Machine Learning (2023) 112:5057–5080
https://doi.org/10.1007/s10994-023-06402-0

UCoDe: unified community detection with graph convolutional networks

Atefeh Moradan[1] · **Andrew Draganov**[1] · **Davide Mottin**[1] · **Ira Assent**[1]

Received: 14 February 2023 / Revised: 7 July 2023 / Accepted: 16 August 2023 /
Published online: 20 September 2023

Abstract

Community detection finds homogeneous groups of nodes in a graph. Existing approaches either partition the graph into disjoint, *non-overlapping*, communities, or determine only *overlapping* communities. To date, no method supports both detections of overlapping and non-overlapping communities. We propose UCoDe, a *unified* method for community detection in attributed graphs that detects both overlapping and non-overlapping communities by means of a novel contrastive loss that captures node similarity on a macro-scale. Our thorough experimental assessment on real data shows that, regardless of the data distribution, our method is either the top performer or among the top performers in both overlapping and non-overlapping detection without burdensome hyper-parameter tuning.

Keywords Community detection · Graph neural networks · Overlapping · Non-overlapping · Modularity

1 Introduction

Community detection (Fortunato, 2010) is the problem of identifying sets of nodes in a graph that share common characteristics. In social networks, community detection identifies groups of individuals who participate in joint activities (e.g. sports clubs) or having similar preferences (Perozzi et al., 2014); in biological networks, communities represent proteins that contribute to a specific disease (Mall et al., 2017). Such

Editors: Fabio Vitale, Tania Cerquitelli, Marcello Restelli, Charalampos Tsourakakis.

✉ Atefeh Moradan
atefeh.moradan@cs.au.dk

Andrew Draganov
draganovandrew@cs.au.dk

Davide Mottin
davide@cs.au.dk

Ira Assent
ira@cs.au.dk

[1] Department of Computer Science, Aarhus University, Aabogade 34, Aarhus 8200, Denmark

networks include information in node attributes that may be helpful when identifying similarities (e.g., the age of a person). However, these attributes are typically not considered by traditional community detection methods, such as spectral clustering (Shi & Malik, 2000), modularity maximization (Newman, 2006), or more recent graph embeddings (Cai et al., 2018), making them ill-suited for detecting node communities in attributed graphs.

In recent years, graph neural networks (GNNs) (Kipf & Welling, 2017; Veličković et al., 2018; Hamilton et al., 2017; Bronstein et al., 2017; Battaglia et al., 2018) have shown superior performance in a number of supervised tasks on graphs, especially link prediction, node classification, and graph classification. GNNs popularity stems from their aptitude to capture complex relationships in networks, typically by means of propagating node attributes and features to neighboring nodes by a message-passing process (Battaglia et al., 2018). These are typically accompanied by *graph pooling* (Bruna et al., 2014; Bianchi et al., 2020; Lee et al., 2019), which aggregates multiple nodes into higher-level representations to reduce the number of parameters of the neural network.

GNNs have propelled advancements in supervised tasks; yet on unsupervised tasks such as community detection, GNNs have not yet received the same attention. Most existing GNN methods do not directly optimize for community detection but achieve the objective indirectly. Unsupervised GNNs, such as the popular Deep Graph Infomax (DGI) (Veličković et al., 2018), find node representations that, in a second step, need to be subjected to a clustering algorithm, such as the widely used k-means, to actually obtain communities.

Recently, a few methods propose GNNs that *explicitly* optimize for community detection. GNNs for *non-overlapping* community detection either optimize for a *single score* or *combine several scores*. Single score methods revisit traditional measures such as min-cut (Bianchi et al., 2020) and modularity (Tsitsulin et al., 2020) objectives to return node-community probabilities. Combined score methods (Zhang et al., 2019, 2020) integrate multiple different objectives. These methods outperform single score methods in non-overlapping community detection, but require substantial tuning to the dataset at hand and are typically less robust and less interpretable than their single objective counterparts.

Non-overlapping community detection aims at returning a single community assignment for each node. As such, they are ill-suited for overlapping community detection. NOCD (Shchur & Günnemann, 2019) is, at the time of writing this paper, the only GNN that optimizes for *overlapping community* detection. In particular, NOCD finds communities that maximize the probability of recovering the graph structure. Yet, this approach constrains the community structure to be overlapping and thus does not capture non-overlapping communities. In conclusion, to date, no GNN detects both overlapping and non-overlapping communities.

Contributions. (1) We introduce a new GNN method, UCoDe, for community detection on graphs. We devise a simple effective single score model which leverages state-of-the-art representations; **(2)** UCoDe features a novel contrastive loss function that promotes both overlapping and non-overlapping communities, thus being the first approach to achieve competitive results across these tasks with a single model. **(3)** We perform extensive experiments on real data, showing that our method outperforms single-objective methods without the need for extensive parameter tuning, achieving quality on par with more complex combined scores.

2 Related work

Before delving into our solution, we provide an overview of the literature on community detection, graph neural networks, graph pooling, and graph embeddings. Table 1 provides a summary of the characteristics of the most important work in the area, highlighting core properties of the methods, such as their ability to capture overlapping and non-overlapping communities and whether they achieve their results in an unsupervised manner with a single score approach.

2.1 Traditional community detection

Community detection has a long history in graph analysis (Fortunato, 2010) with applications across the natural sciences. There are two main categories of community detection: *non-overlapping* community detection, also called partitioning, which seeks an assignment of each node to exactly one community; *overlapping* community detection, seeking a soft-assignment of nodes into potentially multiple communities. A community detection algorithm optimizes a score that describes the cohesiveness of nodes in the community with respect to the rest of the nodes. A number of scores and methods have been proposed based on the graph structure, such as spectral clustering for min-cut (Shi & Malik, 2000), Louvain's method for modularity (Newman, 2006), and the Girvan-Newman algorithm for betweenness (Girvan & Newman, 2002). Other works extend such methods by incorporating node features into the graph analysis (Yang et al., 2013).

