Community Detection through Representation learning in Evolving Heterogenous Networks

A Master's Thesis proposal

EGOR DMITRIEV, Utrecht University, The Netherlands

Recent developments in big data and graph representation learning have allowed researchers to make breakthroughs in social network analysis and the identification of communities. While opening a lot of research opportunities, such approaches are highly limited to snapshots of rapidly evolving social networks. This, in fact, is a great simplification of the real-world situation which is always evolving and expanding by the user and/or machine interactions.

Relying on novel research of dynamic graph representation learning, the goal of my thesis project is to build a framework for community detection and representation in dynamic heterogeneous networks. To verify the merit of the proposed framework, it will be evaluated against baselines on dynamic heterogeneous graphs, and analyzed against gathered twitter dataset on covid measures.

CONTENTS

Author's address: Egor Dmitriev, e.dmitriev@students.uu.nl, Utrecht University, The Netherlands.

1   INTRODUCTION AND BACKGROUND

Social Network Analysis (SNA) is a huge part of the Network Science field and is concerned with the process of investigating social structures that occur in the real-world using Network and Graph Theory. These social structures usually include social media networks [28, 34], economic transaction networks [46, 71, 81], knowledge networks [12, 32], and disease transmission networks [60, 79].

One main issue to address while studying this type of real-world events lies in the identification of meaningful substructures hidden within the overall complex system. The SNA is therefore applied to extract patterns from the data, typically in the form of information flow, identification of high throughput nodes and paths, discovery of communities and clustering. In this thesis, we are going to focus on the problem of community discovery.

This thesis proposal is structured as follows: in this section, we are going to introduce basic concepts and challenges of Dynamic Community Detection. In Section 2 we introduce important concepts used throughout this proposal. In Section 3, a brief literature survey is conducted, identifying the current state of the art and approaches to Dynamic Community Detection. In Section 4, we will describe the issue we are trying to solve as well as formulate the research questions. Subsequently, in Section 5, we elaborate on our proposed methodology which will be used for solving the posed problem and answering the research questions. Finally, in Section 6, the concrete planning for the research project is laid out.

1.1   Community Detection

The problem of partitioning a complex network into *communities* which represent groups of individuals with high interaction density, while individuals from different communities have comparatively low interaction density, is known as Community Detection (CD). CD is a task of fundamental importance within SNA, as it discloses deeper properties of networks. It provides insight into network's internal structure and its organizational principles.

Many useful applications of CD have been studied by researchers including identification of criminal groups [78], social bot detection [42], targeted marketing [61], and public health/disease control [76].

With the explosion of human- and machine-generated data, often collected by social platforms, more datasets are emerging having rich temporal information that can be studied. Most of the CD methods operate only on static networks. Meaning that their temporal dimension is typically omitted, which doesn't yield a good representation of the real world, where networks constantly evolve. Such networks are referred to as dynamic networks, since their components such as nodes and edges may appear and fade from existence. Accordingly, community detection on such dynamic networks is called Dynamic Community Detection (DCD).

DCD algorithms that incorporate additional temporal data are often able to both outperform their counterpart CD algorithms [24, 29, 55, 72], as well as provide additional information about communities for analysis [66]. This additional information comes in form of community events such as (birth, growth, split, merging, and death) or in form of the ability to track the membership of certain individuals over time.

1.2 Challenges in Community Detection

DCD is seen as the hardest problem within Social Network Analysis. The reason for this is mainly because DCD, unlike CD, also involves tracking the found communities over time. This tracking relies on the consistency of the detected communities, as usually slight changes to the network may cause a different community membership assignment. Not properly accounting for this uncertainty may cause community and result drift [19].

Additionally, the increasing richness of the data is not only limited to temporal data. The real-world data often connects entities of different modalities. This multi-modality occurs through the fact that the entities and relations themselves may be of different types (meta topology-based features). For example: users, topics, and documents in a social network or vehicles and landmarks in a traffic network. More complex networks may include asymmetric relationships, and temporal networks may include appearing, disappearing, or streaming edges/nodes.

Another example of multi-modality in networks comes in the form of node and relation features (content-based features). These features may come in the form of structured (numerical, categorical, or vector data) or unstructured data such as images and text. It is of high importance to explore this multi-modal data, as it may not always be possible to explain the formation of communities using network structural information alone.

Finally, a more systematic issue is that there is no common definition for a community structure. Within networks, it is usually described in terms of membership assignment, while in more content-based settings communities are described in terms of modeled topics (that usually represent interest areas) or distributions over latent similarity space. Both definitions have their shortcomings, as they often fail to account for more complex community structures (overlapping and hierarchical communities) and non-linearity of structures typically found in the real world.

The task of community detection is usually compared to graph clustering, which may not always be a fair comparison, as the main focus point in CD algorithms is the fact that the amount of communities is unknown a priori. The communities are never planted in the real-world, and the algorithms should detect them in an unsupervised manner.

2 PRELIMINARIES

In this section, we introduce the notation as well as important concepts that we use throughout this proposal. For notation we try to stay consistent with the literature covered in the next section.

Graphs. A graph G is a tuple $G = (V, E)$ consisting of $n := |V|$ sequentially numbered nodes $V = \{v_1, ..., v_n\}$ and $m := |E|$ edges. Edges can be directed or undirected, represented as either ordered tuples $(u, v) \in E$ or unordered sets $\{u, v\} \in E$ respectively. Unless stated otherwise, we assume that graphs are undirected. An edge $\{u, v\} \in E$ is *incident* to a node v_i, if $v_i \in \{u, v\}$. Two nodes v_i and v_j are *adjacent* if they are connected by an edge, i.e., $\{v_i, v_j\} \in E$. The *neighbors* $\mathcal{N}(v_i) = \{v_j | \{v_i, v_j\} \in E\}$ of a node v_i are nodes that are adjacent to it. The *degree* $k_v := |\mathcal{N}(v)|$ of a node v is equal to its neighbors count. In more mathematical context a graph can be represented as an *adjacency matrix* A where A_{ij}-th cell is valued 1 iff edge $\{v_i, v_j\} \in E$. In some cases we also consider *weighted graphs* $G = (V, E, w)$ with weights $w(v_i, v_j) \in \mathbb{R}$.

Multidimensional Networks. A multidimensional network is an edge-labeled extension of multi-graphs that allows for multiple edges between the same nodes (known as *parallel edges*). A multidimensional network is defined as $G = (V, E, D)$, where D is a set of dimensions. Each link is a triple $(u, v, d) \in E$ where $u, v \in V$ and $d \in D$.

Heterogeneous Graph. A heterogeneous graph $G = (V, E, O)$ extends the notion of a multidimensional network by defining a node type mapping function $\phi : V \rightarrow O_V$ and an edge type mapping function $\psi : E \rightarrow O_E$. O_V and O_E denote sets of predefined node and edge types, where $|O_V| + |O_E| > 2$. A *meta-path* $\mathcal{P}$ of length l within a heterogeneous graph is defined in form of $V_1 \xrightarrow{R_1} V_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} V_{l+1}$ (abbreviated as $V_1 V_2 ... V_{l+1}$) which describes a composite relation $\mathcal{R} = R_1 \circ R_2 \circ \cdots \circ R_{l+1}$ between node types V_1 till V_l. Meta-path based neighbor set $\mathcal{N}^{\mathcal{P}}(v_i)$ is defined as a set of nodes which connect node v_i via meta-path $\mathcal{P}$.

Temporal Network. A temporal network [73] is a graph $G = (V, E, T)$, where V is a set of triplets in form (v, t_s, t_e), with v a being a graph node and $t_s, t_e \in T$ respectively being the birth and death timestamps of the corresponding node (with $t_s \leq t_e$); E is a set of quadruplets (u, v, t_s, t_e) with $u, v \in V$ being graph vertices, and $t_s, t_e \in T$ respectively being the birth and death timestamps of the corresponding edge (with $t_s \leq t_e$). Networks without edge durations ($t_e = \infty$) are often referred to as *contact sequences* and those with duration as *interval graphs*. Differently, a distinction is made between two types of temporal networks, *interaction networks* and *relation networks*. The former models interactions that may repeat as time goes by, while the latter model more stable relationships (friendship, coworker, belonging to the same group, etc.).

Snapshot Network. A snapshot network $\mathcal{G}_\tau$ [19] is defined by an ordered set $\langle G_1, ... G_\tau \rangle$ of τ consecutive snapshots, where $G_i = (V_i, E_i)$ represents a graph with only the set of nodes and edges that appear in the interval (t_i, t_{i+1}). It is worth noting that a temporal network can be discretized into a snapshot network by partitioning it into a series of snapshots at the desired resolution.

Complex Network. A complex network is a graph with non-trivial topological features, which don't occur in simple networks such as lattices or random graphs, but often occur in networks representing real systems. A few common features occurring in complex networks include *scale-free networks* where the degree distribution follows the power law (implying that the degree distribution has no characteristic scale); *small-world networks* which exhibit a *small-worldness* phenomenon where the diameter of the network (degree of separation) is usually small while the clustering coefficient is high; *multidimensional networks*; and *spatial networks* where nodes are embedded in space.

Normalized Mutual Information (NMI). Normalized Mutual Information is a popular measure used to evaluate network partitioning. It is a variant of a common measure in information theory called Mutual Information defined by $I(X; Y) = H(X) - H(X|Y)$ and representing a reduction in entropy of variable X by observing the random variable Y or vice versa. In the context of network partitioning, it, therefore, quantifies the information shared between two partitions. A lower value represents a higher quality partitioning. NMI is defined as $NMI(X; Y) = \frac{I(X;Y)}{H(X)+H(Y)}$ and ensures the resulting value is within a $[0, 1]$ range, therefore allowing for comparison of different size partitions.

3 LITERATURE REVIEW

The problem of dynamic community detection was noticed quite early on within the SNA community and a considerable amount of research has been made in order to provide a comprehensive analysis [19, 73, 82]. While the said research was mostly focused on the discovery of communities using topologically-based features and node connectivity, the covered methods did research the limitations and challenges posed by a temporal context.

In recent years, significant developments have been made in the space of deep learning. Mainly in the development of new deep learning methods capable of learning graph-structured data

[13, 36, 44] which is fundamental for SNA. Because of this, various problems within the field have been revisited, including community detection problems. The approaches have been expanded by incorporation of more complex features, solving the issues concerning multi-modality, and the introduction of unsupervised learning.

Despite this resurgence, the DCD problem has received little attention. Though efforts have been made to incorporate the deep learning methods by introducing content-based similarity [15, 24, 39], the definition of unified constraints for end-to-end learning [14, 40, 57, 92], and usage of graph representation-based CD algorithms [54, 89] within a temporal context, the current state-of-the-art leaves room for improvement.

We structure the literature as follows: first, we describe the various interpretations of the Community Structure in Section 3.1. Next, we explore various approaches and techniques related to Graph Representation Learning in Section 3.2. Then, we provide an overview of the current state-of-the-art approaches for Community Detection and Dynamic Community Detection tasks in Section 3.3 and Section 3.4. Finally, we discuss the ways to evaluate the said algorithms in Section 3.5 and the datasets available in Section 3.6.

3.1 Community Structures

The goal of this section is to introduce fundamental structures for the Dynamic Community Detection task. We do this by combining various definitions used in the relevant literature as well as establishing the purpose for these structures, before proceeding into approaches for detecting communities in the following sections.

