, c_i)\), is given by Equation 2.7 and the gain in attribute similarity, \(\Delta S(z, c_i)\), is given by

\[
\Delta S(z, c_i) = \frac{1}{D} \sum_{j=1}^{D} \sum_{c_k \in C'} \sum_{v,w \in c_k} \frac{\text{sim}(x_{jv}, x_{jw})}{|c_k|^2} - \frac{1}{K} \sum_{c_k \in C} \sum_{v,w \in c_k} \frac{\text{sim}(x_{jv}, x_{jw})}{|c_k|^2}
\]  

(3.7)

where \(D\) is the number of attributes, \(K = |C|, K' = |C'|\), and \(C\) and \(C'\) are the sets of communities before and after assigning \(z\) to \(c_i\), respectively.

Computing the exact gain in attribute similarity in Equation 3.7 requires computing similarities for every community every time a vertex changes its community membership, which would significantly increase the time complexity of the algorithm. Here, instead, we approximate \(K\) as \(K'\) to efficiently estimate the gain in attribute similarity resulting from assigning a vertex \(z\)
to a community $c_i$, $\Delta S(z, c_i)$, as follows.

$$
\Delta S(z, c_i) \approx \frac{1}{K' \cdot D} \sum_{j=1}^{D} \left[ \sum_{c_k \in C} \sum_{v, w \in c_k} \frac{\text{sim}(x_{jv}, x_{jw})}{|c_k|^2} - \sum_{c_k \in C} \sum_{v, w \in c_k} \frac{\text{sim}(x_{jv}, x_{jw})}{|c_i|^2} \right]
$$

$$
\approx \frac{1}{K' \cdot D} \sum_{j=1}^{D} \left[ \sum_{v, w \in c_i \cup \{z\}} \frac{\text{sim}(x_{jv}, x_{jw})}{(|c_i| + 1)^2} - \sum_{v, w \in c_i} \frac{\text{sim}(x_{jv}, x_{jw})}{|c_i|^2} \right]
$$

(3.8)

In the second phase of the algorithm, a new graph is constructed by aggregating the vertices in each community into a single vertex. The number of edges between two vertices in this new graph is given by the sum of the edges between vertices in the two corresponding communities. Likewise, the attribute values of a vertex in this new graph are given by the modes (in the case of binary or categorical attributes) or the means (in the case of numeric attributes) of the attribute values of the vertices in the corresponding community. The first phase of the algorithm is then reapplied on this new graph.

---

**Algorithm 3.1: Greedy algorithm for community detection in attributed graphs**

**Input:** attributed graph at time $t$, $G_t = (V_t, E_t, X_t)$; vector of initial community memberships at time $t$, $C$; weighting parameter, $\alpha$

**Output:** vector of community memberships at time $t$, $C$

1. $G = (V, E, X) \leftarrow G_t$
2. do
   /* Phase one: */
   3. do
      4. $\text{num}_\text{moves} \leftarrow 0$
      5. for each $v \in V$
         6. for each community $c_i \in C$ connected to $v$
            7. $\Delta \mathcal{F}(v, c_i) \leftarrow \alpha \cdot \Delta \mathcal{Q}(v, c_i) + (1 - \alpha) \cdot \Delta S(v, c_i)$
            8. if $\exists c_i \in C$ such that $\Delta \mathcal{F}(v, c_i) > 0$ then
               9. $C[v] \leftarrow \text{arg max}_{c_i \in C} \Delta \mathcal{F}(v, c_i)$
               10. $\text{num}_\text{moves} \leftarrow \text{num}_\text{moves} + 1$
      11. while $\text{num}_\text{moves} > 0$
   /* Phase two: */
   12. Compute the value of the multi-objective function, $\mathcal{F}(C)$, for $G_t$
   13. Build a new graph $G = (V, E, X)$ by aggregating each $c_i \in C$ into one vertex
   14. for each $v \in V$
      15. $C[v] \leftarrow v$
   16. while at least one vertex is moved during phase one
   17. return $C$ with the highest value of the multi-objective function, $\mathcal{F}(C)$, for $G_t$
These two phases are repeated iteratively until no further changes to the community structure can be made. Then, the partition that yields the highest value of the multi-objective function $F$ for the original graph is returned. As mentioned in Chapter 2, there is no guarantee of the optimality of the partition and the output of the algorithm depends on the order in which the vertices are iterated over, although this order does not generally have a significant impact on the value obtained for the objective function [7].

### 3.4.3 Updating Communities in Dynamic Graphs

A common approach for finding communities in dynamic graphs is to iteratively apply a community detection algorithm on each graph $G_t$ at each time step $t$ [3, 13]. However, it is not necessary to run the algorithm from scratch at each time step to obtain a “good” partition of the graph. Initializing the partition with information from the previous time step before applying the community detection algorithm may be more efficient and has been shown to yield more stable communities across time steps [3].

Naively, we can initialize the partition of the graph at time step $t$ with the final partition at the previous time step $t-1$, but this constraint may limit the ability of the algorithm to find “good” communities, particularly if the graph changes significantly between time steps. Other approaches include randomly removing vertices from the partition [3], or removing all vertices that changed between time steps $t-1$ and $t$ [13]. However, none of these approaches take into account the differences in the degree of change of the vertices. Vertices with a higher degree of change are more likely to have changed communities between time steps, and thus, should be removed from the partition with a higher probability.

For this reason, we define a scoring function to measure the degree of change of a vertex between two consecutive time steps, $t-1$ and $t$, in terms of both its structure and its attribute information, where the trade-off between structural and attribute change is given by the weighting parameter $\alpha$ (see Definition 3.2).

**Definition 3.2 (Vertex Change Score)** Let $G_t = (V_t, E_t, X_t)$ and $G_{t-1} = (V_{t-1}, E_{t-1}, X_{t-1})$ be attributed graphs at time steps $t$ and $t-1$, respectively. The change score of a vertex $z$, such that $z \in V' = V_t \cap V_{t-1}$, is given by

$$
H(z) = \alpha \left[ \frac{\sum_{v \in V'} (A_{tvz} \land A_{tvz-1})}{\sum_{v \in V'} (A_{tvz} \lor A_{tvz-1})} \right] + (1 - \alpha) \left[ 1 - \frac{\sum_{j=1}^{D} \text{sim}(x_{tjz}, x_{t-1jz})}{D} \right] \quad (3.9)
$$

where $A_t$ is the adjacency matrix of $G_t$, $x_{tjz}$ is the value of the $j$-th attribute for vertex $z$ in $G_t$, and $\land$ and $\lor$ are the logical AND and OR operators, respectively.

We use this scoring function to identify a partial partition of the graph based on the degree of
change of each vertex and a user-defined update parameter $\beta \in [0, 1]$. Our proposed algorithm for updating communities in dynamic graphs (see Algorithm 3.2) proceeds as follows. First, we set the current communities at time step $t$ to be those at the previous time step $t-1$; that is, $C_t = C_{t-1}$. If $\beta = 0$, all vertices remain in their current communities. If $\beta = 1$, all vertices are removed from their current communities and assigned to singleton communities. If $\beta \in (0, 1)$, the top $\beta \times |V'|$ vertices with the highest change score are removed from their current communities and assigned to singleton communities, while the other $(1 - \beta) \times |V'|$ vertices remain in their current communities. Note that this only applies to vertices present in both $V_{t-1}$ and $V_t$. Vertices present in $V_t$ but not in $V_{t-1}$ are assigned to singleton communities without preprocessing. Finally, we apply our proposed multi-objective algorithm for community detection in attributed graphs (see Algorithm 3.1) with the current communities as the initial partition of the graph.

3.4.4 Tracking Communities in Dynamic Attributed Graphs

A common approach for tracking communities over time is to identify communities at each time step and then match the communities identified at consecutive time steps [2, 21, 43]. As this matching may not be feasible for large-scale networks, existing community tracking frameworks employ heuristic methods to match communities. For example, Greene, Doyle, and Cunningham match a community $c_i \in C_t$ with a community at the following time step $c_j \in C_{t+1}$

\begin{algorithm}
\caption{Algorithm for community updating in dynamic graphs}
\begin{algorithmic}[1]
\Require attributed graph at time $t$, $G_t = (V', E_t, X_t)$; attributed graph at time $t-1$, $G_{t-1} = (V', E_{t-1}, X_{t-1})$; vector of community memberships at time $t-1$, $C_{t-1}$; weighting parameter, $\alpha$; update parameter, $\beta$
\Ensure vector of community memberships at time $t$, $C_t$
1 $C_t \leftarrow C_{t-1}$
2 \If{$\beta == 1$}
3 \For{each $v \in V$}
4 \State $C_t[v] \leftarrow v$
5 \EndFor
6 \ElseIf{$\beta > 0$ and $\beta < 1$}
7 \For{each $v \in V$}
8 \State Compute $\mathcal{H}(v)$
9 \State Let $H$ be the top $\beta \times |V'|$ vertices with the highest change score $\mathcal{H}$
10 \For{each $v \in H$}
11 \State $C_t[v] \leftarrow \max(C_t) + 1$
12 \EndFor
13 \EndIf
14 $C_t \leftarrow \text{Algorithm 3.1}(G_t, C_t, \alpha)$
15 \Return $C_t$
\end{algorithmic}
\end{algorithm}
if the Jaccard similarity index between them is greater than a threshold parameter \( \theta \in [0, 1] \) [21]; that is, communities \( c_i \) and \( c_j \) are matched if

\[
\frac{|c_i \cap c_j|}{|c_i \cup c_j|} > \theta
\]  

(3.10)

To avoid the need for an additional parameter, we simplify this framework by matching a community \( c_i \in C_t \) with a single community at the following time step \( c_j \in C_{t+1} \) that contains the largest number of vertices from \( c_i \); that is, \( c_i \) is matched with the community given by

\[
\arg \max_{c_j \in C_{t+1}} \frac{|c_i \cap c_j|}{|c_i|}
\]  

(3.11)

Further analysis of the communities matched at consecutive time steps allows us to characterize their evolution over time. Current research in community tracking has identified six key events in the evolution of a community: birth, death, merging, splitting, expansion, and contraction [43]. However, these events only consider changes in the structural properties of the communities and do not take into account changes in other sources of information, such as the attribute information of the vertices. Here, instead, we identify changes to both the structure of the communities and the attributes of the vertices, which provides additional information to characterize the evolution of the communities over time.