Overlapping community detection is often approached using algorithms similar to the Expectation-Maximization algorithm for soft-clustering (Dempster et al., 1977) where each point is a distribution over the clusters. Similarly, AGM (Yang & Leskovec, 2012) and BigCLAM (Yang and Leskovec, 2013) formulate the community detection problem as finding soft assignments to communities that maximize the model likelihood. Other traditional methods find overlapping communities by removing high-betweenness edges (Gregory, 2007) or by propagating label information (Gregory, 2010). Lastly, EPM (Zhou, 2015) fits a Bernoulli-Poisson model, SNMF (Wang et al., 2010) and CDE (Li et al., 2018) use non-negative matrix factorization.

2.2 Graph neural networks for community detection

GNNs (Wu et al., 2020) are a family of parametric models that learn node representations by aggregating features over the graph's structure. GNNs exhibit state-of-the-art performance in supervised tasks, such as link prediction, node and graph classification.

Popular GNN models include spectral GNNs (Defferrard et al., 2016; Bronstein et al., 2017), GCNs (Hamilton et al., 2017; Kipf & Welling, 2017), graph autoencoders (GAEs) (Kipf and Welling, 2016), graph isomorphism networks (Xu et al., 2018), and Deep Graph Infomax (DGI) (Veličković et al., 2018). These models compute node features in an unsupervised manner if equipped with a reconstruction loss. A clustering algorithm, such as k -means, can cluster the node features to return communities. Since there is no coupling between such GNN model objectives and the clustering algorithm, the resulting communities may not accurately represent all groups in the graph.

Table 1 Related work in terms of present (✓) and absent (✗) properties

Method	Lloss function				Model		
	Unsuperv	Modularity	Contrastive	Single score	Non-overlap	Overlap	Trainable
DGI (Veličković et al., 2018)	✓	✗	✓	✓	✓	✗	✓
Graclus (Dhillon et al., 2007)	✗	✗	✗	✓	✓	✗	✗
DiffPool (Ying et al., 2018)	✓	✗	✗	✓	✓	✗	✗
MinCut (Bianchi et al., 2020)	✓	✗	✗	✓	✓	✗	✓
CommDGI (Zhang et al., 2020)	✓	✓	✗	✗	✓	✗	✓
DMoN (Tsitsulin et al., 2020)	✓	✓	✗	✓	✓	✗	✓
AGC (Zhang et al., 2019)	✓	✗	✗	✓	✓	✗	✓
NOCD (Shchur & Günnemann, 2019)	✓	✗	✗	✓	✗	✓	✓
DCRN (Liu et al., 2022)	✓	✗	✗	✗	✓	✗	✓
UCoDe	✓	✓	✓	✓	✓	✓	✓

GNNs for community detection

Some GNNs directly optimize for non-overlapping community detection with community-wise loss functions. *Single objective* approaches propose variations of traditional cohesiveness scores, such as min-cut (Bianchi et al., 2020) and modularity (Tsitsulin et al., 2020). Yet, single-objective methods inherit the limitations of the score they aim to optimize, providing community memberships that are subject to the loss objective's definition of community.

CommDGI (Zhang et al., 2020) proposes a *combined objective* as a linear combination of three objectives, the DGI objective (Veličković et al., 2018), modularity, and mutual information. CommDGI's combined objective overcomes the limitations of the single score methods but requires extensive parameter tuning for proper results. Similarly, recent multi-objective methods operate on the pairwise correlation matrix (Liu et al., 2022), unsupervised contrastive relations (Park et al., 2022), KL-divergence between clusters (Zhao et al., 2021) (Bo et al., 2020), and structured encodings (He et al., 2021). These methods, besides employing complex combined objectives, often require initialization with elaborate pre-trained models (Bo et al., 2020; Zhao et al., 2021; Liu et al., 2022), running *k*-means either in the computation of the embeddings (Liu et al., 2022), in each epoch (Sun et al., 2021), or as an initialization step (Bo et al., 2020), and hyperparameter tuning *for each dataset* (Bo et al., 2020; Zhao et al., 2021; Liu et al., 2022; Park et al., 2022). In contrast, our model uses *the same hyperparameters* for all datasets, devises a single-objective contrastive loss, requires no sophisticated initialization, and detects communities without the need to run *k*-means. Nevertheless, in our evaluation, we also compare with DCRN (Liu et al., 2022), the most recent of such combined objective methods.

While models like DMoN (Tsitsulin et al., 2020) return soft community assignments through a softmax output layer, both single and combined objective methods explicitly penalize overlap among communities.

NOCD (Shchur & Günnemann, 2019) proposes an overlapping community detection loss that maximizes the likelihood of Bernoulli-Poisson models (Shchur & Günnemann, 2019). NOCD achieves competitive results on overlapping community detection but cannot directly detect non-overlapping communities.

2.3 Graph pooling

Graph pooling (Bruna et al., 2014; Bianchi et al., 2020; Lee et al., 2019) is an operation that aggregates nodes so as to learn summarized representations. The purpose of graph pooling is to remove redundant information and reduce the number of parameters of the GNN.

Model-free pooling coarsens the graph structure by aggregating nodes without considering the node attributes. Graclus (Dhillon et al., 2007) revisits max-pooling to aggregate similar nodes in a hierarchical fashion. SAGPool (Lee et al., 2019) proposes a self-attention layer to reweigh nodes in the graph. Model-free approaches act as layers in the network and do not provide communities as output.

Model-based pooling learns coarsening operators through a differentiable loss function. DiffPool (Ying et al., 2018) learns a hierarchical clustering assignment of the graph for supervised graph classification. Top-K pooling (Gao & Ji, 2019) trains an autoencoder that assigns a score to each node; the pooling phase retains the k nodes with the highest score. Yet, these methods do not explicitly optimize for cluster assignments resulting in substandard communities (Bianchi et al., 2020).

MinCutPool (Bianchi et al., 2020), although a pooling technique, returns community assignments by optimizing the min-cut objective of spectral clustering (Shi & Malik, 2000). MinCutPool does not require eigendecomposition of the Laplacian matrix and instead propagates node attributes over the GNN.