3.1.1 Communities

Communities in real-world networks can be of different kinds: disjoint (students belonging to different educational institutions), overlapping (person having membership in different social groups) and hierarchical (components of a car). One of the main reasons behind the complexity of CD is that there is not one unique definition of what a community actually is.

The *link-based* (also referred to as classic) community detection methods intuitively describe communities as groups of nodes within a graph, such that the intragroup connections are denser than the intergroup ones. This definition is primarily based on the *homophily* principle, which refers to the assumption that similar individuals are those that are densely connected together. Therefore, these kinds of methods look for sub-graph structures such as cliques and components that identify connectedness within the graph structure to represent the communities.

Unfortunately, in most cases, link-based methods fall short to identify communities of similar individuals. This is mainly due to two facts: (i) many similar individuals in a social network are not explicitly connected together, (ii) an explicit connection does not necessarily indicate similarity, but can be explained by sociological processes such as conformity, friendship or kinship [22, 24].

A more general definition is introduced in [18] to create an underlying concept generalizing all variants found in the literature (Definition 1). In link-based methods, a direct connection is considered as a particular and very important kind of action, while newer methods also consider content or interest overlap.

DEFINITION 1 (COMMUNITY). *A community in a complex network is a set of entities that share some closely correlated sets of actions with the other entities of the community.*

3.1.2 Dynamic Communities

Similar to how communities can be found in static networks, dynamic communities extend this definition by utilizing the temporal dimension to define their life cycle/evolution over a dynamic network. A dynamic community is characterized by a collection of communities and a set of transformations on these communities over time.

This persistence of communities across time subjected to progressive changes is a critical problem to tackle. Though, as noted in Rossetti and Cazabet [73] the problem can be compared to the famous "the ship of Theseus" paradox. Because (verbatim), *deciding if an element composed of several entities at a given instant is the same or not as another one composed of some—or even none—of such entities at a later point in time is necessarily arbitrary and cannot be answered unambiguously.*

Most works agree on two atomic transformations on the communities, namely node/edge appearance and vanishing. In other works such as [8, 15, 66] authors define a more high-level set of transformations (also referred to as events) built on top of the atomic ones. These transformations are more interesting for analytical purposes and include:

- Birth, when a new community emerges at a given time.
- Death, when a community disappears. All nodes belonging to this community lose their membership.
- Growth, when a community acquires some new members (nodes).
- Contraction, when a community loses some of its members.
- Merging, when several communities merge to form a new community.
- Splitting, when a community is divided into several new ones.
- Resurgence, when a community disappears for a period and reappears.

These events/transformations are often not explicitly used during the definition and/or representation of dynamic communities. Nevertheless, most of the methods covered in the following sections do define a way in their algorithm to extract such events from the resulting data.

Finally, it is important to note that dynamic networks can differ in representation. They can be represented as either a time series of static networks (also referred to as snapshots) or as a real-time stream of edges (referred to as temporal networks). Within the global context of dynamic community detection, they can be seen as equivalent, as the conversion between the two representations can be done fairly easily. The latter, temporal networks, are often used to handle incremental changes to the graph and are most commonly applied within real-time community detection settings.

3.2 Graph Representation Learning

The representation-based approaches stem from the field of computational linguistics, which relies heavily on the notion of *distributional semantics*, stating that words occurring in similar contexts are semantically similar. Therefore, the word representations are learned as dense low-dimensional representation vectors (embeddings) of a word in a latent similarity space by predicting words based on their context or vice versa [58, 69]. Using the learned representations' similarity, clustering and other analytical metrics can be computed.

The success of these representation learning approaches has spread much farther than just linguistics as similar ideas are also applied to other fields including graph representation learning. Methods such as Deepwalk [70], LINE [83], and Node2Vec [31] use random walks to sample the neighborhood/context in a graph (analogous to sentences in linguistic methods) and output vector

representations (embeddings) that maximize the likelihood of preserving the topological structure of the nodes within the graph.

Whereas previously the structural information features of graph entities had to be hand-engineered, these new approaches are data-driven, save a lot of time labeling the data, and yield superior representation vectors. The methods can be trained to optimize for *homophily* on label prediction or in an unsupervised manner on link prediction tasks.

Newer approaches introduce the possibility for the fusion of different data types. GraphSAGE [35] and Author2Vec [96] introduce a methodology to use node and edge features during the representation learning process. Other approaches explore ways to leverage heterogeneous information present within the network by using *metapath* based random walks[23] or by representing and learning relations as translations within the embedding space [11]. In Nguyen et al. [65] the authors introduce a method to encode temporal information by adding chronological order constraints to various random walk algorithms. Other relevant advancements within the field include *Graph Convolutional Networks* (GCN) [45] and (Variational) *Graph Auto-Encoders* (GAE) [43] which present more effective ways to summarize and represent larger topological neighborhoods or whole networks.

In the remainder of this section, we briefly describe the working of a selection of influential representation learning algorithms.

3.2.1 Negative Sampling

Negative sampling is a technique used for reducing the calculation complexity of loss for link prediction tasks. Originally introduced in Mikolov et al. [59] this technique was used to optimize the *skip-gram* algorithm, which predicts context words based on a single input word (See Eq. 1). Computing the result of this softmax function is very expensive, as the vocabulary may become very large, and the negative samples outnumber the positive ones by a lot. Negative sampling (Eq. 2) is introduced as an approximation where w_O is the output context word, given the w_I the input word, v_{w_O}, v_{w_I} as their representation vectors respectively, sigmoid function σ, and k negative example words sampled from a random distribution weighted by word frequency $P_n(w)$.

Due to its efficiency, negative sampling is also embraced in graph representation learning, as the link prediction task is synonymous with context word prediction.

$$p\left(w_O \mid w_I\right) = \frac{\exp\left(v'_{w_O}{}^\top v_{w_I}\right)}{\sum_{w=1}^{W} \exp\left(v'_w v_{w_I}\right)} \tag{1}$$

$$\log P(w_O|w_I) \approx \log \sigma\left(v'_{w_O}{}^\top v_{w_I}\right) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)}\left[\log \sigma\left(-v'_{w_i}{}^\top v_{w_I}\right)\right] \tag{2}$$

3.2.2 Node2Vec

Introduced in Grover and Leskovec [31], Node2Vec learns node representations within a graph by introducing a hybrid *second-order* random walk technique. Once random walks are constructed, the *skip-gram* approach along with negative sampling is used to learn the node representations. Since random walks can be efficiently sampled, this helps to avoid memory and runtime complexity issues faced by formerly leading approaches such as spectral/matrix factorization methods.

The random walks start from a random node u and span a length of k nodes. The next node in the walk is determined by the transition probability of each of the neighbors of the current node. See Eq. 3 for calculation of transition probability for first-order random walks where next node u is selected given only the current node v and the corresponding edge weights $w(u, v)$.

$$p(u \mid v) = \frac{w(u, v)}{\sum_{u' \in Nv} w(u', v)} \tag{3}$$

Second-order random walk (Eq. 5) instead considers last two visited nodes v, t and introduces an additional tradeoff weight $\alpha_{pq}(t, x)$ parametrized by p (*in-out bias*) and q (*return ratio*). See Eq. 4 for the trade-off weight, where d_{tx} represents the hop distance between two nodes t and x. Given d_{tx} the first-order random walk is modified to be biased towards two cases: (i) returning to the same node $d_{tx} = 0$, therefore reinforcing exploration of a single neighborhood (BFS), (ii) exiting the neighborhood of a single node (DFS) (See 1). Their experiments have shown that setting a higher q parameter optimizes embeddings for *homophily* and yields better results in clustering tasks, while setting a higher p value promotes structural equivalence, which aids in node classification tasks.

$$\alpha_{pq}(t, x) = \begin{cases} \frac{1}{p} & \text{if } d_{tx} = 0 \\ 1 & \text{if } d_{tx} = 1 \\ \frac{1}{q} & \text{if } d_{tx} = 2 \end{cases} \tag{4}$$

$$p(u \mid v, t) = \frac{\alpha_{pq}(t, u) w(u, v)}{\sum_{u' \in Nv} \alpha_{pq}(t, u') w(u', v)} \tag{5}$$
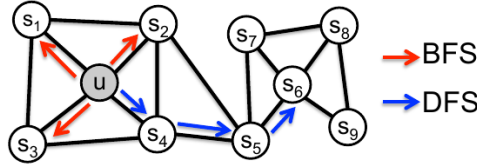

Fig. 1. BFS and DFS search strategies from node u ($k = 3$).

3.2.3 Graph Autoencoder (GAE)

In Kipf and Welling [43], the authors describe the application of the autoencoder concept for graph representation learning. Traditionally, autoencoders consist of an encoder $E : X \rightarrow Z$ translating the input X to a low dimensional latent space vector Z, and a decoder $D : Z \rightarrow \hat{X}$ translating the low dimensional vector Z back to the input space as $\hat{X}$. To optimize both components, *reconstruction error* $L(X, \hat{X}) = ||X - \hat{X}||^2$ and its variants are employed. This ensures cohesion of the latent space.

The architecture for graph autoencoders is similar to the one used in spectral graph convolution networks. The whole graph is represented as a graph Laplacian matrix $S \in \mathbb{R}^{N \times N}$ (mathematical representation of a graph; can be similarity matrix, adjacency, or similar). During encoding, an efficient factorization approximation of this matrix is produced as $Z \in \mathbb{R}^{N \times d}$, while during decoding the inner product is calculated ZZ^T reconstructing the graph Laplacian (Fig. 2).

Employing this approach for graph representation learning yields many benefits, such as flexibility in objective definition and robustness against noise. The main downside of this approach is that working on a full graph is expensive and doesn't scale well.
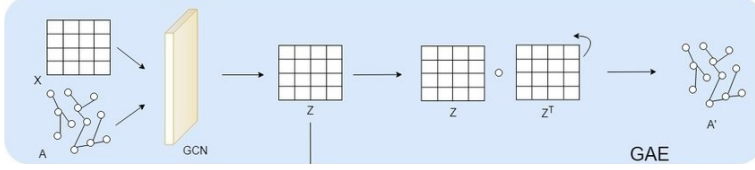

Fig. 2. Graph Autoencoder architecture

3.2.4 GraphSAGE

Existing graph representation learning approaches are *transductive*, meaning the algorithm needs the entire graph to learn the embedding of a node. This approach generalizes poorly, since the addition of a single node requires a rerun of the algorithm. In Hamilton et al. [35] the authors introduce the GraphSAGE representation function learning method which solves this issue by learning a set of aggregator functions to predict the embedding rather than statically learning it (also referred to as *inductive learning*).

First, the layer count parameter k is specified defining the number of aggregations used, and therefore the aggregation neighborhood size (namely k-hop neighborhoods). A forward pass for the "mean" operation-based aggregator is defined as Eq. 6. Where $\mathbf{h}_v^{k-1}$ is the representation vector of a node v at $(k-1)$-th layer, $\mathbf{W} \in \mathbb{R}^{d_k \times d_{k-1}}$ is a matrix of trainable weights, d_k is the dimensionality of the representation vector at k-th layer, and σ is the activation function. By initializing $\mathbf{h}_v^0$ using the node's feature vector and applying aggregator functions recursively given the neighborhood of a node $\mathcal{N}(v)$ the final representation vector $\mathbf{h}_v^k$ is calculated (See Fig. 3).