### 3.4.5 Generating Benchmark Dynamic Attributed Graphs

To evaluate our proposed community detection algorithm, we introduce a methodology to generate benchmark dynamic attributed graphs. This methodology extends current benchmarks for testing community detection algorithms in static graphs [16, 32]. Our benchmark graph generation methodology is as follows.

For \( t = 0 \):

1. Generate an unattributed graph \( G_0 \) with \( n \) vertices partitioned into a set of disjoint communities \( C_0 \), such that the degree of the vertices of \( G_0 \) and the sizes of the communities in \( C_0 \) follow power law distributions, as described in [32]. The partitioning of \( G_0 \) is controlled by a user-defined mixing parameter \( \mu \in [0, 1] \), which indicates the fraction of edges of each vertex with vertices outside its community.

2. For each vertex \( v \), generate a set of \( D \) attribute values \( \{x_{1v}, x_{2v}, ..., x_{Dv}\} \), such that each attribute value is sampled from a uniform distribution and all the vertices in the same community are assigned the same attribute value. For this dissertation, we limit our
benchmark graph generation methodology to binary attributes. However, it could be further extended to include categorical and numeric attributes.

3. For each attribute $i$, draw a random sample of vertices $S$ of size $\lambda \times n$, where $\lambda \in [0,1]$ is a user-defined noise parameter. Introduce noise to attribute $i$ by changing the value of $x_{iv}$ for every vertex $v \in S$.

For $t = 1$ to $T$, where $T$ is the number of time steps:

1. Set $G_t = G_{t-1}$ and $C_t = C_{t-1}$.
2. Draw a random sample of vertices $S$ of size $\delta \times n$, where $\delta \in [0,1]$ is a user-defined change parameter.
3. Move every vertex $v \in S$ to a different community.
4. Modify the edges of every vertex $v \in S$ such that $((1 - \mu) \times 100)\%$ of its neighbors belong to its new community. This ensures that the structure of $G_t$ remains closely defined by the mixing parameter $\mu$.
5. Modify the attribute values of every vertex $v \in S$. For each attribute $i$, $v$ is assigned attribute value $x_{iv}$ such that

$$
\begin{align*}
    x_{iv} &= x_c \text{ with probability } 1 - \lambda \\
    x_{iv} &\neq x_c \text{ with probability } \lambda
\end{align*}
$$

where $x_c$ is the mode of the values for attribute $i$ among the vertices in the new community of $v$. This ensures that the noise in $G_t$ remains closely defined by the noise parameter $\lambda$.

3.5 Experimental Evaluation

We evaluated the accuracy and scalability of our proposed multi-objective algorithm for community detection in dynamic attributed graphs on a variety of synthetically generated benchmark graphs (Section 3.5.1) and large-scale real-world networks (Section 3.5.2).

3.5.1 Benchmark Graphs

We evaluated the performance of our proposed algorithm on dynamic attributed graphs generated using our benchmark graph generation methodology (see Section 3.4.5). We generated a diverse set of graphs by varying the number of vertices ($n = \{1000, 10000\}$), the mixing parameter ($\mu = \{0.1, 0.9\}$), the noise parameter ($\lambda = \{0.1, 0.9\}$), and the change parameter
(\delta = \{0.1, 0.9\})). For simplicity, we considered the number of attributes \((D = 10)\) and the number of time steps \((T = 10)\) to be fixed. Moreover, for each combination of the parameters, we generated ten \((10)\) dynamic attributed graphs to account for non-determinism in the generation process. The average results obtained are presented in the following sections.

**Analysis of Weighting Parameter \(\alpha\)**

To analyze the impact of the weighting parameter \(\alpha\) on the performance of our proposed algorithm, we considered the change parameter \(\delta\) to be fixed and varied the mixing parameter \(\mu\) and the noise parameter \(\lambda\). The modularity and attribute similarity of the graph partitions obtained are shown in Figure 3.1 and Figure 3.2, respectively.

When the mixing parameter is low \((\mu = 0.1)\), the best partition of the graph is clearly defined by its structural properties. For this reason, incorporating attribute information does not have an impact on the performance of the algorithm, which remains constant regardless of the value of \(\alpha\). It is worth noting that, in these cases, incorporating attribute information

![Figure 3.1: Average modularity of the partitions obtained at each time step of the benchmark graphs using our proposed algorithm with update parameter \(\beta = 1\) and multiple values of the weighting parameter \(\alpha\). Benchmark graphs were generated using the following parameters: number of vertices \(n = 1000\), mixing parameter \(\mu = \{0.1, 0.9\}\), noise parameter \(\lambda = \{0.1, 0.9\}\), and change parameter \(\delta = 0.1\).](image)
Figure 3.2: Average attribute similarity of the partitions obtained at each time step of the benchmark graphs using our proposed algorithm with update parameter $\beta = 1$ and multiple values of the weighting parameter $\alpha$. Benchmark graphs were generated using the following parameters: number of vertices $n = 1000$, mixing parameter $\mu = \{0.1, 0.9\}$, noise parameter $\lambda = \{0.1, 0.9\}$, and change parameter $\delta = 0.1$.

does not degrade the quality of the communities with respect to the original single-objective modularity optimization algorithm ($\alpha = 1$), even if the noise parameter is high ($\lambda = 0.9$).

On the other hand, when the mixing parameter is high ($\mu = 0.9$), the best partition of the graph is not clearly defined by its structural properties. Thus, the partition obtained by the original single-objective modularity optimization algorithm ($\alpha = 1$) is of low modularity and low attribute similarity. In these cases, incorporating attribute information improves the quality of the communities in terms of attribute similarity at the expense of a minimal decrease in modularity. As expected, this improvement is inversely proportional to the value of $\alpha$.

These results are in agreement with previous studies that indicate that the performance of structure-only community detection methods degrades with respect to that of attribute-only community detection methods when the graph has an ambiguous structure (i.e., mixing parameter of 0.6 or greater) [16].
Analysis of Update Parameter $\beta$

To analyze the impact of the update parameter $\beta$ on the performance of our proposed algorithm, we considered the mixing parameter $\mu$ and the noise parameter $\lambda$ to be fixed and varied the change parameter $\delta$. The running time of our proposed algorithm and the modularity and attribute similarity of the graph partitions obtained are shown in Figure 3.3 and Figure 3.4, respectively.

We observe that initializing the partition of the graph at each time step before applying the community detection algorithm ($\beta < 1$) generally results in lower running times at a minimal performance cost in terms of modularity and attribute similarity. This is particularly true when the change parameter is low ($\delta = 0.1$), as the communities at time step $t$ are expected to be very similar to those at time step $t - 1$, and less so as the change parameter increases ($\delta = 0.9$).

![Running Time for $\mu = 0.1, \lambda = 0.1, \alpha = 0.5$](image)

Figure 3.3: Average running time (in seconds) at each time step of the benchmark graphs of our proposed algorithm with weighting parameter $\alpha = 0.5$ and multiple values of the update parameter $\beta$. Benchmark graphs were generated using the following parameters: number of vertices $n = \{1000, 10000\}$, mixing parameter $\mu = 0.1$, noise parameter $\lambda = 0.1$, and change parameter $\delta = \{0.1, 0.9\}$.
Parameter Selection

The analysis of the weighting parameter $\alpha$ suggests that its optimal value is inversely proportional to the mixing parameter $\mu$. Similarly, the analysis of the update parameter $\beta$ suggests that its optimal value is directly proportional to the change parameter $\delta$. However, the value of both $\mu$ and $\delta$ is given by the true partition of the graph, which is not known at the time of the parameter selection. For this reason, to select the optimal value of the weighting parameter $\alpha$ and the update parameter $\beta$, we would first need to estimate the value of the mixing parameter and the change parameter from the observable properties of the graph.

The mixing parameter has been shown to have a strong negative correlation with the local clustering coefficient [44]. Therefore, it can be estimated as a function of one minus the local clustering coefficient of the graph (see Figure 3.5).