2.4 Node embedding methods

Node embeddings (Cai et al., 2018; Chami et al., 2020) learn node representations of the graph structure in an unsupervised manner with shallow neural networks (Perozzi et al., 2014; Tang et al., 2015), autoencoders (Wang et al., 2016), or matrix factorization (Ou et al., 2016; Qiu et al., 2018). Similar to GNN-based representations, a clustering algorithm on the embeddings can be used to detect communities from these representations. Node embeddings can be seen as a generalization of dimensionality reduction methods, and tend to preserve the structure, but disregard node attributes.

A few recent works address the problem of attributed node embeddings through matrix factorization (Yang et al., 2015) or deep models (Gao & Huang, 2018). None of these models are designed for community detection. AGC (Zhang et al., 2019) proposes a combined score based on spectral clustering on top of a GNN representation.

3 Communities and modularity

Consider an *attributed graph* $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$ where $\mathcal{V} = \{v_1, .., v_n\}$ is a set of n nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of edges and $\mathbf{A} = \{a_1, ..., a_l\}$ is set of l attributes. Each node v_i has an associated vector $\mathbf{x}_i \in \mathbb{R}^l$ of real features for each attribute. The node features form an $n \times l$ matrix $\mathbf{X} \in \mathbb{R}^{n \times l}$ where each node-feature vector $\mathbf{x}_i$ is a row in such matrix. The *adjacency matrix* is a matrix representation $\mathbf{A}$ of the graph's structure, where $\mathbf{A}_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$, and 0 otherwise. The *degree* d_i of a node v_i is the number of neighbors of node i, i.e., $d_i = \sum_{j=0}^{n} \mathbf{A}_{ij}$; $\mathbf{d}$ is the vector containing the degree $\mathbf{d}_i = d_i$ of all nodes, and $\mathbf{D}$ is the diagonal degree matrix.

Problem (Attributed graph community detection.) We aim to assign each node to at least one of k communities, $\mathcal{C}_1, ..., \mathcal{C}_k$, such that a score of *community cohesiveness* is maximized. The cluster assignment is a probability vector $\mathbf{c}_i$ indicating the probability of node v_i belonging to community $\mathcal{C}_j$. Cluster assignments form a matrix $\mathbf{C} \in [0, 1]^{n \times k}$ where row i contains node i's cluster assignment $\mathbf{c}_i$.

One of the determinant choices for community detection algorithms is the definition of the community cohesiveness score that determines the quality of the cluster assignments. We now review modularity (Newman, 2006), a popular measure for community detection.

3.1 Modularity

Modularity (Newman, 2006) $Q(\mathcal{G};\mathcal{C})$ measures the quality of a partition $\mathcal{C}$ of the nodes of the graph $\mathcal{G}$; a high modularity score indicates that node grouped by $\mathcal{C}$ have dense internal connections and sparse connections to outside nodes. More specifically, modularity captures the difference in density between the edges inside a community $\mathbf{c}_i$ and the edges of a fixed null model:

$$Q(\mathcal{G};\mathcal{C}) = \frac{1}{4\,|\,\mathcal{E}\,|}\sum_{s=1}^{k}\sum_{ij}\left(\mathbf{A}_{ij} - \frac{d_i d_j}{2\,|\,\mathcal{E}\,|}\right)\mathbf{C}_{is}\mathbf{C}_{js}. \tag{1}$$

The quantity $\frac{d_i d_j}{2|\mathcal{E}|}$ is the null-model representing the probability that two nodes v_i, v_j are connected by chance. The null model in the modularity score is the rewiring model, in which each node v_i preserves its degree d_i but connects randomly to any other node in the graph. By defining the *modularity matrix* $\mathbf{B}$ as $\mathbf{B}_{ij} = \mathbf{A}_{ij} - \frac{d_i d_j}{2|\mathcal{E}|}$ Eq. 1 simplifies into

$$Q(\mathcal{G};\mathcal{C}) = \frac{1}{4\,|\,\mathcal{E}\,|}\,\mathrm{Tr}\,(\mathbf{C}^{\top}\mathbf{B}\mathbf{C}) \tag{2}$$

3.2 Limits and pitfalls

Modularity maximization is one of the most popular methods for community detection (Fortunato, 2010). However, its *direct* maximization may fail to provide optimal communities. As shown in (Fortunato & Barthelemy, 2007), modularity may fail to recognize communities that fall below a graph-specific size. Furthermore, modularity is a measure for discrete partitioning and does not perform well in the case of overlapping communities (Devi & Poovammal, 2016). In the following section, we show how to overcome these limitations of modularity by combining the expressiveness of Graph Neural Networks with a novel contrastive modularity loss that captures both overlapping and non-overlapping communities.

4 Our solution: UCoDe

The modularity objective in Eq. 2 is **NP**-hard, but can be solved efficiently with a spectral approach similar to spectral clustering (Newman, 2006) if we allow matrix $\mathbf{C}$ to be real rather than binary. This relaxed objective admits as solutions the k leading eigenvalues of the matrix $\mathbf{B}$. This convenient relaxation enables soft clustering assignments and, in principle, overlapping community detection.

To circumvent the modularity's resolution limit and capture interactions among nodes that are not directly connected, we further assume that $\mathbf{C}$ is the output of a Graph Neural Network model.

4.1 Graph neural network approach

Graph Neural Networks (GNNs) (Kipf & Welling, 2017; Veličković et al., 2018; Hamilton et al., 2017; Bronstein et al., 2017; Battaglia et al., 2018) transform the node attributes by nonlinear aggregation of attributes of each node's neighbors. By virtue of this aggregation mechanism, these networks are called message passing (Battaglia et al., 2018). We now review the Graph Convolutional Network (GCN) model (Kipf & Welling, 2017). We denote as $\mathbf{X}^{[0]}$ the initial node attributes $\mathbf{X}^{[0]} = \mathbf{X}$,

$$\hat{\mathbf{A}} = \mathbf{D}^{-1/2}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-1/2}$$

the normalized adjacency matrix with self-loops, and $\mathbf{W}^{[t]}$ the weight matrix at layer t, which encodes the parameters of the network. The t+1 layer $\mathbf{X}^{[t+1]}$ is

$$\mathbf{X}^{[t+1]} = \sigma(\hat{\mathbf{A}}\mathbf{X}^{[t]}\mathbf{W}^{[t]})$$

The function σ is a non-linear activation function, such as softmax, SeLU, or ReLU. The matrix $\mathbf{W}^{[0]}$ is randomly initialized, typically as $\mathbf{W}^{[0]} \sim \mathcal{N}(0, 1)$. The parameters $\mathbf{W}$ are learned via stochastic gradient descent on a supervised or unsupervised loss function. The result of a GNN in the last layer T is a matrix $\mathbf{X}^{[T]}$ which rows are embeddings of a node in a d-dimensional space.