In follow-up work Ying et al. [103] the authors introduce a random walk simulating sampling-based aggregation approach, which in combination with negative sampling is capable of learning on web-scale graphs (exceeding > billion nodes).

$$\mathbf{h}_v^k \leftarrow \sigma \left(\mathbf{W} \cdot \text{MEAN} \left(\left\{ \mathbf{h}_v^{k-1} \right\} \cup \left\{ \mathbf{h}_u^{k-1}, \forall u \in \mathcal{N}(v) \right\} \right) \right) \tag{6}$$
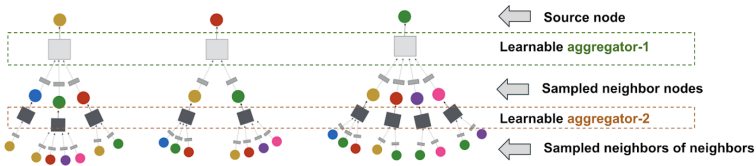

Fig. 3. Overview of GraphSAGE model architecture using depth-2 convolutions ($k = 2$)

3.3 Link-based Approaches

Link-based approaches to (Dynamic) Community Detection rely on connection strength to find communities within the network. The main criteria for communities is the assumed property that

intragroup connections are denser than intergroup ones. The networks are partitioned in such a way, that optimizes for a defined measure characterizing this property.

We start this section by covering the fundamentals of link-based community detection by describing commonly used community quality measures and algorithms for optimizing them. Next, we introduce the link-based DCD problem and the unique challenges that arise as opposed to CD. Then we proceed to cover the current state of the art by describing the related works, their solutions to the said challenges, and possible extensions to the problem.

3.3.1 Community Detection

Different metrics exist for quantifying the characteristic of *homophily* over the connection strength. The most common metric is modularity, which measures the strength of the division of a network into modules (communities). Its popularity stems from the fact that it is bounded and cheap to compute, though it has other issues such as resolution limit (making detecting smaller communities difficult). Other metrics that can be found in the literature include but are not limited to:

- Conductance: the percentage of edges that cross the cluster border
- Expansion: the number of edges that cross the community border
- Internal Density: the ratio of edges within the cluster with respect to all possible edges
- Cut Ratio and Normalized Cut: the fraction of all possible edges leaving the cluster
- Maximum/Average ODF (out-degree fraction): the maximum/average fraction of nodes' edges crossing the cluster border

Modularity. Modularity directly measures the density of links inside a graph and is therefore computed on communities (sets of nodes) individually by weighing edges using community similarity (or exact matching). Calculation of modularity is done by aggregating per community r for each pair of nodes vw the difference between the expected connectivity $\frac{k_v k_w}{2m}$ (expected amount of edges between the nodes) and the actual connectivity A_{vw} (existence of an edge) given their degrees (k_v and k_w). The final result represents the connectivity difference between the current and a random graph, as expected connectivity is determined by random rewirings. Because intracommunity pairs are weighted less than intercommunity pairs, the score can vary. See Eq. 7, where S_{vr} indicates membership of node v for community r, and m represents the total edge count.

$$
Q = \frac{1}{2m} \sum_{vw} \sum_{r} \left[\overbrace{A_{vw}}^{\text{Connectivity}} - \underbrace{\frac{k_v k_w}{2m}}_{\text{Expected Connectivity}} \right] \overbrace{S_{vr} S_{wr}}^{\text{Community Similarity}} \tag{7}
$$

Louvain Method. Finding an optimal partition of a graph into communities is an NP-hard problem. This is because, while calculating the modularity score can be done in log-linear time, all possible node to community assignments still have to be considered. Therefore, heuristic-based methods such as the Louvain method are usually used.

Louvain method [10] is a heuristic-based hierarchical clustering algorithm. It starts by assigning each node in the graph to its own community. Then it merges these communities by checking for each node the change in modularity score produced by assigning it to a neighbor community (based on the existence of a connection). Once the optimal merges are performed, the resulting communities are grouped into single nodes and the process is repeated.

Since modularity changes can be computed incrementally, the complexity of this method is $O(n \log n)$. Additionally, due to the flexibility of the modularity measure, it allows detection of communities in directed and/or weighted graphs.

Label Propagation algorithm. Another way to sidestep the complexity issue is using the Label Propagation algorithm, as it uses the network structure directly to define partitions and doesn't require any priors (such as quality metrics). The intuition for the Label Propagation algorithm is as follows: A single label quickly becomes dominant within a group of densely connected nodes, but usually has trouble crossing sparsely connected regions.

The algorithm starts by assigning each node their own label. After that, for each iteration, each node updates its label to the majority label among its neighbors where ties are broken deterministically. The algorithm stops after a fixed amount of iterations or once it has converged. An important feature of this algorithm is that a preliminary solution can be assigned before each run, therefore only updating existing membership assignments.

3.3.2 Dynamic Community Detection

Dynamic Community Detection can be seen as an extension of Community Detection by the addition of the Community Tracking task. Tracking relies on the coherency and stability of found communities to define their evolution through time. The said properties can not be taken for granted and introduce new challenges when designing DCD methods. The main issue is in fact that they are competitive with each other, causing a trade-off between community coherency and temporal stability.

Various strategies dealing with this trade-off are categorized by Rossetti and Cazabet [73] and Dakiche et al. [19] where authors reach a consensus over three main groups. In the following sections, we briefly introduce these strategies and describe the current state of the art in similar order.

Independent Community Detection and Matching. Also referred to as the two-stage approach. Works by splitting the DCD task into two stages. The first stage applies CD directly to every snapshot of the network. Followed by the second stage, matching the detected communities between the subsequent snapshots.

The advantages of this approach include, the fact that it allows use of unmodified CD algorithms for the first step, and that it is highly parallelizable, as both detection and matching steps can be applied to each snapshot independently. The main disadvantage is the instability of underlying CD algorithms, which may disrupt the community matching process. Many CD methods may give drastically different results in response to slight changes in the network. During matching, it becomes difficult to distinguish between algorithm's instability and the evolution of the network.

In Wang et al. [91] the authors circumvent this instability issue by looking at the most stable part of the communities, namely core/leader nodes. In their research, they observe that in various datasets most of the nodes change dramatically while only a small portion of the network persists stably. To exploit this feature, the algorithm CommTracker is introduced which first detects the said core nodes, and then defines rules to both extract communities as well as their evolutionary events. The community members are assigned based on their connectivity relative to core nodes.

Rossetti [72] proposes a way to detect overlapping communities in dynamic networks. A more robust two-phase community detection method (ANGEL) is proposed to ensure the stability of the found communities. The first phase extracts and aggregates local communities by applying Label

Propagation on the Ego-Graph [1] for each node in the network. The found communities are biased due to their partial view of the network and are merged in the second step based on their overlap, yielding more stable communities with the possibility of overlap. During the matching for each snapshot, a step forward and backward in time is considered where community splits and merges are detected by reusing the matching criteria of the second phase of the CD step.

Dependent Community Detection. Dependent Community Detection strategy works by detecting communities in each snapshot based on the communities found in the previous snapshots. This approach introduces temporal smoothness because the previous solution is reused, making the matching process obsolete. Though as part of the described trade-off it can impact the long-term coherence of the dynamic communities. Mainly, because each step introduces a certain amount of error into the results (community drift) which may get amplified within further iterations (error accumulation). Another disadvantage is the fact that the strategy has limited possibilities of parallelization due to its dependent nature.

To lessen the complexity of continuous re-detection of the communities, some algorithms process the changes incrementally by limiting the change to a local neighborhood. While this approach has many benefits, it is important to note that these algorithms face a problem where only applying local changes can cause communities to drift toward invalid ones in a global context.

He and Chen [37] introduces an efficient algorithm by modifying the Louvain method. Based on the observation that between consecutive timesteps only a fraction of connections changes and do not affect communities dramatically, they argue that if all community nodes remain unchanged, the community also remains unchanged. With this in mind, they make a distinction between two types of nodes, ones that change the connection in a snapshot transition and ones that do not. The former have to be recomputed, while the latter maintain their community label. The nodes that maintain their community are merged into community nodes, where edges are grouped into intercommunity edges and are weighted according to their real connectivity (amount of edges when ungrouped). This simplified graph is passed to the Louvain method algorithm for community detection. By reusing the community assignments temporal smoothness is maintained and due to the incremental nature of this algorithm, the overall complexity remains low.

Guo et al. [33] envision the target network as an adaptive dynamical system, where each node interacts with its neighbors. The interaction will change the distances among nodes, while the distances affect the interactions. The intuition is that nodes sharing the same community move together, and the nodes in different communities keep far away from each other. This is modeled by defining the so-called Attractor algorithm, which consists of three interaction patterns that describe how node connection strength is influenced by neighboring nodes. The edge weights are initialized using Jaccard distance and the propagation is run until convergence. The communities can be extracted by thresholding on edge weight/distance. Thereafter, all changes are treated as network disturbances. The disturbances can be limited to a certain area using a disturbance factor, which defines a bound on the possible propagation.

More recently, the Yin et al. [102] has proposed an evolutionary algorithm by looking at the DCD from an Evolutionary Clustering Perspective. They detect community structure at the current time under the guidance of one obtained immediately in the past by simultaneously optimizing for community quality score (modularity) and community similarity between subsequent time steps (NMI). In the methodology, they describe a method to encode a graph efficiently into a *genetic sequence*. Additionally, new mutation and crossover operators are suggested which maximize either

---

[1] Graph excluding a single node. More formally $G = (V', E)$ is an Ego-Graph of v if $V' = V \setminus \{v\}$

of the two objectives. By using a local search algorithm, building a diverse initial population, and selecting for dominant candidates, the communities maximizing both objectives are obtained.

Simultaneous community detection. The final strategy we consider, sidesteps the matching issue by considering all snapshots of the dynamic network at once. This is done by flattening the network in the temporal dimension and coupling edges between the same nodes at different timesteps. These approaches usually don't suffer from instability or community drift. The disadvantages include that the standard principle of a unique partition for each time step can't be applied, as only the combined network is used, therefore limiting the number of possible algorithms. Handling real-time changes to the graph is often not considered.

Mucha et al. [63] adopt a simple yet powerful solution to this problem by connecting identical nodes between different time steps within the unified network. On this network, they apply a modified Louvain method algorithm to extract the communities whose members can be split over different timesteps.

Ghasemian et al. [26] apply stochastic block model-based approach. They make a distinction between two edge types: (i) spatial edges (edges between neighbors) and (ii) temporal edges (edges between nodes in subsequent timesteps). Using this distinction, they define a *Belief Propagation* equation to learn marginal probabilities of node labels over time. Additionally, in their research, they introduce a way to derive a limit to the detectability of communities. This is, because some communities may not be detectable as their probability nears that of random chance.

3.4 Representation-based Approaches

The main difference between representation-based approaches and link-based approaches is the fact that they usually don't directly model the network based on connections. Instead, they learn an intermediate representation of the graph components on which CD detection is applied. While also relying on the idea of *homophily*, most methods define additional objectives to improve community quality.

The main argument for using representation-based methods is that real-world networks are non-linear, meaning that there may be no connections when they make sense and vice versa [92]. By using deep neural networks to learn these embeddings one can address such non-linearity as they are in general very robust against noise. Other notable benefits to using representation-based approaches include the fact that they compress the data efficiently, as real-world networks are very sparse. They can also represent multi-modal features, network (meta) structure, and temporal dimension by defining them all in a compatible similarity space or learning mappings to this space. Finally, representations are naturally easy to compute similarity on.

In this section, we describe representation-based approaches by covering both CD and DCD approaches. To provide a more cohesive overview of the methods, we group them by their innovations instead.

3.4.1 Affiliation Graph Models

While arguably not being representational by itself, Affiliation Graph Network (AGM) model introduced in Yang and Leskovec [99] is very influential within the deep learning/representation learning CD field.