To estimate the change parameter, we first observe that the distribution of the vertex change scores at a time step $t$ of the benchmark graphs is generally bimodal (see Figure 3.6). More precisely, it is a two-component mixture distribution, where the component with the higher
Figure 3.5: Box plots of estimated mixing parameter (left) and estimated change parameter (right) for benchmark graphs with \( n = 1000 \) vertices

Figure 3.6: Histogram of vertex change scores (left axis) and probability density functions of the two components of a Gaussian mixture model fitted to the vertex change scores using the Expectation Maximization (EM) algorithm (right axis) for benchmark graphs with \( n = 1000 \) vertices and change parameter \( \delta = 0.1 \) (a), 0.5 (b), and 0.9 (c). The corresponding Bayesian information criterion (BIC) values are shown in the upper corner of each figure. Comparable results were obtained for other values of \( \delta \).

mean corresponds to the vertices changing community at time step \( t \). Therefore, the change parameter can be estimated as the mixing proportion of the component with the higher mean obtained after fitting a two-component Gaussian mixture model to the vertex change scores using the Expectation Maximization (EM) algorithm (see Figure 3.5). To avoid convergence to local maxima, we run the EM algorithm ten (10) times using random initial values and return the model that minimizes the Bayesian information criterion (BIC). The low negative BIC values obtained (see Figure 3.6) indicate that the models fit the data well.
Note that estimating these parameters for large-scale networks can be computationally expensive. Alternatively, the results obtained suggest that setting the value of both $\alpha$ and $\beta$ to 0.5 generally yields a good trade-off between efficiency and quality of the communities identified.

**Analysis of Attribute Similarity Gain Estimation**

Finally, we empirically analyzed the efficiency and accuracy of our proposed estimation of the gain in attribute similarity in Equation 3.8 with respect to the exact computation in Equation 3.7. We observe that our proposed estimation has a minimal impact on the modularity and the attribute similarity of the graph partitions obtained, while significantly reducing the running time of the algorithm (see Figure 3.7). These results indicate that our proposed estimation allows us to efficiently compute the gain in attribute similarity without negatively affecting the quality of the communities identified.

### 3.5.2 Real-World Networks

We also evaluated the performance of our proposed algorithm on large-scale dynamic attributed networks constructed from three real-world data sets.

![Modularity, Similarity and Running Time for $\mu = 0.1$, $\lambda = 0.1$, $\alpha = 0.5$, $\beta = 0.5$, $n = 1000$](image)

Figure 3.7: Average *modularity* and *attribute similarity* of the partitions obtained and average *running time* (in seconds) at each time step of the benchmark graphs using our proposed algorithm with weighting parameter $\alpha = 0.5$ and update parameter $\beta = 0.5$. Benchmark graphs were generated using the following parameters: number of vertices $n = 1000$, mixing parameter $\mu = 0.1$, noise parameter $\lambda = 0.1$, and change parameter $\delta = 0.1$. 

44
The DBLP\textsuperscript{1} data set provides publication records from 1991 to 2000. In the corresponding network, an edge between two vertices is present if the authors represented by those two vertices collaborated in a publication. Vertices have 19 categorical attributes representing each author’s areas of publication (e.g., artificial intelligence, bioinformatics, security).

The Yelp\textsuperscript{2} data set provides user reviews of a select set of businesses from 2004 to 2012. In the corresponding network, an edge between two vertices is present if the users represented by those two vertices reviewed the same business. Vertices have 38 categorical attributes representing the type of businesses reviewed by each user (e.g., restaurants, shops, services), as well as a numeric attribute corresponding to the average rating assigned by each user.

The TripAdvisor\textsuperscript{3} data set provides hotel reviews from 2002 to 2012. In the corresponding network, an edge between two vertices is present if the users represented by those two vertices reviewed the same hotel. Vertices have a numeric attribute corresponding to the average rating assigned by each user.

**Experimental Setup**

Dynamic attributed networks were constructed for each real-world data set using its first eight (8) years of data, such that time step $i - 1$ of each network corresponds to the data available up to the $i$-th year. The networks were preprocessed to remove all singleton vertices, as these would not be placed in any community. Further details of the networks are shown in Table 3.1.

We identified communities in these real-world networks using our proposed algorithm, as well as CODICIL [48] and I-Louvain [11], two state-of-the-art algorithms for community detection in

<table>
<thead>
<tr>
<th>Graph</th>
<th>Time Step</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$t = 0$</td>
<td>$t = 1$</td>
<td>$t = 2$</td>
<td>$t = 3$</td>
<td>$t = 4$</td>
<td>$t = 5$</td>
<td>$t = 6$</td>
<td>$t = 7$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBLP</td>
<td>Num Vertices</td>
<td>13,782</td>
<td>26,359</td>
<td>38,797</td>
<td>51,176</td>
<td>64,366</td>
<td>78,089</td>
<td>93,142</td>
<td>110,065</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Num Edges</td>
<td>33,528</td>
<td>71,636</td>
<td>114,950</td>
<td>159,154</td>
<td>213,806</td>
<td>273,188</td>
<td>341,330</td>
<td>426,548</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yelp</td>
<td>Num Vertices</td>
<td>7</td>
<td>380</td>
<td>2,488</td>
<td>8,683</td>
<td>20,132</td>
<td>36,895</td>
<td>61,338</td>
<td>97,039</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Num Edges</td>
<td>14</td>
<td>8,286</td>
<td>90,626</td>
<td>461,468</td>
<td>1,469,412</td>
<td>3,146,770</td>
<td>5,772,004</td>
<td>10,372,332</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TripAdvisor</td>
<td>Num Vertices</td>
<td>15</td>
<td>714</td>
<td>3,564</td>
<td>10,354</td>
<td>39,723</td>
<td>112,190</td>
<td>203,562</td>
<td>297,301</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Num Edges</td>
<td>26</td>
<td>5,116</td>
<td>54,418</td>
<td>292,390</td>
<td>1,485,726</td>
<td>7,866,020</td>
<td>18,708,064</td>
<td>28,288,858</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\textsuperscript{1}dblp.uni-trier.de/xml
\textsuperscript{2}www.yelp.com/dataset\_challenge
\textsuperscript{3}times.cs.uiuc.edu/~wang296/Data
attributed graphs. CODICIL and I-Louvain have been shown to outperform structure-only and attribute-only community detection methods, such as the Louvain method [7] and the k-means clustering algorithm, as well as other methods for community detection in attributed graphs, such as the Inc-Cluster algorithm [65]. Since CODICIL and I-Louvain were not designed for dynamic graphs, we ran them from scratch at each time step. Moreover, as neither of these algorithms can handle both categorical and numeric attributes, we used CODICIL for the networks with categorical attributes, and I-Louvain for the networks with numeric attributes.

All experiments on real-world networks were performed on an Intel machine running RHEL Server 6.7 consisting of two hex-core E5645 processors and 64GB DDR2 RAM. Our proposed algorithm was implemented in C++ and compiled with GCC 4.8.2 using the optimization flag -O3. For simplicity, we set the value of the weighting parameter $\alpha$ and the update parameter $\beta$ to 0.5 for all experiments, as suggested in Section 3.5.1. For the state-of-the-art algorithms, we used Python implementations provided by the authors with its default parameters. Note that the CODICIL implementation returns an augmented graph that combines structural and attribute information, after which a traditional community detection method must be applied. To this end, we used the original Louvain method. However, the Louvain method’s time and space requirements were not included in CODICIL’s time and space measurements.

The algorithms were compared in terms of the quality of the communities identified and the efficiency of the implementation. Due to the existence of multiple definitions of community, evaluating their quality requires the consideration of multiple metrics. To evaluate their structural properties, we computed the modularity and average density [23] of the graph partition, and to evaluate the homogeneity of their attribute information, we computed the average attribute similarity, as well as the average entropy [12] or standard deviation for the networks with categorical or numeric attributes, respectively. The results obtained are shown in Table 3.2.

**Discussion of Results**

With respect to the quality of the communities, our proposed algorithm obtained graph partitions with higher modularity than both state-of-the-art algorithms (an average improvement of 6% and 8% compared to CODICIL and I-Louvain, respectively). For the networks with categorical attributes, our proposed algorithm identified communities with higher average density and attribute similarity than CODICIL (an average improvement of 6% and 5%, respectively), but also higher average entropy. On the other hand, for the networks with numeric attributes, I-Louvain identified communities with higher density, higher attribute similarity, and lower average standard deviation than our proposed algorithm (an average improvement of 4%, 1%, and 2%, respectively). It is worth noting that many communities identified by the algorithms were 2-cliques. Local metrics, such as average density, are likely to be skewed towards these smaller
Table 3.2: Results obtained (number of communities, modularity, average density, attribute similarity, average entropy or average standard deviation, peak memory in MB, and running time in seconds) using our proposed algorithm and a state-of-the-art community detection algorithm (CODICIL [48] for networks with categorical attributes or I-Louvain [11] for networks with numeric attributes) on large-scale real-world networks. Asterisks (*) and daggers (†) indicate that the implementation of the algorithm did not run in the allotted time of 5 hours or with the allotted memory of 64GB, respectively. Best results are shown in bold and underlined.