To train a GNN, we need to specify a differentiable loss function. For instance, in the node classification task, the loss function is typically the binary cross-entropy. An optimizer, such as ADAM (Kingma & Ba, 2014), finds the parameters $\mathbf{W}^{[1]}, ..., \mathbf{W}^{[T]}$ that minimize the loss function.

The choice of the architecture and the loss function are determinant choices for GNNs. In what follows, we present our model UCoDe that integrates the simplicity of single-objective community detection with the power of combined scores, by virtue of a new loss function that encourages robust community memberships while maintaining consistent separation between dissimilar nodes.

4.2 UCoDe loss function

We build our loss function based on community modularity (Eq. 2). We start by showing that the entire matrix $\mathbf{C}^\top\mathbf{B}\mathbf{C}$ can be interpreted as the modularity *across* communities. Afterward, we introduce our contrastive loss and show how such a loss aims to detect overlapping and non-overlapping communities alike.

4.2.1 $\mathbf{C}^\top\mathbf{B}\mathbf{C}$ as modularity across communities

We observe that $\mathbf{C}^\top\mathbf{B}\mathbf{C}$ encodes the modularity matrix at the community scale

$$\mathbf{C}^\top\mathbf{B}\mathbf{C} = \mathbf{C}^\top\left(\mathbf{A} - \frac{\mathbf{d}\mathbf{d}^\top}{2\mid\mathcal{E}\mid}\right)\mathbf{C} = \underbrace{\mathbf{C}^\top\mathbf{A}\mathbf{C}}_{\mathbf{A}^{\mathcal{C}}} - \frac{1}{2\mid\mathcal{E}\mid}\underbrace{(\mathbf{C}^\top\mathbf{d})(\mathbf{d}^\top\mathbf{C})}_{\mathbf{D}_{\mathcal{C}}\mathbf{D}_{\mathcal{C}}^\top} = Q_M$$

We refer to Q_M as the community-wise modularity matrix.

In the simple setting where $\mathbf{C}$ is binary, such that $\mathbf{C} \in \{0, 1\}^{n\times k}$, then Q_M reasonably represents the modularity across the community graph. Note that $\mathbf{A}^{\mathcal{C}}$ is the weighted adjacency matrix of a graph where nodes are communities and the weight $\mathbf{A}^{\mathcal{C}}_{ij}$ is twice the number of edges between community $\mathcal{C}_i$ and community $\mathcal{C}_j$. The diagonal entries $\mathbf{A}^{\mathcal{C}}_{ii}$, therefore, represent the weight from community i to itself and are equal to double the number of edges between the nodes within community $\mathcal{C}_i$. We also observe that $\mathbf{D}_{\mathcal{C}} = \mathbf{C}^\top\mathbf{d}$ is the community degree matrix and that $\mathbf{D}_{\mathcal{C}}\mathbf{D}_{\mathcal{C}}^\top/(2\mid\mathcal{E}\mid)$ represents the likelihood of an edge existing between communities. As such, we can interpret Q_M as the modularity of the graph in which nodes are replaced with their corresponding communities.

In the more practical case of non-binary community memberships with $\mathbf{C} \in [0,1]^{n \times k}$, we can interpret Q_M as the modularity across "fuzzy" communities, where each entry of the matrix is proportional to the corresponding community membership strengths.

We can now state our objective as maximizing the diagonal values of Q_M while minimizing off-diagonal entries that correspond to dissimilar communities. Clearly, then, our target diagonal values should be 1. However, setting the target off-diagonal values to 0 would penalize overlapping community detection. For this reason, we define a target distribution $y \in \mathbb{R}^{2k}$ as follows:

$$y_i = \begin{cases} 1 & \text{if } i \leq k \\ \delta & \text{otherwise} \end{cases} \tag{3}$$

where δ is a threshold parameter set to 0 in the non-overlapping setting and a pre-determined value in the overlapping setting.[1] A $2k$ vector is necessary to enforce the similarity between the first 1, .., k elements and dissimilarity among the next $k+1, \ldots, 2k$ elements. Under the distribution in Eq. 3, we optimize for community-wide modularity by matching intra-community similarities $(Q_M)_{ii}$ to the target $y_{i;i \leq k}$ and inter-community similarities $(Q_M)_{jl;l \neq j}$ to the target $y_{j;j>k}$. Thus, our loss function becomes

$$\mathcal{L}_{UCoDe} = -\frac{1}{2k}\sum_{i=1}^{k}\underbrace{((\mathbf{y}_i \log(\mathrm{dg}(\sigma(Q_M))_i)}_{\text{intra-community}} + \underbrace{(1-\mathbf{y}_{k+i})\log(1-\mathrm{dg}(\mathcal{P}(\sigma(Q_M)))_i)}_{\text{inter-community}} \tag{4}$$

where dg extracts the vector of the diagonal of a matrix, $\mathcal{P}$ returns a random row-permuted matrix, and σ is the element-wise sigmoid. The row permutation ensures that every community is compared repulsively to another community, as the post-permutation diagonal contains the community modularity between separate clusters. Although the loss allows for including multiple permutations $\mathcal{P}$ of the modularity matrix, in practice, we only consider one as we find that this choice strikes a balance between speed and quality. Thus, this loss function has the straightforward interpretation of clustering similar groups of nodes while encouraging separation between dissimilar ones.

Note that $\mathcal{L}_{UCoDe}$ has a natural relationship to cross-entropy and contrastive objective functions. In the non-overlapping setting, it corresponds to the cross-entropy loss as it represents the KL divergence between Bernoulli random variables. Our target is not a probability distribution in the overlapping setting, however, requiring us to scale the loss by $(1+\delta)$ to recover the cross-entropy interpretation.