The AGM models a network as a bipartite graph and is represented by the following equation $B(V, C, M, \{p_c\})$, where V represents nodes, C set of communities, M node-community memberships and $\{p_c\}$ model parameters (a single probability p_c per community). It can model non-overlapping,

overlapping, and hierarchical communities by defining rules on membership sets in M. AGM can be used in both generative and discriminative settings.

The generative scenario works as follows: Given an AGM $F(V, C, M, \{p_c\})$, generate links between each pair of nodes exceeding a baseline probability p. This is done by considering that, according to AGM, each pair of nodes in community C_i is connected with a probability p_{c_i}. Therefore, the probability of two nodes having a connection is proportional to the number of communities they share (which can be derived from the model).

The discriminative scenario is defined as: Given a graph G, find a model $F(V, C, M, \{p_c\})$ that may have generated it. By assuming that the graph was generated using an AGM, the parameters M, number of communities $|C|$ and $\{p_c\}$ have to be found. The process of finding such a model for the graph involves max likelihood fitting. AGM is relaxed to have membership strengths F_{uC}, which helps to define the probability of nodes u and v connecting through community C as $P_C(u, v)$, and by themselves $P(u, v)$ (by marginalizing over communities). Using this, a probability $P(G|F)$ can be constructed, quantifying how well the model fits the data. Finally, gradient ascent can be applied to optimize for the model's parameters.

3.4.2   Graph Reinforcement

The first group of methods we consider are Graph Reinforcement methods. These methods use representation-based learning techniques to enhance the graph by adding valuable edges or reducing the noise by removing noisy connections. This is usually done by training a model on a link-prediction task. A notable benefit of this approach is that other well-known CD methods can be used on the enhanced graph afterward.

Kang et al. [41] present a pre-processing method for strengthening the community structure of a graph by adding non-existing predicted intracommunity edges and deleting existing predicted intercommunity edges. Their strategy is to learn topological embedding using a graph representation learning algorithm (Node2Vec) based on the existing link prediction task. The similarity is computed between different node pairs as they are put into buckets. Then with the assumption of *homophily*, the buckets with a higher value can be considered holding intracommunity connections, while buckets with lower value hold intercommunity connections. The buckets from both extremes are used to create or delete edges. The preemptive CD is done to greedily guide pair-wise similarity computation and avoid a high complexity. Once the reinforced graph is constructed, already existing CD algorithms can be applied.

Jia et al. [40] solves the issue of detecting overlapping communities by proposing the Community-GAN algorithm, which jointly optimizes for node and community representations. First, they define a method for efficient motif (in their case clique) sampling from the graph. Cliques are considered positive samples, while simple vertex subsets are considered negative samples. Then, they define a GAN based structure for learning representational vectors where the generator G tries to learn $p_{true}(m|\mathbf{v}_c)$ as preference distribution of motifs to generate vertex subsets $\mathbf{v}_c$ most likely to be real motifs m. Discriminator D tries to learn the probability of a vertex subset being a real motif, therefore creating a minimax game of progressively optimizing embeddings to be able to encode rich information about network topology.

Both components (G and D) are implemented as a modified relaxed AGM model with a more general definition to be able to handle the motif generation (rather than edge generation). The probability of a set of vertices being a motif is defined in terms of their probability being a motif through a community, therefore making them community-aware as they now represent the affiliation weight

between a vertex and a community. The number of communities is tuned on the link prediction task.

3.4.3 Multi-objective optimization

Another subject where representation-based approaches excel is multi-objective optimization. Usually, a combined objective is defined in terms of a community quality, temporal consistency, or homophily measure. These measures in turn use the proximity between the representation vectors to be able to back-propagate the combined error and optimize the representation (function) directly.

In Rozemberczki et al. [75] authors propose a method that learns cluster centers along with node embeddings. They define an objective function as a combination of three terms: (i) *normalization term* - ensures embeddings are centered at the origin, (ii) *proximity term* - forces nodes with similar neighborhoods to be embedded close, (iii) *cluster quality term* - forces nodes to be close to their nearest cluster. Additionally, a "*social network cost*" term is added as a regularizer to optimize for proximity between nodes within the same cluster. During training, the cluster quality term is annealed to ensure convergence, and negative sampling is employed to avoid large softmax costs.

Yang et al. [97] suggest a similar idea of combining embedding and clustering tasks to solve them in an end-to-end manner. In their work, they employ a Deep Denoise Autoencoder (DAE) to learn topological information of the network by optimizing for reconstruction loss. To learn cluster/community centers they define GRACE cluster module which first computes soft cluster assignment matrix Q by utilizing the embeddings and cluster centers which contain probabilities q_{ik} of node i belonging to cluster k. The clustering loss is defined as (Kullback–Leibler) KL-divergence between the soft clustering Q and auxiliary target distribution P which is computed by squaring and normalizing the soft assignments to reinforce more confident clustering results, while preventing the formation of excessively large clusters. Both embeddings and clustering are optimized alternatively until convergence.

Ma et al. [57] proposed a novel approach to constructing community-aware dynamic network embeddings by leveraging multi-objective optimization and extending it with a temporal stability objective. They adopt a Graph Autoencoder structure, which works by encoding the full graph into a lower-dimensional structure and decoding it again into a graph. Assuming a well-tuned autoencoder, this allows authors to encode the input network (and its nodes) into a more efficient representation of vectors which characterize the network well.

The objective function they use is defined by three terms (i) the *reconstruction error term* - minimizing the distance between the input graph and the autoencoder output, (ii) the local *structure/homophily preservation term* - minimizing first- and second-order proximity between connected nodes, and (iii) the *community evolution preservation term* - maximizing temporal smoothness of communities at different granularity levels, given their representation as an aggregation of their members.

The initial community assignment is generated using the Louvain method for high-level communities and using K-means for fine-grained communities, given a max community size parameter w. After that, embeddings at each snapshot are optimized by employing a dependent community detection-like strategy.

Wang et al. [92] employs a similar strategy for DCD detection by utilizing the Graph Autoencoder architecture. Authors add an additional community score term to the objective function, therefore also minimizing the distance between nodes in the same community. At last, K-means is run on the representational vectors to detect communities at different timesteps while reusing the outputs from the previous step.

3.4.4 Multi-modal community detection

Another way to improve community quality is by incorporating multi-modal features. These can come in the form of node attributes, content-based or meta-topological data. These representations are incorporated into learned embedding vectors either by direct learning, incorporating them into the objective function, or use of pre-trained models.

In Fani et al. [24] the authors describe their method for identifying user communities through multi-modal feature learning. First, user embeddings are learned based on their temporal and content similarity by looking at topics of interest. Per-user, a heat map is constructed, measuring the user's interest over time and topic axes. By considering users like-minded if their heat maps overlap enough, they train low-dimensional content embeddings spanning this user similarity space. Next, they use random walk-based GNN methods to learn topological similarity embeddings for network nodes. Finally, they modify the graph by setting edge weights proportionally to node proximity in this combined embeddings space. After that, the Louvain method is applied to extract these time and content-aware communities.

3.5 Evaluation

In the previous section, we have described the different variations of community structure definitions, as well as the approaches used for detecting communities. In this section, we will cover how the found dynamic community structures can be evaluated in a more general setting to allow a comparison of different approaches. Dynamic community detection problems can be seen as a combination of two tasks, matching and tracking. Due to the large difference between the two tasks, the evaluation is usually conducted separately for each of the tasks.

In the following sections, we cover the evaluation methods for both tasks by grouping the approaches used in the literature in three classes, namely, annotated, metric-based, and task-specific.

3.5.1 Annotated

Evaluation of detected (dynamic) communities becomes much easier when the *ground-truth communities* are provided. The evaluation is then done by comparing the difference between the produced communities and the effective ones. To perform this comparison, the information theory-based metric Normalized Mutual Information (NMI) is used, which quantifies the "amount of information" that can be obtained about one community by observing the other [51].

A possible drawback of this measure is that its complexity is quadratic in terms of identified communities. In [74] an alternative measure (NF1) with linear complexity is introduced which similarly to the F1 score uses the trade-off between precision and recall (of the average of harmonic means) of the matched communities. In the follow-up work [72] the authors describe a way to apply this measure within the context of DCD by calculating this score for all the snapshots and aggregating the results into one single measure.

Aside from NMI other measures are employed such as Jaccard Coefficient, Accuracy and Rand-Index measuring community overlap [56, 62, 97], Overlapping-NMI [101] and Omega-Index[2] [99].

In the real world, there are usually no ground-truth communities. Therefore, this approach is typically applied on synthetic datasets where the communities and their dynamicity is sampled from a distribution. An alternative approach some papers take is by defining ground-truth communities

---

[2]Omega-Index measures is the accuracy on estimating the number of communities that each pair of nodes shares

using the metadata and node attributes present within the datasets. Some datasets may include annotated communities, but this is not common within DCD datasets.

3.5.2 Metric based

Another way to evaluate and compare different CD algorithms without knowing ground-truth communities is using a quality function.

Network-based metrics. The first group of measures we consider operates directly on the network structure. They are most commonly used to evaluate link-based methods, as their results are network partitioning sets. Modularity is the most widely used measure [64, 80], since it measures the strength of the division of a network into modules. Networks with high modularity have dense connections between the nodes within the modules, and sparse connections between nodes in different modules. Other measures are used as well, including:

- **Conductance**: the percentage of edges that cross the cluster border
- **Expansion**: the number of edges that cross the community border
- **Internal Density**: the ratio of edges within the cluster with respect to all possible edges
- **Cut Ratio and Normalized Cut**: the fraction of all possible edges leaving the cluster
- **Maximum/Average ODF** (out-degree fraction): the maximum/average fraction of nodes' edges crossing the cluster border
- **Triangle Participation Ratio TPR**: measures fraction of triads within the community. A higher TPR indicates a denser community structure

Proximity-based measures. Proximity-based measures are often used to evaluate clustering tasks but are also often employed for representation-based CD methods since there is a large overlap in their methodology. Additionally, representation-based approaches have the benefit of being able to quantify both nodes and communities as a d-dimensional vector, enabling a more direct comparison of the two [89]. The most common measures include:

- **Silhouette Coefficient**: is defined as $S(i) = \frac{b(i)-a(i)}{\max\{a(i),b(i)\}}$, where $a(i)$ defines the mean distance from node i to other nodes in the same cluster while $b(i)$ is mean distance to any node not in the same cluster. It measures cohesion of a cluster/community and indicates how well a node is matched to its own cluster.
- **Davies-Bouldin Index**: is the ratio of the sum of the average distance to the distance between the centers of mass of the two clusters. In other words, it is defined as a ratio of within-cluster, to the between cluster separation. This measure is defined as an average over all the found clusters and is therefore also a good measure to deciding how many clusters should be used.
- **Calinski-Harabasz Index**: is the ratio of the between-cluster to the within-cluster variance. It measures cohesion (how well its members fit the cluster), as well as compares it to other clusters (separation).