<table>
<thead>
<tr>
<th>Graph</th>
<th>Proposed Algorithm</th>
<th>State-of-the-art: CODICIL/I-Louvain</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Num Mod Den Att Ent Peak Run</td>
<td>Num Mod Den Att Ent Peak Run</td>
</tr>
<tr>
<td></td>
<td>Sim/ Std Mem Time</td>
<td>Sim/ Std Mem Time</td>
</tr>
<tr>
<td>DBLP + Categorical Attributes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t = 0</td>
<td>4,006 1.00 0.91 0.95 0.22 6.08 0.16</td>
<td>4,023 1.00 0.91 0.95 0.21 22.37 518.38</td>
</tr>
<tr>
<td>t = 1</td>
<td>6,019 0.99 0.88 0.94 0.42 15.41 0.45</td>
<td>6,322 0.99 0.88 0.94 0.39 40.27 2,011.42</td>
</tr>
<tr>
<td>t = 2</td>
<td>7,075 0.98 0.86 0.93 0.68 28.69 0.96</td>
<td>8,026 0.98 0.88 0.93 0.55 60.82 4,209.35</td>
</tr>
<tr>
<td>t = 3</td>
<td>7,662 0.97 0.86 0.92 0.91 44.43 1.56</td>
<td>9,322 0.97 0.87 0.93 0.72 78.45 7,401.62</td>
</tr>
<tr>
<td>t = 4</td>
<td>8,132 0.95 0.86 0.92 1.12 63.34 3.23</td>
<td>10,449 0.95 0.88 0.93 0.87 100.88 10,397.90</td>
</tr>
<tr>
<td>t = 5</td>
<td>8,655 0.94 0.87 0.92 1.27 85.94 7.41</td>
<td>11,566 0.95 0.88 0.93 0.99 124.93 17,271.63</td>
</tr>
<tr>
<td>t = 6</td>
<td>9,152 0.93 0.87 0.92 1.41 113.00 12.69</td>
<td>* * * * * *</td>
</tr>
<tr>
<td>t = 7</td>
<td>9,921 0.92 0.88 0.92 1.55 143.69 25.29</td>
<td>* * * * * *</td>
</tr>
<tr>
<td>Yelp + Categorical Attributes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t = 0</td>
<td>2 0.49 0.83 0.60 3.59 1.52 0.00</td>
<td>2 0.49 0.83 0.60 3.59 5.04 0.14</td>
</tr>
<tr>
<td>t = 1</td>
<td>17 0.40 0.66 0.65 6.44 1.75 0.01</td>
<td>13 0.24 0.50 0.51 5.81 8.15 1.32</td>
</tr>
<tr>
<td>t = 2</td>
<td>126 0.58 0.79 0.74 5.44 3.10 0.38</td>
<td>51 0.48 0.61 0.63 5.56 29.36 55.57</td>
</tr>
<tr>
<td>t = 3</td>
<td>65 0.66 0.82 0.83 6.25 7.80 2.48</td>
<td>85 0.62 0.70 0.73 5.90 121.69 971.89</td>
</tr>
<tr>
<td>t = 4</td>
<td>129 0.69 0.91 0.91 6.09 18.99 59.84</td>
<td>166 0.65 0.79 0.83 5.77 374.77 8,033.96</td>
</tr>
<tr>
<td>t = 5</td>
<td>254 0.72 0.94 0.94 5.96 38.04 179.60</td>
<td>* * * * * *</td>
</tr>
<tr>
<td>t = 6</td>
<td>464 0.67 0.96 0.95 5.89 67.53 699.93</td>
<td>* * * * * *</td>
</tr>
<tr>
<td>t = 7</td>
<td>839 0.66 0.98 0.72 5.99 116.14 299.81</td>
<td>* * * * * *</td>
</tr>
<tr>
<td>Yelp + Numeric Attributes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t = 0</td>
<td>2 0.49 0.83 0.71 0.77 1.52 0.00</td>
<td>2 0.49 0.83 0.71 0.77 35.60 0.41</td>
</tr>
<tr>
<td>t = 1</td>
<td>15 0.40 0.65 0.72 0.74 1.76 0.01</td>
<td>9 0.39 0.63 0.72 0.73 60.76 1.60</td>
</tr>
<tr>
<td>t = 2</td>
<td>41 0.61 0.73 0.73 0.73 3.04 0.27</td>
<td>32 0.52 0.86 0.76 0.72 641.06 46.32</td>
</tr>
<tr>
<td>t = 3</td>
<td>63 0.68 0.80 0.74 0.88 7.80 2.31</td>
<td>61 0.50 0.73 0.77 0.82 7,890.45 398.87</td>
</tr>
<tr>
<td>t = 4</td>
<td>124 0.68 0.91 0.79 0.85 18.98 22.61</td>
<td>128 0.49 0.96 0.80 0.81 21,104.88 2,730.09</td>
</tr>
<tr>
<td>t = 5</td>
<td>225 0.70 0.95 0.81 0.82 38.07 122.61</td>
<td>† † † † † †</td>
</tr>
<tr>
<td>t = 6</td>
<td>399 0.71 0.95 0.83 0.70 67.80 512.50</td>
<td>† † † † † †</td>
</tr>
<tr>
<td>t = 7</td>
<td>735 0.70 0.97 0.76 0.99 115.97 901.80</td>
<td>† † † † † †</td>
</tr>
<tr>
<td>TripAdvisor + Numeric Attributes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t = 0</td>
<td>6 0.71 1.00 0.73 1.05 1.64 0.00</td>
<td>6 0.71 1.00 0.73 1.05 35.58 0.50</td>
</tr>
<tr>
<td>t = 1</td>
<td>144 0.94 0.95 0.75 0.86 1.97 0.01</td>
<td>144 0.94 0.95 0.75 0.86 80.74 3.98</td>
</tr>
<tr>
<td>t = 2</td>
<td>425 0.95 0.92 0.74 0.87 3.31 0.05</td>
<td>439 0.95 0.93 0.74 0.86 911.01 109.32</td>
</tr>
<tr>
<td>t = 3</td>
<td>618 0.95 0.90 0.73 0.88 8.28 0.25</td>
<td>696 0.92 0.94 0.74 0.87 9,451.34 778.46</td>
</tr>
<tr>
<td>t = 4</td>
<td>1,636 0.93 0.93 0.73 0.93 28.10 1.83</td>
<td>† † † † † †</td>
</tr>
<tr>
<td>t = 5</td>
<td>2,179 0.88 0.94 0.73 0.98 94.06 20.97</td>
<td>† † † † † †</td>
</tr>
<tr>
<td>t = 6</td>
<td>2,850 0.86 0.95 0.73 1.01 203.46 138.74</td>
<td>† † † † † †</td>
</tr>
<tr>
<td>t = 7</td>
<td>3,475 0.84 0.95 0.73 0.99 334.98 385.27</td>
<td>† † † † † †</td>
</tr>
</tbody>
</table>
communities, and global metrics, such as modularity, might provide a better estimate of the quality of the graph partition as a whole.

With respect to the efficiency of the algorithms, the CODICIL implementation required, on average, 2000 times more time and 6 times more peak memory than the proposed algorithm, while the I-Louvain implementation required, on average, 900 times more time and 400 times more peak memory. Moreover, the CODICIL implementation was not able to run on five (5) time steps in the allotted time of five (5) hours, while the I-Louvain implementation was not able to run on seven (7) time steps with the allotted memory of 64GB. Even though these measurements are implementation-dependent, they empirically confirm the observations regarding the high time and space complexity of these algorithms made in Section 3.2.

Overall, the results obtained show that our proposed algorithm is able to efficiently identify communities of high quality in large-scale real-world networks. While the state-of-the-art algorithms obtained some graph partitions of higher average density and attribute similarity and lower average entropy and standard deviation, their time and memory requirements were considerably higher (see Figure 3.8). This suggests that our proposed algorithm is able to achieve a better balance between efficiency and quality of the communities identified.

We further analyzed the results obtained using our proposed algorithm by tracking the evolution of the largest community in the DBLP network at $t = 0$. We chose the DBLP network because it only has categorical attributes, which can be visually inspected. The size of the community, the percentage of vertices from the previous time step that remain in the community,

![Running Time and Peak Memory of Algorithms on Real-World Networks](image)

Figure 3.8: Running time (left, in seconds) and peak memory (right, in MB) obtained at each time step of the real-world networks using our proposed algorithm, CODICIL [48], and I-Louvain [11]. Line styles and colors are used to denote the algorithms and marker types are used to denote the data sets. Missing results indicate that the implementation of the algorithm did not run in the allotted time of 5 hours or with the allotted memory of 64GB.
and the top three (3) attributes of the vertices in the community at each time step are shown in Table 3.3. We observe that, from \( t = 0 \) to \( t = 3 \), the community is expanding. At \( t = 4 \), the community contracts and its top two (2) attribute changes. Further changes are observed at \( t = 6 \) and \( t = 7 \). These changes in the attribute information can also be observed in the word clouds of attributes shown in Figure 3.9. While further analysis of the DBLP network is required to interpret these changes, the results obtained show that incorporating the attribute information of the vertices into the community tracking process allows us to describe the evolution of the communities more intuitively.

Table 3.3: Number of vertices (Num Vertices), percentage of vertices from previous time step (% \( t - 1 \) Vertices), and top 3 attributes per time step for the largest community in the DBLP network at \( t = 0 \), illustrating the evolution of this community from \( t = 0 \) to \( t = 7 \). Attributes in the DBLP network, such as Computer Science (CS), Math, Systems, Engineering (Eng), and Artificial Intelligence (AI), represent authors’ areas of publication.

<table>
<thead>
<tr>
<th>Graph</th>
<th>Time Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBLP</td>
<td>( t = 0 )</td>
</tr>
<tr>
<td>Num Vertices</td>
<td>185</td>
</tr>
<tr>
<td>% ( t - 1 ) Vertices</td>
<td>NA</td>
</tr>
<tr>
<td>Top 1 Attribute</td>
<td>CS</td>
</tr>
<tr>
<td>Top 2 Attribute</td>
<td>Math</td>
</tr>
<tr>
<td>Top 3 Attribute</td>
<td>Systems</td>
</tr>
</tbody>
</table>

Figure 3.9: Word clouds of attributes per time step for the largest community in the DBLP network at \( t = 0 \), illustrating the evolution of this community from \( t = 0 \) to \( t = 7 \). Attributes in the DBLP network, such as Math, Systems, Engineering, and Artificial Intelligence (AI), represent authors’ areas of publication. The Computer Science attribute was omitted from the word clouds as it is the top 1 attribute in all communities.
3.6 Conclusion

In this chapter, we proposed an efficient multi-objective algorithm for community detection in dynamic attributed graphs. While several methods exist to detect communities in attributed graphs [11, 12, 16, 48, 64, 65] and in dynamic graphs [3, 13], this is, to the best of our knowledge, the first methodology to take into account both the attribute information of the vertices and the dynamic nature of the graph, along with the structural properties.