4.2.2 A loss for overlapping and non-overlapping communities

Our loss in Eq. 4 clearly encourages non-overlapping community structure by maximizing the diagonal of Q_M and minimizing the off-diagonal. It is less clear whether such a loss also supports overlapping community detection. To this end, we consider the bowtie graph depicted in Fig. 1 with 5 vertices and edges $\mathcal{E} = \{[v_1, v_2], [v_1, v_3], [v_2, v_3], [v_3, v_4], [v_3, v_5], [v_4, v_5]\}$; $v_{1,2,4,5}$ have degree 2, v_3 has degree 4. The optimal overlapping clustering then groups vertices v_1, v_2, v_3 into community c_1, v_3, v_4, v_5 into c_2 with v_3 shared among c_1 and c_2.

[1] $\delta = 0.85$ in all datasets in our experimental cohort.

Fig. 1 Bowtie graph

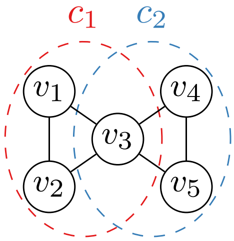


Table 2 NMI scores optimizing only *intra-community* similarity with target function $y_{i;i<k}$, *inter-community* similarity with target function $y_{i;i>k}$, and the UCoDe objective in Eq. 4

	Non-overlapping			Overlapping			
	Cora	Citeseer	Pubmed	fb_348	fb_414	fb_686	fb_1684
Intra-community	51.8	28.0	20.9	23.3	33.5	14.6	25.4
Inter-community	0.0	0.0	0.0	16.3	23.7	11.5	25.7
UCoDe	57.4	**41.0**	**25.0**	**33.9**	**59.9**	**22.1**	**33.3**

The values in bold indicate the model's superior performance, achieved through optimizing both inter-community, intra-community. We did not perform any t-test because of the low variance

If we assume our loss is minimized by non-overlapping communities, it would incentivize orthogonal binary community indicator vectors. WLOG, let $c_1^n = [1, 1, 0, 0, 0]^\top$ and $c_2^n = [0, 0, 1, 1, 1]^\top$ be two such non-overlapping communities. Comparing this to the optimal overlapping clustering $c_1^o = [1, 1, 0.5, 0, 0]^\top$ and $c_2^o = [0, 0, 0.5, 1, 1]^\top$, we obtain

$$\mathcal{L}_{UCoDe}(c_1^n, c_2^n) = 0.124 > \mathcal{L}_{UCoDe}(c_1^o, c_2^o) = 0.094$$

An exhaustive search over all possible communities shows that the minimum of the loss function is the clustering $\mathcal{C} = [c_1^o, c_2^o]$. As such, the loss already encourages overlapping communities. Yet, the value of δ can increase to allow for additional overlap-sensitivity if necessary. In the future, one could consider varying δ on a per-community basis.

We support the above example with an ablation study across datasets. Table 2 shows that optimizing both elements of the contrastive loss yields the best overlapping *and* non-overlapping NMI.

4.3 UCoDe architecture

The main purpose of our GCN is to learn the community assignment matrix **C** using the graph structure and the node attributes. Our architecture is a two-layer GCN (Kipf & Welling, 2017):

$$GCN(\hat{\mathbf{A}}, \mathbf{X}) = \mathtt{RReLU}(\hat{\mathbf{A}} \cdot \underbrace{\mathtt{SiLU}(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}^{[0]})}_{\mathbf{X}^{[1]}} \mathbf{W}^{[1]})$$
$$\text{where } \hat{\mathbf{A}} = \mathbf{D}^{-1/2}(\mathbf{A} + \mathbf{I})\mathbf{D}^{-1/2} \tag{5}$$

The last layer of our GCN outputs community assignments via

$$\mathbf{C} = GCN(\hat{\mathbf{A}}, \mathbf{X})$$

This architecture, although simple, allows for propagating information over the entire graph, thus capturing relationships within the graph's structure and the nodes' attributes.

5 Experiments

In this section, we empirically evaluate UCoDe in comparison with state-of-the-art approaches for community detection on several benchmark graph datasets. We analyze our results in both non-overlapping community detection (graph partitioning), and in overlapping community detection in Sect. 5.1 where nodes may be assigned to more than one community (as discussed in Sect. 5.2). We further analyse the stability of the performance 5.3 and sensitivity of our approach to its few hyperparameters (Sect. 5.4).

We implement UCoDe using PyTorch version 1.10.0 and Python v3.8. We release the implementation of UCoDe at https://github.com/AU-DIS/UCODE. We evaluate our methods on a 14-core Intel Core i9 10940X 3.3GHz machine with 256GB RAM.

Our method UCoDe outputs an assignment matrix $\mathbf{C}$ where c_{ij} represents the likelihood of node v_i belonging to community j. For non-overlapping community detection, we assign the node to the community with the highest score, i.e., $\arg\max_j c_{ij}$.

In additional experiments, we also investigate a second version, **UCoDe**^{k}, which applies the k-means algorithm on the representations obtained by the RReLU function in Eq. 5. Studying this version, we show the benefit of our method compared to decoupled community detection approaches. The results suggest that k-means contributes only marginal quality improvement, which confirms the validity of our efficient end-to-end loss function for community detection.

Adapting to regularization We note that the values in Q_M can be positive or negative and are not necessarily bounded. The sigmoid is thus necessary in order to calculate the cross-entropy to the target distribution. However, we found empirically that the division by $4|\mathcal{E}|$ in Eq. 2 settles the values in $\mathbf{C}^\top\mathbf{B}\mathbf{C}$ close to 0, leaving the sigmoid outputs near 1/2. To this end, we apply a logarithm in $\mathbf{C}^\top\mathbf{B}\mathbf{C}$ that preserves the ordering but amplifies the values. In preliminary experiments, we empirically confirmed that this approach sufficiently amplifies the values so as to achieve good performance when using network regularization.

Competitors We collect results for a number of state-of-the-art non-overlapping (Sect. 1 in the appendix) and overlapping (Sect. 1 in the appendix) community detection methods.