Stability-based measures. To evaluate temporal stability and consistency of the node and community structures, measures based on temporal smoothness are proposed in Ma et al. [57]. These measures compare the evolution rate of network structures against the evolution rate of the whole network given node embeddings between two consecutive snapshots. The intuition is that rapidly evolving structures relative to the global evolution rate are temporally unstable and thus of low quality. Metrics proposed include:

- **Network Stability**: Is defined as Eq. 8 and evaluates the evolution ratio of the low-dimensional node representations to the network representations between snapshots at a-th time stamp.
- **Community Stability**: Is defined as Eq. 9 and computes stability of communities in dynamic networks on the embedded low-dimensional representations. It is represented as the evolution ratio of the low-dimensional community representations to the network representations between snapshots at a-th time stamp.

$$p_s^a = \frac{\left(\left\|\mathrm{H}^{a+1} - \mathrm{H}^a\right\|_2^2\right) / \left\|\mathrm{H}^a\right\|_2^2}{\left(\left\|\mathrm{A}^{a+1} - \mathrm{A}^a\right\|_2^2\right) / \left\|\mathrm{A}^a\right\|_2^2} \tag{8}$$

$$p_c^a = \sum_{k=1}^{q} \left(\frac{\left(\left\|\mathrm{H}_{c_k}^{a+1} - \mathrm{H}_{c_k}^a\right\|_2^2\right) / \left\|\mathrm{H}_{c_k}^a\right\|_2^2}{\left(\left\|\mathrm{A}_{c_k}^{a+1} - \mathrm{A}_{c_k}^a\right\|_2^2\right) / \left\|\mathrm{A}_{c_k}^a\right\|_2^2} \right) / q \tag{9}$$

Where H_i^a and $H_{c_k}^a$ represent the embedding vector for a node i and a community k at a-th timestamp, respectively. Similarly, separate node and community based adjacency matrices are defined as A^a and $A_{c_k}^a$. Therefore, in this case $\left\|\mathrm{H}_{c_k}^{a+1} - \mathrm{H}_{c_k}^a\right\|_2^2$ represents the normalized euclidean distance between the two clusters while $\left\|\mathrm{A}_{c_k}^{a+1} - \mathrm{A}_{c_k}^a\right\|_2^2$ represents the normalized euclidean distance between the adjacency matrices of the two clusters. For network stability, no such grouping is done and evaluation is performed on node-based (global) representation and adjacency matrices.

3.5.3 Task specific

In [68] the authors criticize annotation and metric-based CD evaluation approaches by proving that they introduce severe theoretical and practical problems. For one, they prove the no-free lunch theorem for CD, i.e., they prove that algorithmic biases that improve performance on one class of networks must reduce performance on others. Therefore, there can be no algorithm that is optimal for all possible community detection tasks, as the quality of communities may differ by the optimized metrics. Additionally, they demonstrate that when a CD algorithm fails, the poor performance is indistinguishable from any of the three alternative possibilities: (i) the metadata is irrelevant to the network structure, (ii) the metadata and communities capture different aspects of network structure, (iii) the network itself lacks structure. Therefore, which community is optimal should depend on its subsequent use cases and not a single measure.

To address this issue, it is common to evaluate the algorithm on both earlier described approaches as auxiliary tasks. The general sentiment behind it is, that communities have better quality if they improve an underlying application. In the following sections, we describe a few commonly used auxiliary tasks in the literature.

Link-prediction. A common way to evaluate the quality of extracted node embeddings within Graph Representation learning is using the link-prediction task. As link-prediction can be done in an unsupervised manner, it does not require additional labeling for evaluation. The edge set of the input network is split into a train set on which the model is trained, and a test set which is used to evaluate the link prediction. When using node representation learning, the predictions are defined by proximity between the nodes and a threshold. To quantify the quality of the results, classification metrics are usually employed such as accuracy, precision, recall, and f-score.

In the context of CD and DCD, the link prediction is modified to measure the predicting capability of the community embeddings. Fani et al. [24] defines a user prediction task to predict which users posted a certain news article at a certain time. Their methodology incorporates finding the closest community to a given a news item at time t. All members of the given community are seen as predicted users over which the classification metrics are calculated. Similarly, Ma et al. [57] modify the task by predicting whether the edges will still exist within the next timestamp to also quantify the temporal prediction capability of the trained embeddings.

Recommendation Tasks. Another way the quality of node representations can be evaluated is by using recommendation tasks. Instead of predicting a single item like in link-prediction, items are ranked based on their recommendation confidence. Using the ranked list, standard information retrieval metrics such as precision at rank, mean reciprocal rank, and success at rank k can be computed. This approach is applied to CD [24, 39, 75] by ranking recommendations per community instead of on an individual basis. Communities with higher scores therefore would indicate high similarity between their members.

3.6 Datasets

In this section, we curate popular datasets used in the literature for CD and DCD tasks. First, we discuss methods for generating synthetic datasets containing ground-truth communities. We conclude this section with an overview of real-world dynamic networks.

3.6.1 Synthetic Datasets

Synthetic datasets are datasets generated by specific models using manually designed rules. These datasets are generated with communities in mind and therefore most of the time contain ground-truth communities which can be used for the evaluation of CD algorithms. In this section, we describe a selection of popular synthetic dataset generation techniques.

The most common benchmark dataset was introduced in Lancichinetti et al. [50]. Their method for generating a network creates communities and their nodes with size and degree drawn from a power-law distribution. Thereafter, node edge rewiring is applied to enforce a mixture parameter μ which determines the ratio of in- to between-community connections. In their follow-up [49] the authors extend their method to work for overlapping communities.

In Greene et al. [30] the authors modify the Lancichinetti et al. [50] method to generate dynamic community networks. They do this by defining a set of community events that can be used as seeds for the temporal evolution of the network. The subsequent snapshots of the graph are defined by this set of events.

Granell et al. [29] propose a model for generating simple dynamic networks based on stochastic block models (SBM). The time evolution consists of periodic oscillation of the system's structure between configurations. Once the initial network is created, it is divided into subgraphs each having certain inter- and intraconnectivity probability. The edges between the timestamps are drawn from binomial distributions, which themselves are modified by stochastic community-based events such as shrinking, growing, merging, and splitting.

Ghalebi et al. [25] similarly suggests an SBM model-based approach to generate dynamic community networks. Additionally, they propose a methodology to learn model parameters from existing real-world networks.

3.6.2   Real-world Datasets

Real-world network datasets are often categorized based on the source they model, such as social networks, biological networks, and webpage networks. In Table 1 we present an overview of the most commonly used datasets in the literature. For each of the datasets, we describe their scale in terms of their node and edge counts. Additionally, we specify the presence of meta-topological information in the form of node types, whether the network is dynamic, and whether it includes content-based (unstructured/textual) information.

Table 1. Summary of real-world datasets. The first two columns indicate the source and reference work for the given dataset. Subsequent columns define the statistics about the node and edge counts. Node/Edge types defines the amount of different types of nodes/edges, indicating whether a network is heterogeneous. The "Temporal" and "Interaction" columns indicate whether the network is a temporal network and/or is an interaction network respectively. The "Annotated" column indicates whether ground-truth communities available for the given network. Finally, "Contentual" column indicates presence of unstructured text data features within the dataset.

Dataset	Reference	Nodes	Edges	Node/Edge Types	Temporal	Interaction	Annotated	Contentual
Zachary karate club	[104]	34	78	1/1	N	N	Y	Y
Football	[27]	115	613	1/1	N	N	N	N
Star Wars Social	[5]	113	1599	1/2	Y	Y	N	N
Enron	[85]	605K	4.1M	2/3	Y	Y	Y	Y
IMDB 5000	[3]	16K	52K	4/4	Y	N	N	N
DBLP-HCN	[84, 100]	11K	16K	3/2	Y	N	N	Y
DBLP-V1	[84, 100]	1.2M	2.4M	3/3	Y	N	N	Y
DBLP-V3	[84, 100]	2.7M	8.2M	3/3	Y	N	N	Y
Weibo	[2]	8.3M	49M	2/3	Y	Y	N	N
FB-wosn	[86]	64K	1.3M	2/2	Y	Y	N	N
LastFM	[16]	272K	350K	3/3	Y	Y	Y	N
Reddit	[9]	61M	1.2B	4/1	Y	Y	Y	N
Digg	[38]	142K	3.7M	3/2	Y	Y	N	N
Wiki-RFA	[93]	10K	159K	1/1	Y	Y	Y	N
Bitcoin	[47]	5K	35K	1/1	Y	Y	N	N
Rumor Detection	[48]	54M	1.9B	2/2	Y	Y	N	N
sx-mathoverflow	[4]	24K	506K	1/3	Y	Y	N	N
sx-superuser	[4]	194K	1.4M	1/3	Y	Y	N	N
Eu-core network	[4]	1005	25K	1/1	N	Y	Y	Y
com-Youtube	[4]	1.1M	298K	1/1	N	N	Y	Y
116th House of Representatives	[1]	6249	12K	3/4	N	N	Y	N
social-distancing-student	[88]	93K	3.7M	3/7	Y	N	N	N

4 RESEARCH QUESTIONS

The main goal of my thesis is to build a framework for community detection and representation in dynamic heterogeneous networks.

This is, to enable dynamic community analysis on the datasets described in Wang et al. [88]. The data described is collected from the Twitter social platform and is dynamic, heterogeneous, and rich in content-based (unstructured text) data. To the best of our knowledge, there are currently no dynamic community detection algorithms that can handle this data without information loss.

As there are no alike algorithms, direct comparison is not possible. To both validate the merit of our methods as well as the quality of the results, we spread our research over four research questions.

Research Question 1 (Information). Does addition of meta-topological and/or content-based information improve quality of detected communities?

While focusing on link-based features, CD algorithms treat all nodes alike. Therefore, ignoring arguably most important structural features, namely node types. Additionally, supplementary node types can be constructed from categorical features, enhancing network topology and solving issues requiring topic modeling.

Similarly, recent improvements in natural text processing allow for efficient representation of natural text, which is proven to improve the quality of node embeddings. As the formation of communities is not purely a sociological process, the CD problem should benefit from the incorporation of such content-based features.

Research Question 2 (Scale). Does usage of graph representation function learning techniques improve scale of CD beyond current state-of-the-art?

The previously mentioned representation-based DCD method use spectral graph representation methods which operate on the whole network at once. More recent graph representation approaches instead, learn a graph representation function by sampling the network using random walks or convolutions.

This has a two-fold positive effect on the scalability of the algorithms. Computation can be done more efficiently as opposed to spectral methods which rely on adjacency matrices. By learning a representation function, embeddings are computed on-demand instead of being held in memory for the entire network, therefore, limiting the impact of big networks. Other benefits may include the fact that they would be a suitable choice for processing streaming edge network variants.

Research Question 3 (Modelling). Does making temporal features implicit in node representations provide better quality communities as opposed to making them explicit?

Throughout the literature, various ways are used to incorporate temporal aspects into node embeddings. The implicit approach aims to make the embeddings temporally aware, while the explicit approach creates a separate embedding for each snapshot. While methods using either approach have presented good results in the literature, it is important to analyze the potential trade-off and benefits of both.

Research Question 4 (Results). Do community-aware node embeddings perform well on both node as well as community evaluation based tasks?

While the main task of the algorithm is to find high-quality dynamic communities, the result also includes community-aware dynamic embeddings. Aside from testing the quality of the communities, it is important to compare how this community awareness affects the embeddings on dynamic node representation tasks such as link prediction and node classification.

5 APPROACH

To answer our research questions, the methodology is split into multiple parts which address the algorithm architecture, construction of the objective function, DCD result extraction, selection of the baselines, and setup of evaluation benchmarks. While split, it is important to note that these tasks have a large overlap and won't be considered in isolation.

The final result of my thesis will consist of the graph representation learning-based framework for DCD within dynamic heterogeneous graphs, a set of results comparing the algorithm to the current state-of-the-art approaches on various related tasks, and a set of ablation tests providing empirical corroboration for important design choices.