We evaluated our proposed algorithm on a variety of synthetically generated benchmark dynamic attributed graphs, as well as on large-scale real-world networks. The results obtained show that our proposed algorithm is able to identify graph partitions of high modularity and high attribute similarity more efficiently than state-of-the-art methods for community detection [11, 48] in terms of both space and time. Moreover, unlike these state-of-the-art methods, our proposed algorithm is able to handle multiple attribute types, which is essential given the heterogeneous nature of many real-world networks.
Chapter 4

Predicting Future Communities in Dynamic Graphs

4.1 Introduction

As explained in Chapter 3, complex networks are often dynamic in nature; that is, they change over time. For example, in social networks, users may join or leave the network (i.e., vertex insertion or deletion) and relationships between users may be formed or dissolved (i.e., edge insertion or deletion). In Chapter 3, we incorporated these changes to update communities at each time step more efficiently. However, when detecting communities in dynamic graphs, additional research questions arise.

While most of the research on community detection has focused on static graphs, several methods have been recently proposed not only to detect communities in dynamic graphs (e.g., [13, 3]), but also to track these communities and characterize their evolution over time [2, 21, 43] as mentioned in Chapter 3. Furthermore, the problem of link prediction in dynamic graphs—that is, predicting the insertion of edges in future time steps—has been well studied and numerous link prediction methods have been proposed [36, 62].

On the other hand, little attention has been given to the prediction of communities in future time steps, an important task that, if successfully achieved, could improve our understanding of the networks’ dynamics. For this reason, we capitalize on existing research on link prediction to address the unsolved problem of community prediction. To this end, we propose a methodology to predict future communities in dynamic graphs that incorporates edge likelihoods obtained using link prediction methods into the community detection process.

The main contributions of this chapter are as follows. First, we define the problem of community prediction in dynamic graphs (Section 4.3). Second, we formulate and empirically evaluate a general hypothesis for the prediction of future communities that states that if we can pre-
dict the future edges of a dynamic graph, then we can predict its future community structure (Section 4.4). And third, we propose a community prediction methodology that aims to predict the future community structure of a dynamic graph more efficiently and accurately than state-of-the-art methodologies [40] by using link prediction methods (Section 4.5).

The rest of this chapter is organized in the following way. In Section 4.2, we present an overview of existing methodologies for community prediction and highlight their limitations. In Section 4.3, we formally define the problem of community prediction in dynamic graphs. In Section 4.4, we formulate and evaluate our hypothesis for the prediction of future communities in dynamic graphs. In Section 4.5, we describe our proposed community prediction methodology. In Section 4.6, we discuss the results obtained using our proposed community prediction methodology on synthetically generated dynamic graphs and on real-world networks. Finally, we present our conclusions in Section 4.7.

4.2 Related Work

Few approaches have been proposed to predict future communities in dynamic graphs [40, 19, 58]. Most of these approaches involve building regression or classification models to predict the future evolution of the communities. For example, Goldberg et al. build linear regression models to estimate the lifespan of a community based on its properties (e.g., size, density) at previous time steps [19]. Similarly, Takaffoli, Rabbany, and Zaïane build classification models to predict the survival or death of a community based on properties of its central vertices (e.g., degree centrality, closeness centrality), properties of the community itself (e.g., size, density), and the rate of change of these properties over time [58]. For communities predicted to survive, further models are built to predict whether they will merge, split, expand, or contract. While these methodologies allow us to predict some key events in the evolution of the communities, none are able to estimate the future community structure of the graph.

On the other hand, Ngonmang and Viennet build classification models to predict, for every pair of vertices in the graph, the existence of an edge between them in a future time step. These models are built using features such as the number of common neighbors, the number of common community members, and a Boolean variable indicating whether there is currently an edge between them. A community detection method is then applied on this predicted graph to estimate its future community structure [40]. While this approach allows us to predict future communities, the quality of these communities is highly dependent on the accuracy of the classification models. Moreover, these existing methodologies require computing multiple features and building regression or classification models, which can be computationally expensive and may limit their practical applicability to large-scale real-world networks.
4.3 Problem Statement

Let $G = (V,E)$ be an unweighted, undirected graph, where $V$ is the set of vertices and $E$ is the set of edges. We define a dynamic graph as an ordered set $G = \{G_t\}_{t=1}^{T}$, where $T$ is the number of time steps and $G_t$ is the graph at time step $t$. Note that we assume that corresponding vertices are labeled consistently across different time steps.

The problem of community prediction in dynamic graphs is defined as follows. Given a graph $G_t \in G$, partition $G_t$ into a set of disjoint communities $C_{t+1} = \{c_1, c_2, ..., c_{K_{t+1}}\}$, where $\bigcup_{i=1}^{K_{t+1}} c_i = V_t$ and $c_i \cap c_j = \emptyset$ for all $c_i, c_j \in C_{t+1}$ with $i \neq j$, such that vertices in the same community are more likely to be densely connected between each other in an unseen future time step $t + 1$ than those in different communities.

4.4 Hypothesis

The central hypothesis of this chapter states that if we can predict future edges in a dynamic graph, then we can use this information to predict its future community structure. We evaluated this hypothesis empirically on synthetically generated dynamic graphs.

4.4.1 Data

At the initial time step $t$, we generated an unweighted, undirected graph $G_t = (V,E)$ with $|V| = 1000$ vertices partitioned into communities, such that the degree of the vertices and the size of the communities follow power law distributions and the fraction of edges of each vertex pointing outside its community is given by a mixing parameter $\mu \in [0,1]$. At every subsequent time step $t + i$, we inserted $\delta \cdot |E|$ new edges to the graph, where $\delta \in [0,1]$ is the percentage of new edges inserted, such that the probability of inserting a new edge between vertices $v$ and $w$ is proportional to the number of common neighbors between them. Moreover, we only inserted new edges between vertices in different communities because inserting new edges within the communities will not change the community structure of the graph.

Note that, in this dissertation, we assume that only new edges are inserted between time steps. Extending our hypothesis to include vertex insertion, as well as edge and vertex deletion, is the subject of future work.

4.4.2 Evaluation

To evaluate the accuracy of the community prediction, we computed the Adjusted Rand Index (ARI) [26]—a widely used metric to measure the similarity between two partitions—between the communities predicted at time step $t$ for time step $t + 1$ and the communities identified
at time step $t + 1$ using a community detection method. Given two partitions of a graph, $X = \{X_1, X_2, \ldots, X_r\}$ and $Y = \{Y_1, Y_2, \ldots, Y_s\}$, the ARI between them is given by

$$ARI(X, Y) = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left[ \sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2} \right]/\binom{n}{2}}{\frac{1}{2} \left[ \sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2} \right] - \left[ \sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2} \right]/\binom{n}{2}}$$

(4.1)

where $n_{ij} = |X_i \cap Y_j|$, $a_i = \sum_j n_{ij}$, $b_j = \sum_i n_{ij}$, and $n$ is the number of vertices in the graph.

As a baseline, we predicted the communities for time step $t + 1$ to be those identified at time step $t$ using a community detection method. As shown in Figure 4.1, the accuracy of the baseline decreases as the mixing parameter of the graph, $\mu$, and the percentage of new edges inserted, $\delta$, increases. This is expected, since graphs with higher mixing parameters have less well-defined community structures, which are more likely to change when new edges are inserted.

We then predicted the communities for time step $t + 1$ assuming that a certain percentage of future edges were predicted correctly or incorrectly. The results obtained on $N = 100$ synthetic graphs are shown in Figure 4.2. When future edges were predicted correctly, we observe a statistically significant ($p < 0.01$) increase in ARI with respect to the baseline that varies from 2% to 12%, when 20% to 90% of the edges were predicted correctly. This suggests that, as stated in our hypothesis, we can predict the future community structure of a graph using information from its predicted future edges. At the same time, no statistically significant ($p < 0.01$) decrease in ARI with respect to the baseline is observed when future edges were predicted incorrectly, which suggests that these results may be robust to errors in the prediction of future edges.

![Baseline Adjusted Rand Index](image)

Figure 4.1: Baseline Adjusted Rand Index (ARI) obtained on $N = 100$ synthetic dynamic graphs with $\delta = 0.1$ and various values of $\mu$ (left) and with $\mu = 0.5$ and various values of $\delta$ (right), where $\mu$ is the mixing parameter of the graph and $\delta$ is the percentage of new edges inserted. Shaded regions indicate plus or minus one standard deviation centered around the mean.
Figure 4.2: Predicted and baseline Adjusted Rand Index (ARI) obtained on $N = 100$ synthetic dynamic graphs with $\mu = 0.5$ and $\delta = 0.1$, where $\mu$ is the mixing parameter of the graph and $\delta$ is the percentage of new edges inserted, for various percentages of edges predicted correctly (left) and incorrectly (right). Shaded regions indicate plus or minus one standard deviation centered around the mean.

We also evaluated the state-of-the-art community prediction methodology by Ngonmang and Viennet [40] described in Section 4.2. To this end, we trained classification models to predict communities at time step $t$ for time step $t+1$ and then tested these models by predicting communities at time step $t+1$ for time step $t+2$. As suggested by the authors, we used support vector machines (SVM) with radial basis function (RBF) kernel as the classification method and applied random undersampling to balance the distribution of the classes.