Quality measures For both tasks of overlapping and non-overlapping community detection, we provide the Normalized Mutual Information (NMI) between the cluster assignments and the ground-truth communities. In addition, for non-overlapping community detection we provide the pairwise F1 score between all node pairs and their corresponding ground-truth community; we also provide two intrinsic quality measures, namely

Table 3 Datasets and their main characteristics

Dataset	$\lvert \mathcal{V} \rvert$	$\lvert \mathcal{E} \rvert$	$\lvert \mathbf{A} \rvert$	Dens	Comm	Overl
Cora	2700	5300	1433	.04	7	✗
Citeseer	3300	4600	3703	.04	6	✗
Pubmed	19,700	44,300	500	.01	3	✗
Amz-Pho	7700	71,800	745	.11	8	✗
Amz-PC	13,700	143,600	767	.07	10	✗
CoA-CS	18,300	81,900	6805	.01	15	✗
CoA-Phy	34,500	247,900	8415	.01	5	✗
Fb-348	224	3200	21	12.80	14	✓
Fb-414	150	1700	16	15.10	7	✓
Fb-686	168	1600	9	11.30	14	✓
Fb-698	61	270	6	14.80	13	✓
Fb-1684	786	14,000	15	4.50	17	✓
Fb-1912	747	30,000	29	10.80	46	✓
Eng	14,900	49,300	4800	.04	16	✓

modularity (Eq. 1) and network conductance (Yang & Leskovec, 2015). The network conductance ($\mathcal{C}$) measures how well-connected the nodes in the communities are related to the escape probabilities of random walks. Modularity (Q) (Newman, 2006) assesses whether intra-community nodes are more densely connected than their inter-community counterparts. We report the average value of each measure over 10 runs of the algorithms.

Data We perform experiments on 14 real-world graphs with non-overlapping and overlapping communities. The largest graph has 34.5K nodes and 247K edges. Further details on the datasets, quality measures and parameter settings can be found in Table 3. Our choice of datasets includes graphs with different types of communities, density and attributes, as well as the largest networks evaluated by the competitors.

- **Cora**, **Citeseer**, and **Pubmed** (Sen et al., 2008) are co-citation networks among papers where attributes are bag-of-words representations of the paper's abstracts, and labels are paper topics.
- **Amz-Pho** and **Amz-PC** (Shchur et al., 2018) are subsets of the Amazon co-purchase graph with the frequency of products purchased together; attributes are bag-of-words representations of product reviews, and class labels are product categories.
- **CoA-CS** and **CoA-Phy** (Shchur et al., 2018) are co-authorship networks based on the MS Academic Graph (MAG) for the computer science and physics fields respectively; attributes are collections of paper keywords; class labels indicate common fields of study.
- **Fb-X** datasets (Mcauley & Leskovec, 2014) are ego-nets from Facebook where X is the id of the central node.
- **Eng** (Shchur & Günnemann, 2019) is a co-authorship graph from MAG.

5.1 Non-overlapping community detection

We begin our experimental evaluation with an overall comparison of methods for non-overlapping community detection across different datasets. We compare with the methods described in Sect. 1 in the appendix. We additionally include **NOCD** (Shchur &

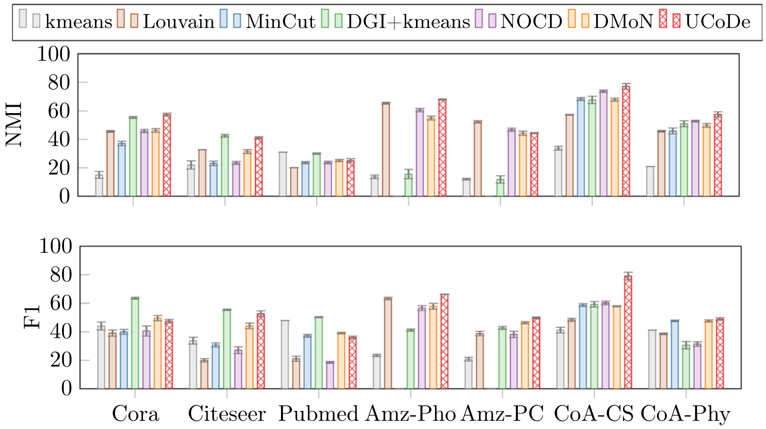


Fig. 2 NMI, F1, and confidence intervals, for non-overlapping community detection

Günnemann, 2019), a state-of-the-art GNN for overlapping community detection. To obtain non-overlapping clusters, we assign each node to the cluster with the highest probability.

UCoDe parameter setup We train UCoDe for 1000 epochs, which shows consistent results across datasets and tasks. We use two GCN layers with a hidden dimension 256. We default to producing $k = 16$ communities for all datasets as this choice is consistent with MinCut (Bianchi et al., 2020) and DMoN (Tsitsulin et al., 2020) and, in a set of preliminary experiments, we found the performance with $k = 8$ and $k = 32$ to give inferior results. We apply batch normalization in both internal layers and set a learning rate 10^{-3} for the Adam optimizer (Kingma & Ba, 2014) for learning. We add weight decay to both weight matrices with regularization strength $\lambda = 10^{-1}$.

We additionally experimented with GraphSAGE (Hamilton et al., 2017) for the internal propagation layer, but opt for GCN (Kipf & Welling, 2017) due to the superior performance in our analyses.

5.1.1 Analysis of ground-truth communities

We compare the methods in terms of NMI and F1-score with respect to ground-truth communities. As Fig. 2 confirms, UCoDe is the most robust choice for non-overlapping communities across datasets. Regardless of dataset characteristics, we observe that UCoDe attains competitive results even where existing approaches under-perform in several datasets. Indeed, a more detailed analysis reveals that UCoDe ranks *on average* higher than any other competitor (Sect. 2 in appendix). The additional k-means clustering offered to DGI^{k}and UCoDe^{k}offers a competitive edge only on three of the seven datasets. Further, note that on the denser Amz-PC and Amz-Pho, methods like MinCut and DCRN fail to converge. They provide overall lower scores, indicating that graph pooling and combined-objectives are not viable approaches for the community detection task. Our method outperforms traditional methods, such as k-means, demonstrating an advantage of a graph-learning approach over attribute clustering to capture the structural characteristics of a graph. NOCD fares relatively good against methods explicitly targeting non-overlapping communities, but still fails to provide competitive results against UCoDe.