5.1 Representation learning

The core part of the proposed framework is the representation learning algorithm, as it is responsible for both graph sampling, as well as provides the architecture for the deep neural network (DNN) used to learn the node representation. The representation learning can be split into three main components. Each of the components respectively has sufficient prior work and has been used in literature. The challenge is in combining them into an efficient architecture for representation learning capable of maximizing set objective function.

5.1.1 Graph Sampling

Graph sampling is a way to enforce the learning of desired topological properties from the network. Various ways to sample graphs can be found in the literature. For both shallow as well as graph representation function learning approaches the trade-off lies between depth-first search (DFS) [31, 83] and breadth-first search (BFS) [70] based approaches. While the literature has shown that DFS based approaches work better for optimizing homophily [31], our goal is to explore the benefits of both as a hybrid approach within our multi-objective setting. Additionally, we plan on supporting both heterogeneous graph sampling [23, 90, 96, 98, 103] and temporally aware sampling [20, 65, 95] by adopting extensions described in the literature.

5.1.2 Neural Network Architecture

The architecture and training strategy of the underlying neural network is crucial for a well-performing algorithm. Many methods introduced in the previous section already outline an effective architecture suitable for training on the introduced sampling method. Because both data and requirements for our task differ, the algorithms can not be used out of the box. The final architecture should support heterogeneous graph samples, temporally-aware samples, and node features.

The core of our network will be a representation function-based algorithm [35, 103] adopted to deal with different node and edge types, therefore generating embeddings for a node's sampled neighborhood in time. Related literature will be used as inspiration to improve model's performance in content-rich networks [96, 103], using attention-based mechanisms [6, 77, 90], and alternative training strategies such (variational or diffusion) auto-encoders, GAN [43, 52].

5.1.3 Feature Fusion

Many of the real-world datasets are feature-rich. Some features are structured and can be passed to the neural network with minimal pre-processing. Some feature types require additional attention. Natural (unstructured) text features can be aggregated into a single representation vector using pre-trained embeddings [21, 69], and (large) categorical features may be transformed into network nodes, thus moving the information into the topological domain [17, 94].

5.2 Objective Function

A crucial part of representation-based DCD methods is the objective function. By utilizing node (and community) representation vectors, one can optimize the network to maximize a differentiable multi-objective function using back-propagation. During the training process, the focus can be shifted between the different objectives. By focusing on defining necessary criteria for dynamic community detection, which will give a specification for our multi-objective function.

Cohesion. The most common definition states, that communities are characterized by denser inter-community connections compared to intracommunity density. Representation methods extend this definition by noting that the density of the connections is can be represented by the topological similarity measure of two nodes. Clustering methods further extend this definition by defining similarity on multi-modal embeddings, therefore keeping the definition consistent for feature-rich networks. A viable choice for community cohesion measure would be the Silhouette Coefficient (See Section 3.5)

Homophily. To ensure reliable computation of the cohesion measure, the representation vectors need to be accurate. A way to train these representations is by assuming homophily, which states that the more two nodes occur in the same context, the more similar they are. This is translated to graph representation learning problems by using node neighborhood as context. Either using first-order proximity where nodes should occur in their counterpart's context or by utilizing second-order proximity where two nodes are similar if they share the same context (through common nodes). Hybrid approaches exist which optimize for both, since they model different semantics. Similarly, this idea can also be extended to feature aware embeddings, therefore, extending the definition of a community through transitivity when the above definition for cohesion is utilized. Homophily is usually measured by the distance of connected node their embeddings within a similarity space (euclidean, cosine, etc.).

Temporal Smoothness. When talking about dynamic networks and communities, temporal smoothness should also be considered. Between subsequent timesteps, the dynamic networks often evolve, but not by a large amount. While individual nodes may change drastically within a single timestep, the communities are seen as more stable structures within the networks. Therefore, the evolution of the communities should not exceed to global (network) evolution rate.

In most literature, this temporal smoothness is indirectly handled by result matching or reuse of results from previous timesteps. Within representation-based approaches, this property can be quantified and optimized for using temporal-stability measures. A similar approach is employed to keep the embedding space temporally stable, while only individual nodes may change.

5.3 Community Detection

In the final step of the framework, dynamic communities need to be identified. This may be done by simultaneously training community embeddings along with the node embeddings [54, 57, 92], therefore having the advantage that objective function can directly influence the resulting

communities. Other approaches instead operate on the resulting embedding space or the augmented graphs to extract the resulting communities using link-based methods such as the Louvain method or density-based clustering algorithms such as K-means, BIRCH [105], or OPTICS [7] yielding the benefit of losing the community count assumption.

In our approach, we plan to focus on direct community optimization, while avoiding hard-coding the model to specific assumptions using spectral clustering-based techniques and soft assignment clustering [53, 57].

5.4 Benchmarks and Baselines

As the research questions posed in Section 4 are all mostly of a quantitative nature, it is very important to set up appropriate benchmarks to provide a significant answer. As a direct comparison between methods is not always possible, we define auxiliary task benchmarks for testing algorithms on desired properties as well as reuse benchmarks used in previous literature to provide a fair comparison.

5.4.1 Benchmarks and Evaluation

To provide an answer for Research Question 1, the quality of the algorithm on both static and dynamic communities needs to be compared against the benchmarks for various configurations (considering the content and/or meta-topological data). As our baselines include both representation- as well as link-based approaches, the benchmarks should cover measures used in both groups. To evaluate the quality of the communities, annotation-based approaches (computing NMI and NF1) and quality metric-based evaluation approaches will be employed (See Section 3.5). Since our definition of community slightly differs from the literature as it encompasses network external information (content) we will also employ task-based evaluation such as recommendation tasks (follower recommendation, hashtag recommendation – depending on the dataset).

The Research Question 3 is of a more exploratory nature concerning the modeling of temporal information. It aims to determine whether having the ability to track the communities through time yields better results in practice, as opposed to having communities incorporate their temporal information implicitly in their definition. This evaluation is conducted as an ablation test and similarly focuses on quality measures and task-based evaluation.

The Research Question 2 aims to compare the scalability of our approach to the current representation-based approaches. Therefore, a rough complexity analysis, as well as performance benchmarking (computation time), shall be conducted.

Finally, Research Question 4 addresses the usability of our dynamic community detection results to other tasks concerning dynamic node representation learning. Here, we, make use of defined auxiliary tasks (recommendation and link-prediction) to compare our method against other dynamic node representation learning algorithms.

For benchmarking, the datasets of different scales and properties are chosen (Table 2). The synthetic dataset generation method introduced in Greene et al. [30] will be used to create additional networks with ground-truth communities.

Table 2. Overview of the datasets used for evaluation. All the datasets will be used for the quality measure-based as well as auxiliary task-based evaluation. The "Annotated" column indicates whether a dataset is eligible for annotation-based evaluation, i.e., it contains ground-truth communities.

Dataset	Nodes	Edges	Temporal	Annotated
Zachary karate club	34	78	N	Y
Football	115	613	N	N
Star Wars Social	113	1599	Y	N
Enron	605K	4.1M	Y	Y
IMDB 5000	16K	52K	Y	N
DBLP-HCN	11K	16K	Y	Y
DBLP-V1	1.2M	2.4M	Y	Y
DBLP-V3	2.7M	8.2M	Y	Y
sx-mathoverflow	24K	506K	Y	N
sx-superuser	194K	1.4M	Y	N
Eu-core network	1005	25K	N	Y
com-Youtube	1.1M	298K	N	Y
116th House of Representatives	6249	12K	N	N
social-distancing-student	93K	3.7M	Y	N

5.4.2   Baselines

To give a fair representation of the state-of-the-art, the following methods are selected as baselines. The selection is based on the category of communities they learn, diversification of techniques, and competitiveness with the ideas introduced as part of our framework.

Static Community Detection. For Research Question 1, we will evaluate our algorithm against baselines in Table 3. We use both static as well as dynamic algorithms as baselines to identify the benefit of dynamic community detection over static communities.

Table 3. List of community detection methods we will use as baselines. The "dynamic" column indicates whether the algorithm is capable of detecting dynamic communities, and the "method" column indicates whether it is a link-based or representation-based algorithm.

Reference	Dynamic	Method
He and Chen [37]	N	Link-based
Blondel et al. [10]	N	Link-based
Rozemberczki et al. [75]	N	Representation-based
Cavallari et al. [14]	N	Representation-based
Jia et al. [40]	N	Representation-based
Rossetti [72]	Y	Link-based
Wang et al. [87]	Y	Link-based
Greene et al. [30]	Y	Link-based
Wang et al. [92]	Y	Representation-based
Ma et al. [57]	Y	Representation-based

Dynamic Representation Learning. To answer Research Question 3, we will evaluate the algorithm against other dynamic representation learning algorithms to verify that learned node embeddings are still usable for node-level predictions tasks as well as community-level tasks.

- Nguyen et al. [65]
- Wu et al. [95]
- Pareja et al. [67]

5.5 Extensions

If the timing of the research project permits, while not the main focus of the research, additional extensions of the algorithms for future work may be explored. These extensions may encompass exploring evolutional event detection within the extracted dynamic communities.

6 PLANNING

In this section, the approximate planning for my thesis is discussed. The research phase of the thesis is broken down into two-week blocks. Below a timeline of these sprints can be found along with goals expected to achieve by the end of the block. The final deadline for the thesis is June 13th, 2022.

6.1 Timeline

6.1.1 January 24 - 2 weeks

Preparing the datasets, and setting up benchmarks for the baseline methods. Implementation of baseline if the implementation is not publicly available.

6.1.2 February 5 - 2 weeks

Finishing up the benchmark implementation. Conducting the evaluation and collection of the results on the baselines for the RQ's.

6.1.3 February 19 - 2 weeks

Setting up graph sampling pipeline and experimentation with various representation-based approaches. Collecting performance results for ablation tests.

6.1.4 March 3 - 2 weeks

Experimenting and implementing incorporation of community-based objectives into the framework. Conducting experiments with temporal modeling for Research Question 3.

6.1.5 March 19 - 2 weeks

Finishing up experimenting and implementing incorporation of community-based objectives into the framework. Experimenting with community extraction and sub-sampling.

6.1.6 April 2 - 2 weeks

Evaluation and tuning of the algorithm parameters to collect results for Research Question 1 and Research Question 4

6.1.7 April 10 - 2 weeks

Evaluation and optimization of the algorithm for scalability. Collecting data for Research Question 2.

6.1.8   April 30 - 2 weeks

Writing thesis: describing the full methodology

6.1.9   May 14 - 2 weeks

Writing thesis: describing evaluation and summarizing the results

6.1.10   May 28 - 2 weeks

Completing the thesis and ensuring everything is ready for delivery.

REFERENCES

[1] The 116th U.S. House of Representatives. https://kaggle.com/aavigan/house-of-representatives-congress-116.

[2] Home — The 13th International Conference on Web Information Systems Engineering. http://www.wise2012.cs.ucy.ac.cy/challenge.html.

[3] IMDB 5000 Movie Dataset. https://kaggle.com/carolzhangdc/imdb-5000-movie-dataset.

[4] SNAP: Network datasets: Math Overflow temporal network. https://snap.stanford.edu/data/sx-mathoverflow.html.

[5] Star Wars Social Network. https://kaggle.com/ruchi798/star-wars.

[6] Sami Abu-El-Haija, Bryan Perozzi, Rami Al-Rfou, and Alex Alemi. Watch your step: Learning node embeddings via graph attention. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 9198–9208, Red Hook, NY, USA, December 2018. Curran Associates Inc.