The results obtained on $N = 10$ synthetic graphs are shown in Figure 4.3. We observe that the state-of-the-art methodology was not able to predict any future edges correctly and, as a result, the ARI obtained was equal to that of the baseline. This indicates that, as suggested in Section 4.2, the accuracy of the state-of-the-art methodology is highly dependent on the accuracy of the classification models and is not robust to errors in the prediction of future edges.

4.5 Community Prediction in Dynamic Graphs

In this dissertation, we propose an alternative approach to community prediction that consists of applying link prediction methods to estimate edge likelihoods and then using this information to predict the future community structure of the graph. The reasoning behind this approach is that two currently non-adjacent vertices with a high edge likelihood may become connected in a future time step and thus may eventually belong to the same community.
The use of link prediction methods for community prediction has been previously suggested [40]; however, the value of this approach has not been fully investigated. We posit that this approach has multiple advantages over existing community prediction methodologies. First, it allows us to predict future communities more efficiently, as it reduce[s] the computational complexity of the problem by minimizing the number of features that need to be computed and the number of classification models that need to be built. Second, unlike the state-of-the-art methodology [40], in which the prediction of future edges and the prediction of future communities are performed separately, the proposed methodology directly incorporates edge likelihoods into the community detection process. This allows us to predict future communities more accurately, as the results are expected to be more robust to errors in the prediction of future edges.

The proposed community prediction methodology (see Algorithm 4.1) proceeds as follows.

For every pair of vertices \( v, w \in V \), we compute the likelihood of the existence of an edge between them, \( \mathcal{L}_{vw} \), using a link prediction method. Note that this link prediction method must return a score quantifying the likelihood of the existence of the edge as opposed to simply indicating whether or not this edge may be present in a future time step.

We use edge likelihood \( \mathcal{L}_{vw} \) to assign a weight between vertices \( v \) and \( w \), \( W_{vw} \), given by

\[
W_{vw} = \begin{cases} 
A_{vw} + \mathcal{L}_{vw} & \text{if } \mathcal{L}_{vw} \geq \theta \\
A_{vw} & \text{if } \mathcal{L}_{vw} < \theta 
\end{cases}
\]  

\[ (4.2) \]
where $A$ is the adjacency matrix of the graph (i.e., $A_{vw} = 1$ if vertices $v$ and $w$ are adjacent and $A_{vw} = 0$ otherwise) and $\theta$ is a threshold parameter. Here, we set the threshold parameter $\theta$ to be the average edge likelihood over all pairs of currently adjacent vertices in the graph; that is,

$$\theta = \frac{1}{m} \sum_{v,w \in V} A_{vw} \cdot L_{vw}$$

(4.3)

where $m = \frac{1}{2} \Sigma_{vw} A_{vw}$ is the number of edges in the graph.

Note that this weight definition guarantees that pairs of currently adjacent vertices have a high weight, while pairs of vertices with a low edge likelihood have a weight of zero. This is intended to minimize the impact on our results of errors in the prediction of future edges.

Next, we construct a weighted graph with $W$ as the adjacency matrix; that is, two vertices $v$ and $w$ are adjacent if $W_{vw} > 0$ and $W_{vw}$ is the weight of the edge between them.

Finally, we apply a community detection method on the weighted graph. The communities identified are returned as the predicted future communities of the original graph.

---

**Algorithm 4.1: Proposed community prediction methodology**

Input : graph at current time step $t$, $G = (V,E)$

Output: vector of predicted community memberships for future time step $t + 1$, $C_{t+1}$

1. Compute matrix of edge likelihoods $L$ using link prediction method
2. Compute weight matrix $W$ given by Equation 4.2
3. Construct weighted graph $G'$ with adjacency matrix $W$
4. Identify communities $C_{t+1}$ in weighted graph $G'$ using community detection method
5. return $C_{t+1}$

---

### 4.5.1 Computing Edge Likelihoods

To compute the matrix of edge likelihoods (step 1 of Algorithm 4.1), we consider a variety of link prediction methods, including local similarity-based methods [36, 62], global similarity-based methods [36, 62], and a state-of-the-art method based on matrix perturbations [35]. As previously discussed, we only consider link prediction methods that return edge likelihoods as opposed to predicted future edges. Thus, classification-based methods (e.g., [33]) were not considered.
Local Similarity-based Methods

Local similarity-based link prediction methods, such as Common Neighbors (CN), Jaccard’s Coefficient, Adamic-Adar (AA) and Resource Allocation (RA), quantify the likelihood of the existence of an edge between two vertices based on the number of common neighbors between them, with AA and RA assigning more weight to the less-connected common neighbors [36, 62]. The CN, Jaccard, AA, and RA similarities between two vertices \( v \) and \( w \), \( S_{vw}^{CN} \), \( S_{vw}^{Jaccard} \), \( S_{vw}^{AA} \), and \( S_{vw}^{RA} \), respectively, are given by

\[
S_{vw}^{CN} = |\Gamma(v) \cap \Gamma(w)| \tag{4.4}
\]

\[
S_{vw}^{Jaccard} = \frac{|\Gamma(v) \cap \Gamma(w)|}{|\Gamma(v) \cup \Gamma(w)|} \tag{4.5}
\]

\[
S_{vw}^{AA} = \sum_{z \in \Gamma(v) \cap \Gamma(w)} \frac{1}{\log k_z} \tag{4.6}
\]

\[
S_{vw}^{RA} = \sum_{z \in \Gamma(v) \cap \Gamma(w)} \frac{1}{k_z} \tag{4.7}
\]

where \( \Gamma(v) \) is the set of neighbors of vertex \( v \) and \( k_v \) is the degree of vertex \( v \). The complexity of these local similarity-based link prediction methods is \( O(|V|^2 \cdot \Gamma_{max}) \), where \( |V| \) is the number of vertices in the graph and \( \Gamma_{max} \) is the maximum vertex degree, which is generally \( << |V| \).

Global Similarity-based Methods

Global similarity-based link prediction methods, such as the Katz index, quantify the likelihood of the existence of an edge between two vertices based on the number of paths of all lengths between them [36, 62]. The Katz similarity between two vertices \( v \) and \( w \), \( S_{vw}^{Katz} \), is given by

\[
S_{vw}^{Katz} = \Lambda A_{vw} + \Lambda^2 (A^2)_{vw} + \Lambda^3 (A^3)_{vw} + \ldots \tag{4.8}
\]

where \( A \) is the adjacency matrix of the graph, \( 0 < \Lambda < 1/\lambda_{max} \) is a damping factor that controls the weights assigned to each path length, and \( \lambda_{max} \) is the largest eigenvalue of \( A \). The Katz similarity matrix, \( S^{Katz} \), can then be computed as

\[
S^{Katz} = (I - \Lambda A)^{-1} - I \tag{4.9}
\]

where \( I \) is the identity matrix. The computation of the Katz similarity matrix is dominated by the matrix inversion operation. Thus, its complexity is \( O(|V|^{2.807}) \) using the Strassen algorithm, where \( |V| \) is the number of vertices in the graph.
Structural Perturbation Method

The structural perturbation method (SPM) [35] is a state-of-the-art link prediction method that has been shown to outperform CN, AA, RA and Katz in experimental evaluations. The SPM computes edge likelihoods by randomly perturbing a small subset of the edges of the graph and then estimating the adjacency matrix of the perturbed graph via eigenvalue decomposition. The computation of the SPM is dominated by the eigenvalue decomposition operation. Thus, its complexity is $O(|V|^3 \cdot N)$, where $|V|$ is the number of vertices in the graph and $N$ is the number of random runs performed.

The SPM algorithm [35] is shown in Algorithm 4.2.

**Algorithm 4.2: Structural perturbation method**

<table>
<thead>
<tr>
<th>\textbf{Input}</th>
<th>\textbf{Output}</th>
</tr>
</thead>
<tbody>
<tr>
<td>graph, $G = (V, E)$; number of random runs, $N$</td>
<td>matrix of edge likelihoods, $L$</td>
</tr>
</tbody>
</table>

1. for $i \leftarrow 1$ to $N$ do
2. Randomly select a subset of edges $\Delta E \subset E$ of size $p^H \cdot |E|$ with $p^H = 0.1$
3. Let $\Delta A$ be the adjacency matrix of graph $\Delta G = (V, \Delta E)$
4. Let $A^R$ be the adjacency matrix of graph $G^R = (V, E - \Delta E)$
5. Compute the eigenvalues, $\lambda_k$, and the corresponding eigenvectors, $v_k$, of $A^R$
6. $\Delta \lambda_k \leftarrow \frac{v_k^T \Delta A v_k}{v_k^T v_k}$
7. $L_i \leftarrow \sum_{|V|} (\lambda_k + \Delta \lambda_k) v_k v_k^T$
8. return $L \leftarrow \frac{1}{N} \sum_{i=1}^{N} L_i$

4.5.2 Detecting Communities

To identify communities in the weighted graph (step 4 of Algorithm 4.1), we employ a community detection method that partitions the graph into communities by maximizing a function of the structural quality of the partition. As in Chapter 2 and Chapter 3, we use the well-known modularity function for this purpose [39].