Table 4 Graph conductance $\mathcal{C}$ (low is better) and modularity Q

Method	Cora		Citeseer		Pubmed		Amz-Pho		Amz-PC		CoA-CS		CoA-Phy	
	$\mathcal{C}$	Q	$\mathcal{C}$	Q	$\mathcal{C}$	Q	$\mathcal{C}$	Q	$\mathcal{C}$	Q	$\mathcal{C}$	Q	$\mathcal{C}$	Q
k-means	19.0	64.0	26.1	54.2	19.7	54.2	16.9	63.7	83.0	4.0	45.9	20.9	46.0	33.3
DCRN	<u>11.0</u>	71.0	<u>5.6</u>	76.6	**7.8**	0.0	–	–	–	–	21.3	70.0	**13.1**	65.3
DGI^k	12.4	70.7	6.1	74.4	<u>12.4</u>	52.7	49.3	22.4	72.0	12.6	33.6	58.6	38.6	51.2
MinCut	22.0	70.3	11.6	80.5	34.8	58.1	–	–	–	–	<u>19.8</u>	**72.8**	28.8	62.9
NOCD	14.0	**78.3**	6.5	**84.0**	22.2	64.8	<u>14.4</u>	<u>68.8</u>	25.0	**59.0**	20.6	71.8	24.9	<u>65.0</u>
DMoN	22.3	68.1	**4.6**	75.3	17.0	**69.2**	19.1	65.3	<u>19.7</u>	55.8	20.0	<u>72.3</u>	23.8	**65.8**
UCoDe^k	12.3	72.1	8.1	74.8	9.7	54.0	26.3	53.3	44.6	30.0	23.7	66.5	<u>15.8</u>	60.9
UCoDe	**10.9**	<u>76.1</u>	7.1	<u>80.9</u>	17.8	<u>65.4</u>	**9.4**	**69.4**	**13.4**	<u>56.0</u>	**13.9**	70.9	18.7	63.1
Louvain	12.5	81.3	6.2	89.1	15.2	76.9	10.1	74.7	21.0	64.4	17.3	73.6	22.7	66.5

Best performer in bold; second best performer underlined. Louvain is included for reference since a direct comparison is not possible as it is not possible to set the number of communities

In conclusion, there is no clear second choice, promoting UCoDe to be the method of choice, as it shows consistent behavior across datasets.

5.1.2 Analysis of conductance and modularity

We now turn our attention to intrinsic measures to analyze the impact of the various objective functions on community connectedness. Table 4 reports conductance ($\mathcal{C}$) and modularity (Q). UCoDe shows the best performance in terms of conductance, which means that UCoDe is particularly good at identifying well-connected communities. This makes sense, as our loss function specifically encourages high intra-connections and low inter-connections.

At the same time, DMoN, which optimizes for modularity, does not consistently attain the best modularity. Yet, UCoDe attains modularity superior to DMoN in most datasets, although not explicitly encouraging modularity. This indicates that the contrastive loss in UCoDe indeed yields a more nuanced community structure than can be obtained through optimizing modularity alone. This is even more notable when considering the other measures where UCoDe outperforms DMoN.

In conclusion, the empirical evaluation clearly shows that our model is highly robust and widely applicable in the non-overlapping setting, obtaining competitive results across the evaluation metrics and datasets rather than targeting any single one. We note that methods that directly optimize modularity achieve good modularity scores at the expense of performance on other measures. UCoDe instead achieves competitive results across every metric with little-to-no hyperparameter tuning.

5.2 Overlapping community detection

Here, we analyze the performance of UCoDe on overlapping community detection. The list of competitors is described in Sect. 1 in the appendix.

Table 5 NMI for overlapping community detection; CDE, SNMF, and BigClam results are from (Shchur & Günnemann, 2019)

Dataset	CDE	SNMF	BigClam	NOCD	COPRA	DMoN	UCoDe
Fb-348	24.8	13.5	26.0	33.6	14.6	19.9	**33.9**
Fb-414	28.7	32.5	48.3	53.0	45.4	39.0	**59.5**
Fb-686	13.5	11.6	13.8	18.5	9.0	12.6	**22.1**
Fb-698	31.6	28.0	**45.6**	34.4	38.5	19.8	34.9
Fb-1684	28.8	13.0	32.7	30.0	32.7	32.7	**33.3**
Fb-1912	15.5	23.6	21.4	**35.7**	27.6	24.7	33.1
Eng	–	10.1	7.9	**33.3**	1.0	28.5	**33.2**

We highlight the model with the best performance. We did not perform any t-test because of the low variance

UCoDe parameter setup While UCoDe does not require hyperparameter tuning across datasets, it requires small adaptations across tasks to accommodate for the uncertain nature of overlapping communities. To reflect the intrinsic dimensionality of each dataset that grows with the number of nodes (Tsitsulin et al., 2019), we set **the size of the first layer** to 128 while keeping the output layer's size fixed to the number of communities k. We apply **batch normalization** after the first graph convolutional layer. We add **weight decay** to both weight matrices with regularization strength $\lambda = 10^{-2}$. The rest of the hyperparameters are the same as in non-overlapping community detection.

We set the **diagonal elements** of the permuted matrix $\mathcal{P}(Q_M)$ in Eq. 4 to a value $\delta \in [0, 1]$ to avoid penalizing intra-cluster connections. We find experimentally $\delta = 0.85$ to attain good experimental results *on all datasets*, without the need for further tuning.

Community assignment. In the overlapping scenario, we set a threshold p for scores c_{ij} above which a node i is assigned to a community j. We set a threshold that exhibits good average performance on all the datasets, thereby eschewing per-dataset tuning. Our first threshold p_1 is the average of the exp of the assignment scores, i.e., $\frac{1}{nk}\sum_{ij}\exp(c_{ij})$, where the exp encourages sparsity by distributing the values on the range $[0, +\infty)$. We note in Fig. 5 that this choice corresponds to elbow points in a grid search. For the NOCD model, we set $p_2 = 0.5$ as in their experiments. We evaluate the DMoN model using p_1 and p_2, and $p_3 = \mathbb{E}[\mathbf{C}]$ and report results with p_3 as they were the highest in all experiments.