[7] Mihael Ankerst, Markus M. Breunig, Hans-peter Kriegel, and Jörg Sander. OPTICS: Ordering Points To Identify the Clustering Structure. pages 49–60. ACM Press, 1999.

[8] Sitaram Asur, Srinivasan Parthasarathy, and Duygu Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. *ACM Transactions on Knowledge Discovery from Data*, 3(4):16:1–16:36, December 2009. ISSN 1556-4681. doi: 10.1145/1631162.1631164.

[9] Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. The Pushshift Reddit Dataset. *arXiv:2001.08435 [cs]*, January 2020.

[10] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, October 2008. ISSN 1742-5468. doi: 10.1088/1742-5468/2008/10/P10008.

[11] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[12] Julia Brennecke and Olaf Rank. The firm's knowledge network and the transfer of advice among corporate inventors—A multilevel network study. *Research Policy*, 46(4):768–783, May 2017. ISSN 0048-7333. doi: 10.1016/j.respol.2017.02.002.

[13] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric Deep Learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, July 2017. ISSN 1558-0792. doi: 10.1109/MSP.2017.2693418.

[14] Sandro Cavallari, Vincent W. Zheng, Hongyun Cai, Kevin Chen-Chuan Chang, and Erik Cambria. Learning Community Embedding with Community Detection and Node Embedding on Graphs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 377–386, Singapore Singapore, November 2017. ACM. ISBN 978-1-4503-4918-5. doi: 10.1145/3132847.3132925.

[15] Rémy Cazabet, Hideaki Takeda, Masahiro Hamasaki, and F. Amblard. Using dynamic community detection to identify trends in user-generated content. *Social Network Analysis and Mining*, 2012. doi: 10.1007/s13278-012-0074-8.

[16] Òscar Celma. Music Recommendation and Discovery in the Long Tail. *undefined*, 2008.

[17] Weijian Chen, Fuli Feng, Qifan Wang, Xiangnan He, Chonggang Song, Guohui Ling, and Yongdong Zhang. CatGCN: Graph Convolutional Networks with Categorical Node Features. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2021. ISSN 1041-4347, 1558-2191, 2326-3865. doi: 10.1109/TKDE.2021.3133013.

[18] Michele Coscia, Fosca Giannotti, and Dino Pedreschi. A classification for community discovery methods in complex networks. *Statistical Analysis and Data Mining*, 4(5):512–546, October 2011. ISSN 1932-1864. doi: 10.1002/sam.10133.

[19] Narimene Dakiche, Fatima Benbouzid-Si Tayeb, Yahya Slimani, and Karima Benatchba. Tracking community evolution in social networks: A survey. *Information Processing & Management*, 56(3):1084–1102, May 2019. ISSN 0306-4573. doi: 10.1016/j.ipm.2018.03.005.

[20] Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. HyTE: Hyperplane-based Temporally aware Knowledge Graph Embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language*

Processing, pages 2001–2011, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1225.

[21] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*, May 2019.

[22] Chris Diehl, Galileo Namata, and Lise Getoor. Relationship Identification for Social Network Discovery. pages 546–552, January 2007.

[23] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. Metapath2vec: Scalable Representation Learning for Heterogeneous Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 135–144, New York, NY, USA, August 2017. Association for Computing Machinery. ISBN 978-1-4503-4887-4. doi: 10.1145/3097983.3098036.

[24] Hossein Fani, Eric Jiang, Ebrahim Bagheri, Feras Al-Obeidat, Weichang Du, and Mehdi Kargar. User community detection via embedding of social network structure and temporal content. *Information Processing & Management*, 57 (2):102056, March 2020. ISSN 03064573. doi: 10.1016/j.ipm.2019.102056.

[25] Elahe Ghalebi, Baharan Mirzasoleiman, Radu Grosu, and Jure Leskovec. Dynamic network model from partial observations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 9884–9894, Red Hook, NY, USA, December 2018. Curran Associates Inc.

[26] Amir Ghasemian, Pan Zhang, Aaron Clauset, Cristopher Moore, and Leto Peel. Detectability Thresholds and Optimal Algorithms for Community Structure in Dynamic Networks. *Physical Review X*, 6(3):031005, July 2016. doi: 10.1103/PhysRevX.6.031005.

[27] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, June 2002. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.122653799.

[28] Martin Grandjean. A social network analysis of Twitter: Mapping the digital humanities community. *Cogent Arts & Humanities*, 3(1):1171458, December 2016. ISSN null. doi: 10.1080/23311983.2016.1171458.

[29] Clara Granell, Richard K. Darst, Alex Arenas, Santo Fortunato, and Sergio Gómez. Benchmark model to assess community structure in evolving networks. *Physical Review E*, 92(1):012805, July 2015. doi: 10.1103/PhysRevE.92.012805.

[30] Derek Greene, Dónal Doyle, and Pádraig Cunningham. Tracking the Evolution of Communities in Dynamic Social Networks. In *2010 International Conference on Advances in Social Networks Analysis and Mining*, pages 176–183, August 2010. doi: 10.1109/ASONAM.2010.17.

[31] Aditya Grover and Jure Leskovec. Node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 855–864, New York, NY, USA, August 2016. Association for Computing Machinery. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939754.

[32] Tom Gruber. Collective knowledge systems: Where the Social Web meets the Semantic Web. *Journal of Web Semantics*, 6(1):4–13, February 2008. ISSN 1570-8268. doi: 10.1016/j.websem.2007.11.011.

[33] Qian Guo, Lei Zhang, Bin Wu, and Xuelin Zeng. Dynamic community detection based on distance dynamics. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 329–336, August 2016. doi: 10.1109/ASONAM.2016.7752254.

[34] Loni Hagen, Thomas Keller, Stephen Neely, Nic DePaula, and Claudia Robert-Cooperman. Crisis Communications in the Age of Social Media: A Network Analysis of Zika-Related Tweets. *Social Science Computer Review*, 36(5):523–541, October 2018. ISSN 0894-4393. doi: 10.1177/0894439317721985.

[35] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 1025–1035, Red Hook, NY, USA, December 2017. Curran Associates Inc. ISBN 978-1-5108-6096-4.

[36] William L. Hamilton, Rex Ying, and Jure Leskovec. Representation Learning on Graphs: Methods and Applications. *arXiv:1709.05584 [cs]*, April 2018.

[37] Jialin He and Duanbing Chen. A fast algorithm for community detection in temporal network. *Physica A: Statistical Mechanics and its Applications*, 429, July 2015. doi: 10.1016/j.physa.2015.02.069.

[38] Tad Hogg and Kristina Lerman. Social dynamics of Digg. *EPJ Data Science*, 1(1):1–26, December 2012. ISSN 2193-1127. doi: 10.1140/epjds5.

[39] Mingqing Huang, Qingshan Jiang, Qiang Qu, Lifei Chen, and Hui Chen. Information fusion oriented heterogeneous social network for friend recommendation via community detection. *Applied Soft Computing*, 114:108103, January 2022. ISSN 1568-4946. doi: 10.1016/j.asoc.2021.108103.

[40] Yuting Jia, Qinqin Zhang, Weinan Zhang, and Xinbing Wang. CommunityGAN: Community Detection with Generative Adversarial Nets. In *The World Wide Web Conference*, pages 784–794, San Francisco CA USA, May 2019. ACM. ISBN 978-1-4503-6674-8. doi: 10.1145/3308558.3313564.

[41] Yoonsuk Kang, Jun-Seok Lee, Won-Yong Shin, and Sang-Wook Kim. Community reinforcement: An effective and efficient preprocessing method for accurate community detection. *Knowledge-Based Systems*, page 107741, November

2021. ISSN 0950-7051. doi: 10.1016/j.knosys.2021.107741.

[42] Arzum Karataş and Serap Şahin. A Review on Social Bot Detection Techniques and Research Directions. October 2017.

[43] Thomas N. Kipf and Max Welling. Variational Graph Auto-Encoders. *arXiv:1611.07308 [cs, stat]*, November 2016.

[44] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv:1609.02907 [cs, stat]*, February 2017.

[45] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv:1609.02907 [cs, stat]*, February 2017.

[46] Xiang Kong. Why are social network transactions important? Evidence based on the concentration of key suppliers and customers in China. *China Journal of Accounting Research*, 4(3):121–133, September 2011. ISSN 1755-3091. doi: 10.1016/j.cjar.2011.06.003.

[47] Srijan Kumar, Francesca Spezzano, V. S. Subrahmanian, and Christos Faloutsos. Edge Weight Prediction in Weighted Signed Networks. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 221–230, December 2016. doi: 10.1109/ICDM.2016.0033.

[48] Sejeong Kwon, Meeyoung Cha, and Kyomin Jung. Rumor Detection over Varying Time Windows. *PLOS ONE*, 12(1): e0168344, January 2017. ISSN 1932-6203. doi: 10.1371/journal.pone.0168344.

[49] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 80(1):016118, July 2009. doi: 10.1103/PhysRevE.80.016118.

[50] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4):046110, October 2008. doi: 10.1103/PhysRevE.78.046110.

[51] Andrea Lancichinetti, Santo Fortunato, and János Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015, March 2009. ISSN 1367-2630. doi: 10.1088/1367-2630/11/3/033015.

[52] Henry Li, Ofir Lindenbaum, Xiuyuan Cheng, and Alexander Cloninger. Variational Diffusion Autoencoders with Random Walk Sampling. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 362–378, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58592-1. doi: 10.1007/978-3-030-58592-1_22.

[53] Hongmin Li, Xiucai Ye, Akira Imakura, and Tetsuya Sakurai. *Divide-and-Conquer Based Large-Scale Spectral Clustering*. May 2021. doi: 10.13140/RG.2.2.15207.37281.

[54] Sungsu Lim, Junghoon Kim, and Jae-Gil Lee. BlackHole: Robust community detection inspired by graph drawing. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, pages 25–36, May 2016. doi: 10.1109/ICDE.2016.7498226.

[55] Xin Liu, Weichu Liu, Tsuyoshi Murata, and Ken Wakita. Community Detection in Multi-Partite Multi-Relational Networks Based on Information Compression. *New Generation Computing*, 34(1):153–176, March 2016. ISSN 1882-7055. doi: 10.1007/s00354-016-0206-1.

[56] Linhao Luo, Yixiang Fang, Xin Cao, Xiaofeng Zhang, and Wenjie Zhang. Detecting Communities from Heterogeneous Graphs: A Context Path-based Graph Neural Network Model. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 1170–1180. Association for Computing Machinery, New York, NY, USA, October 2021. ISBN 978-1-4503-8446-9.

[57] Lijia Ma, Yutao Zhang, Jianqiang Li, Qiuzhen Lin, Qing Bao, Shanfeng Wang, and Maoguo Gong. Community-aware dynamic network embedding by using deep autoencoder. *Information Sciences*, 519:22–42, May 2020. ISSN 0020-0255. doi: 10.1016/j.ins.2020.01.027.

[58] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781 [cs]*, September 2013.

[59] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. *arXiv:1310.4546 [cs, stat]*, October 2013.

[60] Martina Morris, editor. *Network Epidemiology: A Handbook for Survey Design and Data Collection*. International Studies in Demography. Oxford University Press, Oxford, 2004. ISBN 978-0-19-926901-3. doi: 10.1093/0199269017.001.0001.