For a weighted graph $G$ partitioned into communities, the modularity of the partition of $G$ [38] is given by

$$Q(C) = \frac{1}{2m} \sum_{v,w \in V} \left[ W_{vw} - \frac{k_v k_w}{2m} \right] \delta(c_v, c_w) \quad (4.10)$$

where $W$ is the weighted adjacency matrix of the graph, $m = \frac{1}{2} \sum_{v,w} W_{vw}$, $k_v = \sum_w W_{vw}$, $c_v$ is the community of vertex $v$, and $\delta(c_v, c_w)$ is the Kronecker delta function (i.e., $\delta(c_v, c_w) = 1$ if
As discussed in Chapter 2 and Chapter 3, maximizing modularity is a computationally hard problem, but several heuristic algorithms have been proposed to efficiently identify graph partitions with high modularity [18]. Here, we again use the Louvain method introduced in Chapter 2 [7], a well-known single-objective algorithm for modularity maximization. This algorithm partitions a graph into communities by iteratively and sequentially assigning each vertex \( v \) to the community \( c_i \) that yields the highest positive gain in modularity, \( \Delta Q(v, c_i) \), until no further improvement can be achieved.

The Louvain method [7] is shown in Algorithm 4.3.

---

**Algorithm 4.3: Louvain method for community detection**

**Input**: graph, \( G = (V, E) \)

**Output**: vector of community memberships, \( C \)

1. do
   /* Phase one: */
   2. for each \( v \in V \) do
      3. \( C[v] \leftarrow v \)
   4. do
      5. \( \text{num}\_\text{moves} \leftarrow 0 \)
      6. for each \( v \in V \) do
         7. Compute \( \Delta Q(v, c_i) \) for each community \( c_i \in C \) connected to \( v \)
         8. if \( \exists c_i \in C \) such that \( \Delta Q(v, c_i) > 0 \) then
            9. \( C[v] \leftarrow \arg \max_{c_i \in C} \Delta Q(v, c_i) \)
            10. \( \text{num}\_\text{moves} \leftarrow \text{num}\_\text{moves} + 1 \)
      while \( \text{num}\_\text{moves} > 0 \)
   /* Phase two: */
   12. Build a new graph \( G = (V, E) \) by aggregating each \( c_i \in C \) into one vertex
   while at least one vertex is moved during phase one
14. return \( C \)

---

4.6 Experimental Evaluation

In this section, we present the results obtained using our proposed community prediction methodology on a variety of synthetically generated dynamic graphs (Section 4.6.1), as well as on dynamic graphs constructed from real-world data sets (Section 4.6.2).
4.6.1 Synthetic Graphs

Data

We evaluated the performance of the proposed community prediction methodology on synthetic dynamic graphs generated as described in Section 4.4.1.

Evaluation

At the initial time step $t$ of the synthetic graphs, we predicted the communities for time step $t + 1$ using the proposed community prediction methodology. As a baseline, we predicted the communities for time step $t + 1$ to be those identified at time step $t$. We estimated edge likelihoods using six link prediction methods (i.e., CN, Jaccard’s Coefficient, AA, RA, Katz and SPM), identified communities using the Louvain method [7], and evaluated the accuracy of the community prediction by computing the ARI between the communities predicted at time step $t$ and the communities identified at time step $t + 1$.

The results obtained are shown in Figure 4.4. We observe that local similarity-based link prediction methods, such as CN, Jaccard’s Coefficient, AA, and RA, consistently yielded better results than link prediction methods with higher computational complexity, such as Katz and SPM. On synthetic graphs with $\mu = 0.5$, these local similarity-based link prediction methods yielded a statistically significant ($p < 0.01$) increase in ARI with respect to the baseline of 6%. This falls within the expected increase in ARI resulting from using information from predicted future edges to predict the future community structure of the graph, as determined in Section 4.4.2. A statistically significant ($p < 0.01$) increase in ARI was also observed on synthetic graphs with $\mu = 0.9$.

We also computed the accuracy of the link prediction methods, which is given by the percentage of future edges predicted correctly. We consider an edge to be predicted correctly if the edge likelihood assigned to it is greater than or equal to the threshold parameter in Equation 4.3. The results obtained are shown in Figure 4.5. We observe that the link prediction accuracy is generally low, even in cases where we obtained a statistically significant increase in ARI with respect to the baseline (e.g., on synthetic graphs with $\mu = 0.5$). This suggests that the proposed community prediction methodology is more robust to errors in the prediction of future edges than the state-of-the-art methodology [40], as evidenced by the results for the latter shown in Figure 4.3.

This robustness to errors can be at least partly attributed to the use of a weighted adjacency matrix of edge likelihoods instead of a binary adjacency matrix of predicted future edges. In fact, if we modify the proposed community prediction methodology to use predicted future edges instead of edge likelihoods, we no longer observe an increase in ARI with respect to the baseline (see Figure 4.6).
Figure 4.4: Adjusted Rand Index (ARI) obtained using the baseline and the proposed community prediction methodology with multiple link prediction methods (Common Neighbors, Jaccard’s Coefficient, Adamic-Adar, Resource Allocation, Katz, and Structural Perturbation Method) on \( N = 100 \) synthetic dynamic graphs with \( \delta = 0.1 \) and various values of \( \mu \), where \( \mu \) is the mixing parameter of the graph and \( \delta \) is the percentage of new edges inserted. Red boxes indicate a statistically significant \((p < 0.01)\) improvement with respect to the baseline. Comparable results were obtained for other values of \( \delta \).

Finally, it is worth noting that the accuracy of the link prediction increases as the mixing parameter increases. This can be explained by the structural properties of the synthetic graphs (see Figure 4.7). In graphs with low mixing parameter, most edges between vertices with common neighbors are already present at the initial time step \( t \). Thus, most of the new edges inserted at time step \( t + 1 \) are between vertices with few or no common neighbors, which are more difficult to predict correctly using existing link prediction methods. The opposite situation occurs in graphs with high mixing parameter.
Figure 4.5: Percentage of edges predicted correctly obtained using multiple link prediction methods (Common Neighbors, Jaccard’s Coefficient, Adamic-Adar, Resource Allocation, Katz, and Structural Perturbation Method) on $N = 100$ synthetic dynamic graphs with $\delta = 0.1$ and various values of $\mu$, where $\mu$ is the mixing parameter of the graph and $\delta$ is the percentage of new edges inserted. Comparable results were obtained for other values of $\delta$. 
Figure 4.6: Adjusted Rand Index (ARI) obtained using the baseline and a modified version of the proposed community prediction methodology that uses predicted future edges instead of edge likelihoods on $N = 100$ synthetic dynamic graphs with $\mu = 0.5$ and $\delta = 0.1$, where $\mu$ is the mixing parameter of the graph and $\delta$ is the percentage of new edges inserted. Red boxes indicate a statistically significant ($p < 0.01$) improvement with respect to the baseline. Comparable results were obtained for other values of $\mu$.

Figure 4.7: Median number of common neighbors between vertices incident to edges present at the initial time step $t$ (left) and incident to edges inserted at the following time step $t + 1$ (right) on $N = 100$ synthetic dynamic graphs with $\delta = 0.1$ and various values of $\mu$, where $\mu$ is the mixing parameter of the graph and $\delta$ is the percentage of new edges inserted. Shaded regions indicate plus or minus one standard deviation centered around the mean.

4.6.2 Real-World Networks

Data

We also evaluated the performance of the proposed community prediction methodology on dynamic graphs constructed from three real-world data sets: DBLP\textsuperscript{1}, TripAdvisor\textsuperscript{2}, and Yelp\textsuperscript{3}.

\footnote{1dblp.uni-trier.de/xml} \footnote{2times.cs.uiuc.edu/~wang296/Data} \footnote{3www.yelp.com/dataset_challenge}
In the DBLP network, an edge between two vertices is present if the authors represented by the vertices collaborated on a publication. In the TripAdvisor and Yelp networks, an edge between two vertices is present if the users represented by the vertices reviewed the same business.

The structural properties of the real-world networks are shown in Table 4.1. Each network contains six time steps and each time step corresponds to a snapshot of the data up to a given year. As we assume that no new vertices are inserted between time steps, only those vertices present in the initial snapshot of the data \((t = 0)\) were considered for the experimental evaluation. It is worth noting that the real-world networks exhibit diverse characteristics, from low mixing parameter and percentage of new edges inserted in the case of the DBLP network to high mixing parameter and percentage of new edges inserted in the case of the Yelp network.

**Evaluation**

At each time step \(t\) of the real-world networks, we predicted the communities for time step \(t + 1\). As with the synthetic graphs, we identified communities using the Louvain method \([7]\) and evaluated the accuracy of the community prediction by computing the ARI between the communities predicted at time step \(t\) and the communities identified at time step \(t + 1\).