[61] Mohammad Mosadegh and Mehdi Behboudi. Using Social Network Paradigm for Developing a Conceptual Framework in CRM. *Australian Journal of Business and Management Research*, 1:63–71, August 2011. doi: 10.52283/NSWRCA.AJBMR.20110104A06.

[62] Nairouz Mrabah, Mohamed Bouguessa, Mohamed Fawzi Touati, and Riadh Ksantini. Rethinking Graph Auto-Encoder Models for Attributed Graph Clustering. *arXiv:2107.08562 [cs]*, July 2021.

[63] Peter J. Mucha, Thomas Richardson, Kevin Macon, Mason A. Porter, and Jukka-Pekka Onnela. Community Structure in Time-Dependent, Multiscale, and Multiplex Networks. November 2009. doi: 10.1126/science.1184819.

[64] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics*, 69(6 Pt 2):066133, June 2004. ISSN 1539-3755. doi: 10.1103/PhysRevE.69.066133.

[65] Giang Hoang Nguyen, John Boaz Lee, Ryan A. Rossi, Nesreen K. Ahmed, Eunyee Koh, and Sungchul Kim. Continuous-Time Dynamic Network Embeddings. In *Companion of the The Web Conference 2018 on The Web Conference 2018 - WWW '18*, pages 969–976, Lyon, France, 2018. ACM Press. ISBN 978-1-4503-5640-4. doi: 10.1145/3184558.3191526.

[66] Gergely Palla, Albert-László Barabási, and Tamás Vicsek. Quantifying social group evolution. *Nature*, 446(7136): 664–667, April 2007. ISSN 1476-4687. doi: 10.1038/nature05670.

[67] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:5363–5370, April 2020. doi: 10.1609/aaai.v34i04.5984.

[68] Leto Peel, Daniel B. Larremore, and Aaron Clauset. The ground truth about metadata and community detection in networks. *Science Advances*, 3(5):e1602548, 2017. doi: 10.1126/sciadv.1602548.

[69] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162.

[70] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. DeepWalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 701–710, New York, NY, USA, August 2014. Association for Computing Machinery. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623732.

[71] Stephen D. Pryke. Analysing construction project coalitions: Exploring the application of social network analysis. *Construction Management and Economics*, 22(8):787–797, October 2004. ISSN 0144-6193. doi: 10.1080/0144619042000206533.

[72] Giulio Rossetti. ANGEL: Efficient, and effective, node-centric community discovery in static and dynamic networks. *Applied Network Science*, 5(1):26, June 2020. ISSN 2364-8228. doi: 10.1007/s41109-020-00270-6.

[73] Giulio Rossetti and Rémy Cazabet. Community Discovery in Dynamic Networks: A Survey. *ACM Computing Surveys*, 51(2):35:1–35:37, February 2018. ISSN 0360-0300. doi: 10.1145/3172867.

[74] Giulio Rossetti, Luca Pappalardo, and Salvatore Rinzivillo. A Novel Approach to Evaluate Community Detection Algorithms on Ground Truth. In Hocine Cherifi, Bruno Gonçalves, Ronaldo Menezes, and Roberta Sinatra, editors, *Complex Networks VII: Proceedings of the 7th Workshop on Complex Networks CompleNet 2016*, Studies in Computational Intelligence, pages 133–144. Springer International Publishing, Cham, 2016. ISBN 978-3-319-30569-1. doi: 10.1007/978-3-319-30569-1_10.

[75] Benedek Rozemberczki, Ryan Davies, Rik Sarkar, and Charles Sutton. GEMSEC: Graph embedding with self clustering. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '19, pages 65–72, New York, NY, USA, August 2019. Association for Computing Machinery. ISBN 978-1-4503-6868-1. doi: 10.1145/3341161.3342890.

[76] Marcel Salathé and James H. Jones. Dynamics and Control of Diseases in Networks with Community Structure. *PLOS Computational Biology*, 6(4):e1000736, April 2010. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1000736.

[77] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. Dynamic Graph Representation Learning via Self-Attention Networks. *arXiv:1812.09430 [cs, stat]*, June 2019.

[78] Hamed Sarvari, Ehab Abozinadah, Alex Mbaziira, and Damon Mccoy. Constructing and Analyzing Criminal Networks. In *2014 IEEE Security and Privacy Workshops*, pages 84–91, May 2014. doi: 10.1109/SPW.2014.22.

[79] Erick Stattner and Nicolas Vidot. Social network analysis in epidemiology: Current trends and perspectives. In *2011 FIFTH INTERNATIONAL CONFERENCE ON RESEARCH CHALLENGES IN INFORMATION SCIENCE*, pages 1–11, May 2011. doi: 10.1109/RCIS.2011.6006866.

[80] Xing Su, Shan Xue, Fanzhen Liu, Jia Wu, Jian Yang, Chuan Zhou, Wenbin Hu, Cecile Paris, Surya Nepal, Di Jin, Quan Z. Sheng, and Philip S. Yu. A Comprehensive Survey on Community Detection with Deep Learning. *arXiv:2105.12584 [cs]*, May 2021.

[81] Gayatri Swamynathan, Christo Wilson, Bryce Boe, Kevin Almeroth, and Ben Y. Zhao. Do social networks improve e-commerce? a study on social marketplaces. In *Proceedings of the First Workshop on Online Social Networks*, WOSN '08, pages 1–6, New York, NY, USA, August 2008. Association for Computing Machinery. ISBN 978-1-60558-182-8. doi: 10.1145/1397735.1397737.

[82] Imane Tamimi and Mohamed El Kamili. Literature survey on dynamic community detection and models of social networks. In *2015 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, pages 1–5, October 2015. doi: 10.1109/WINCOM.2015.7381332.

[83] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. LINE: Large-scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, pages 1067–1077, Republic and Canton of Geneva, CHE, May 2015. International World Wide Web Conferences Steering Committee.

ISBN 978-1-4503-3469-3. doi: 10.1145/2736277.2741093.

[84] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. ArnetMiner: Extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 990–998, New York, NY, USA, August 2008. Association for Computing Machinery. ISBN 978-1-60558-193-4. doi: 10.1145/1401890.1402008.

[85] Victoria Vanburen, David Villarreal, Thomas McMillen, and Andrew Minnicks. Enron Dataset Research: E-mail Relevance Classification. *Technical Reports-Computer Science*, January 2009.

[86] Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P. Gummadi. On the evolution of user interaction in Facebook. In *Proceedings of the 2nd ACM Workshop on Online Social Networks*, WOSN '09, pages 37–42, New York, NY, USA, August 2009. Association for Computing Machinery. ISBN 978-1-60558-445-4. doi: 10.1145/1592665.1592675.

[87] Peizhuo Wang, Lin Gao, and Xiaoke Ma. Dynamic community detection based on network structural perturbation and topological similarity. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(1):013401, January 2017. ISSN 1742-5468. doi: 10.1088/1742-5468/2017/1/013401.

[88] Shihan Wang, Marijn Schraagen, Erik Tjong Kim Sang, and Mehdi Dastani. Public Sentiment on Governmental COVID-19 Measures in Dutch Social Media. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online, December 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.nlpcovid19-2.17.

[89] Wei Wang, Feng Xia, Hansong Nie, Zhikui Chen, Zhiguo Gong, Xiangjie Kong, and Wei Wei. Vehicle Trajectory Clustering Based on Dynamic Representation Learning of Internet of Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, PP:1–10, June 2020. doi: 10.1109/TITS.2020.2995856.

[90] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous Graph Attention Network. In *The World Wide Web Conference*, WWW '19, pages 2022–2032, New York, NY, USA, May 2019. Association for Computing Machinery. ISBN 978-1-4503-6674-8. doi: 10.1145/3308558.3313562.

[91] Yi Wang, Bin Wu, and Nan Du. Community Evolution of Social Network: Feature, Algorithm and Model. *arXiv:0804.4356 [physics]*, April 2008.

[92] Zhen Wang, Chunyu Wang, Chao Gao, Xuelong Li, and Xianghua Li. An evolutionary autoencoder for dynamic community detection. *Science China Information Sciences*, 63(11):212205, September 2020. ISSN 1869-1919. doi: 10.1007/s11432-020-2827-9.

[93] Robert West, Hristo S. Paskov, Jure Leskovec, and Christopher Potts. Exploiting Social Network Structure for Person-to-Person Sentiment Analysis. *Transactions of the Association for Computational Linguistics*, 2:297–310, October 2014. ISSN 2307-387X. doi: 10.1162/tacl_a_00184.

[94] Chengyuan Wu and Carol Hargreaves. *Topological Machine Learning for Mixed Numeric and Categorical Data*. March 2020.

[95] Jiaming Wu, Meng Liu, Jiangting Fan, Yong Liu, and Meng Han. SageDy: A Novel Sampling and Aggregating Based Representation Learning Approach for Dynamic Networks. In Igor Farkaš, Paolo Masulli, Sebastian Otte, and Stefan Wermter, editors, *Artificial Neural Networks and Machine Learning – ICANN 2021*, Lecture Notes in Computer Science, pages 3–15, Cham, 2021. Springer International Publishing. ISBN 978-3-030-86383-8. doi: 10.1007/978-3-030-86383-8_1.

[96] Xiaodong Wu, Weizhe Lin, Zhilin Wang, and Elena Rastorgueva. Author2Vec: A Framework for Generating User Embedding. *arXiv:2003.11627 [cs, stat]*, March 2020.

[97] Carl Yang, Mengxiong Liu, Zongyi Wang, Liyuan Liu, and Jiawei Han. Graph Clustering with Dynamic Embedding. *arXiv:1712.08249 [physics]*, December 2017.

[98] Carl Yang, Yuxin Xiao, Yu Zhang, Yizhou Sun, and Jiawei Han. Heterogeneous Network Representation Learning: A Unified Framework with Survey and Benchmark. *IEEE Transactions on Knowledge and Data Engineering*, PP:1–1, December 2020. doi: 10.1109/TKDE.2020.3045924.

[99] Jaewon Yang and Jure Leskovec. Community-Affiliation Graph Model for Overlapping Network Community Detection. In *2012 IEEE 12th International Conference on Data Mining*, pages 1170–1175, December 2012. doi: 10.1109/ICDM.2012.139.

[100] Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth. In *Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*, MDS '12, pages 1–8, New York, NY, USA, August 2012. Association for Computing Machinery. ISBN 978-1-4503-1546-3. doi: 10.1145/2350190.2350193.

[101] Fanghua Ye, Chuan Chen, and Zibin Zheng. Deep Autoencoder-like Nonnegative Matrix Factorization for Community Detection. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, pages 1393–1402, New York, NY, USA, October 2018. Association for Computing Machinery. ISBN 978-1-4503-6014-2. doi: 10.1145/3269206.3271697.

[102] Ying Yin, Yuhai Zhao, He Li, and Xiangjun Dong. Multi-objective evolutionary clustering for large-scale dynamic community detection. *Information Sciences*, 549:269–287, March 2021. ISSN 0020-0255. doi: 10.1016/j.ins.2020.11.025.

[103] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 974–983, New York, NY, USA, July 2018. Association for Computing Machinery. ISBN 978-1-4503-5552-0. doi: 10.1145/3219819.3219890.

[104] Wayne Zachary. An Information Flow Model for Conflict and Fission in Small Groups1. *Journal of anthropological research*, 33, November 1976. doi: 10.1086/jar.33.4.3629752.

[105] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: An efficient data clustering method for very large databases. *ACM SIGMOD Record*, 25(2):103–114, June 1996. ISSN 0163-5808. doi: 10.1145/235968.233324.