Table 4.1: Number of vertices, number of edges, mixing parameter \((\mu)\), and percentage of new edges \((\delta)\) per time step of dynamic graphs constructed from real-world data sets

<table>
<thead>
<tr>
<th>Network</th>
<th>Number of Vertices</th>
<th>Time Step</th>
<th>Number of Edges</th>
<th>(\mu)</th>
<th>(\delta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBLP</td>
<td>16,991</td>
<td>(t = 0)</td>
<td>16,764</td>
<td>0.0</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 1)</td>
<td>17,971</td>
<td>0.0</td>
<td>7.20%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 2)</td>
<td>19,213</td>
<td>0.0</td>
<td>7.41%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 3)</td>
<td>20,231</td>
<td>0.0</td>
<td>6.07%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 4)</td>
<td>21,249</td>
<td>0.0</td>
<td>6.07%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 5)</td>
<td>22,016</td>
<td>0.0</td>
<td>4.58%</td>
</tr>
<tr>
<td>TripAdvisor</td>
<td>10,354</td>
<td>(t = 0)</td>
<td>146,195</td>
<td>0.0</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 1)</td>
<td>228,139</td>
<td>0.1</td>
<td>56.05%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 2)</td>
<td>299,831</td>
<td>0.1</td>
<td>49.04%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 3)</td>
<td>357,405</td>
<td>0.3</td>
<td>39.38%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 4)</td>
<td>388,455</td>
<td>0.3</td>
<td>21.24%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 5)</td>
<td>426,244</td>
<td>0.3</td>
<td>25.85%</td>
</tr>
<tr>
<td>Yelp</td>
<td>8,683</td>
<td>(t = 0)</td>
<td>230,734</td>
<td>0.1</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 1)</td>
<td>476,583</td>
<td>0.6</td>
<td>106.55%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 2)</td>
<td>722,836</td>
<td>0.6</td>
<td>106.73%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 3)</td>
<td>969,422</td>
<td>0.6</td>
<td>106.87%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 4)</td>
<td>1,184,258</td>
<td>0.7</td>
<td>93.11%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(t = 5)</td>
<td>1,408,565</td>
<td>0.6</td>
<td>97.21%</td>
</tr>
</tbody>
</table>

65
First, we predicted communities on the real-world networks assuming that a certain percentage of future edges were predicted correctly. The results obtained are shown in Figure 4.8. We observe a statistically significant \( p < 0.01 \) increase in ARI with respect to the baseline when future edges were predicted correctly. This increase varies from 23% to 69% for the DBLP network and from 65% to 141% for the TripAdvisor network, when 20% to 80% of the edges were predicted correctly. For the Yelp network, this increase was not consistent across all time steps, possibly due to the particular characteristics of this network (i.e., high mixing parameter and percentage of new edges inserted). However, we still observe an average increase in ARI that varies from 8% to 61%, when 20% to 80% of the edges were predicted correctly.

Next, we predicted communities on the real-world networks using the proposed community prediction methodology. Only local similarity-based link prediction methods were considered to estimate the edge likelihoods as these methods have lower computational complexity and were shown to consistently yield better results on the synthetic graphs. The results obtained
are shown in Figure 4.9. Jaccard’s Coefficient and RA generally yielded better results than CN and AA. However, in all cases, the ARI obtained using the proposed community prediction methodology was comparable to or lower than the baseline, which can be explained by the low accuracy of the link prediction methods on the real-world networks (i.e., less than 2% of the edges were predicted correctly).

Figure 4.9: Adjusted Rand Index (ARI, left) and percentage of edges predicted correctly (right) obtained using the baseline and the proposed community prediction methodology with local similarity-based link prediction methods (Common Neighbors, Jaccard’s Coefficient, Adamic-Adar, and Resource Allocation) on dynamic graphs constructed from real-world data sets: DBLP (upper row), TripAdvisor (middle row), and Yelp (lower row)
These results are consistent with recent studies that have found the performance of existing link prediction methods on real-world networks to be poor, with a link prediction accuracy “in the single digits in percentages” [34]. This indicates that link prediction is still an unsolved problem. Existing link prediction methods generally assume that missing edges between “similar” vertices are more likely to be inserted in future time steps. However, the low accuracy of these methods on real-world network exposes the limitations of this assumption. Thus, a better understanding of the underlying dynamics modulating the insertion of edges on real-world networks is needed. This would allow us to develop more accurate link prediction methods, which could lead to improvements in the prediction of future communities in dynamic graphs.

Finally, it is worth noting that the ARI obtained using the state-of-the-art community prediction methodology [40] on the real-world networks is also not greater than the baseline (see Figure 4.10). However, the link prediction accuracy is considerably higher (i.e., between 6% and 13% for the DBLP network and between 12% and 35% for the TripAdvisor network). This

![Graphs showing ARI and percentage of edges predicted correctly for DBLP and TripAdvisor networks](image-url)

Figure 4.10: Adjusted Rand Index (ARI, left) and percentage of edges predicted correctly (right) obtained using the baseline and the state-of-the-art (SOTA) community prediction methodology [40] on dynamic graphs constructed from real-world data sets: DBLP (upper row) and TripAdvisor (lower row). Due to its high computational complexity, the SOTA methodology did not run on the Yelp network in the time allotted for the experimental evaluation.
again suggests that the proposed community prediction methodology is more robust to errors in the prediction of future edges than the state-of-the-art methodology [40]. Moreover, due to the high computational complexity associated with computing multiple features and building classification models, the state-of-the-art methodology [40] does not scale well to large graphs, such as the Yelp network, which contains over one million edges.

4.7 Conclusion

In this chapter, we proposed a methodology for the prediction of future communities in dynamic graphs using link prediction methods. Unlike existing community prediction methodologies [40], we directly incorporate edge likelihoods obtained using these link prediction methods into the community detection process, which allows us to predict future communities more efficiently and accurately.

We evaluated our proposed community prediction methodology on both synthetic and real-world networks. The results obtained on the synthetic graphs show statistically significant improvements with respect to the baseline in the prediction of future communities using local similarity-based link prediction methods. On the other hand, no statistically significant improvements were observed on the real-world networks due to the low accuracy of the link prediction methods on these networks. These results indicate that, even though numerous link prediction methods have been proposed, this is still an unsolved problem. Thus, more accurate link prediction methods are needed to further improve the accuracy of our proposed community prediction methodology.
Chapter 5

Conclusion

5.1 Conclusion

As reiterated throughout this dissertation, complex systems are commonly represented as networks, where vertices represent objects in the system and edges represent relationships or interactions between these objects. Due to the ubiquity of complex systems, interest in the field of network analysis has grown considerably in the past years with numerous graph mining algorithms being proposed to identify patterns in complex networks. Most of these algorithms identify patterns that satisfy a single criterion, which is usually related to the structure of the graph. However, identifying “interesting” patterns in complex networks often requires looking beyond the structure of the graph by incorporating additional sources of information and addressing the dynamic nature of the network.

In this dissertation, we postulated that the emphasis of graph mining research in general, and community detection research in particular, should shift from traditional single-objective algorithms to multi-objective algorithms that optimize multiple objective functions simultaneously while also taking into account the dynamic nature of the network. To illustrate the value of this multi-objective approach, we tackled three distinct research objectives related to community detection in complex networks. In Chapter 2, we proposed a multi-objective algorithm to detect communities associated with a response variable of interest and applied it to identify communities in climate networks associated with seasonal rainfall variability. In Chapter 3, we proposed a multi-objective algorithm to detect communities in dynamic attributed graphs. And in Chapter 4, we proposed a methodology to predict future communities in dynamic graphs using link prediction methods.

The main contributions of this dissertation are as follows.

First, we showed that adopting a multi-objective approach that incorporates multiple sources of information, such as a response variable of interest, allows us to identify more meaningful
patterns in complex networks. Specifically, the results presented in Chapter 2 show that our proposed multi-objective algorithm to detect communities associated with a response variable of interest is able to identify communities with greater predictive power for this response variable than state-of-the-art single objective algorithms that only consider the structural properties of the graph [57].

Second, we showed that taking into account the dynamic nature of complex networks allows us to identify patterns more efficiently. Specifically, the results presented in Chapter 3 show that our proposed multi-objective algorithm to detect communities in dynamic attributed graphs is able to identify communities with high modularity and attribute similarity more efficiently in terms of both space and time than state-of-the-art multi-objective algorithms that do not take into account the dynamic nature of the network [11, 48].

Finally, we further addressed the dynamic nature of complex networks by predicting future communities in dynamic graphs, an important yet seldom studied task. The results presented in Chapter 4 show that our proposed methodology is able to predict the community structure of the graph in future time steps and is more efficient and more robust to errors in the prediction of future edges than state-of-the-art community prediction methodologies [40]. However, more accurate link prediction methods are needed to improve the accuracy of the prediction of future communities on real-world networks.

5.2 Future Work

We conclude this dissertation by describing some potential topics for future research that expand upon our work.

First, throughout this dissertation, we investigated the value of our proposed multi-objective approach exclusively in the context of the community detection task. Investigating the value of this approach for additional graph mining tasks is an important area of future work. For example, another potential application of our proposed multi-objective approach would be in the context of the anomaly detection task—that is, identifying anomalous patterns in complex networks [47]. Anomaly detection methods generally identify anomalous vertices, edges, or subgraphs by taking into account only the structural properties of the graph. As was the case with community detection, incorporating additional sources of information into the anomaly detection process may allow us to identify more “interesting” anomalous patterns.

Second, the multi-objective algorithm proposed in Chapter 2 to detect communities associated with a response variable of interest was only evaluated on climate networks. However, this algorithm and the underlying problem of response-guided community detection could also be applied in other scientific domains. For example, another potential application of our proposed multi-objective algorithm would be in the context of community detection in brain networks.
Like climate networks, brain networks are constructed from spatiotemporal data and contain communities representing neighboring or functionally related cortical regions [37].

Third, the community prediction methodology proposed in Chapter 4 assumes that only new edges are inserted between time steps. While this assumption was sufficient to perform our experimental evaluation on synthetically generated dynamic graphs, it limits the applicability of our proposed methodology on real-world networks. Thus, an important area of future work in community prediction is the extension of our proposed methodology to also consider vertex insertion, edge deletion, and vertex deletion.

Finally, the results obtained using our proposed community prediction methodology on real-world networks did not show any significant improvements with respect to the baseline due to the low accuracy of the link prediction methods on these networks. Thus, another important area of future work in community prediction is the design of novel link prediction methods that model the underlying dynamics modulating the insertion of edges in complex networks more accurately. Such link prediction methods could lead to further improvements in the prediction of future communities using our proposed methodology.
REFERENCES


[31] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative anal-