5.2.1 Analysis of ground-truth communities

Overlapping community detection results are given in Table 5 and verify that UCoDe outperforms the state-of-the-art methods on the majority of datasets. The direct optimization of modularity in DMoN cannot easily detect overlapping communities, as opposed to our contrastive modularity loss. More importantly, UCoDe outperforms NOCD in many cases, a GCN that directly aims to detect overlapping communities. Lastly, we note that none of the other methods in Table 5 obtain comparable results to our method. This suggests that our method, that requires *no hyperparameter tuning*, is an effective choice for *both* overlapping and non-overlapping community detection.

Table 6 Non-overlapping community detection: NMI and confidence intervals

Dataset	DGI^k	MinCut	NOCD	DMoN	UCoDe^k	UCoDe
Cora	55.4 ± 0.7	37.1 ± 1.6	45.8 ± 1.2	46.3 ± 1.3	55.7 ± 0.6	57.4 ± 1.0
Citeseer	42.5 ± 1.0	23.1 ± 1.6	23.4 ± 1.1	31.4 ± 1.4	44.4 ± 0.6	41.0 ± 0.9
Pubmed	30.0 ± 0.5	23.6 ± 0.8	23.7 ± 0.9	25.1 ± 0.8	23.5 ± 0.7	25.0 ± 1.4
Amz-Pho	15.6 ± 3.3	–	60.6 ± 1.2	55.0 ± 1.4	61.0 ± 1.0	67.8 ± 0.4
Amz-PC	11.8 ± 2.6	–	46.8 ± 1.1	44.3 ± 1.5	38.5 ± 0.4	44.4 ± 0.4
CoA-CS	67.5 ± 2.6	68.1 ± 1.1	73.6 ± 0.9	67.7 ± 1.1	78.4 ± 0.2	77.0 ± 2.1
CoA-Phy	51.0 ± 2.1	45.9 ± 2.1	52.8 ± 0.6	49.8 ± 1.4	55.3 ± 1.7	57.5 ± 1.9

Table 7 Overlapping community detection: NMI and confidence intervals

Dataset	NOCD	UCoDe
Fb-348	33.6 ± 0.9	33.9 ± 1.1
Fb-414	53.0 ± 1.1	59.5 ± 1.1
Fb-686	18.5 ± 1.0	22.1 ± 1.2
Fb-698	34.4 ± 0.3	34.9 ± 1.4
Fb-1684	30.0 ± 1.9	33.3 ± 1.2
Fb-1912	35.7 ± 1.6	33.1 ± 0.6
Eng	33.3 ± 1.8	33.2 ± 0.9

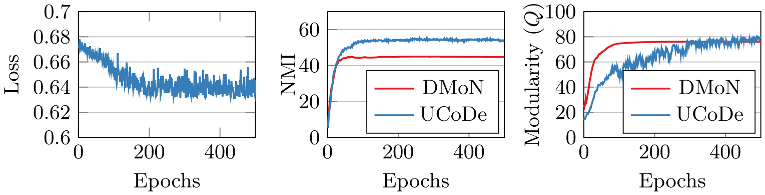


Fig. 3 Training (Cora): UCoDe quickly minimizes the loss (left); NMI increases steadily and achieves 21% higher value than DMoN (center); UCoDe gradually outperforms DMoN's modularity (right). The alternating convergence pattern is characteristic of contrastive loss

5.3 Stability analysis

After having established the top-performers in the respective tasks, namely MinCut, DGI^k , NOCD, and DMoN for non-overlapping community detection, and NOCD for overlapping community detection, we analyze them in terms of variance. Table 6 shows the NMI and the confidence intervals at 95% level. UCoDe retains competitive stability across datasets. More interestingly, while our k-means variant, UCoDe^k, attains lower variance due to the k-means clustering, UCoDe is typically comparable and sometimes more stable than the competitors.

In overlapping community detection in Table 7, we compare only with NOCD that competes with UCoDe. The probabilistic nature of the two methods is reflected in the deviation, which is typically around 1.0. However, in most of the cases, the deviation does not affect the final result and shows that UCoDe is competitive regardless of the variance.

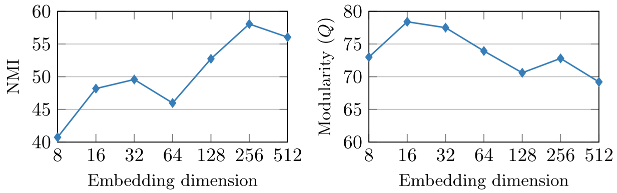
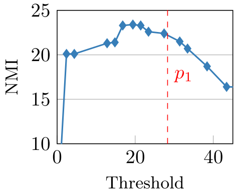


Fig. 4 Impact of the embedding dimension for non-overlapping dataset Cora (similar for other data). The maximum modularity (right) does not correspond to the best NMI (left). The optimal embedding dimension for the intermediate layer is 256

Fig. 5 NMI vs. community assignment threshold; Fb-686 dataset



5.4 Sensitivity analysis

We analyse UCoDe as a function of the number of training epochs. Figure 3 gives results for non-overlapping community detection for the Cora dataset (other datasets show similar trends). As expected in Fig. 3 (left), the loss decreases with more epochs and converges after only about 100 epochs. This behaviour is confirmed in the NMI score (center). Figure 3 (right) compares the modularity score for DMoN and UCoDe per training epoch. We note that while the contrastive loss stabilizes after 200 epochs, the modularity continues to increase until it outperforms DMoN, confirming our analysis in Sect. 4.1.

We study the impact of the embedding dimension on the quality of the communities. In Fig. 4, we report both NMI and modularity for the Cora dataset (other datasets show similar trends). For NMI, we note that increasing the dimension is beneficial until the dimension reaches 256-512. After that point, the quality plateaus and gently decreases. We settle on 256 dimensions as it exhibits a consistent behaviour across datasets and tasks. For overlapping community detection, 128 and 256 dimensions display comparable results; we opt for 128 for the sake of efficiency.

On the other hand, modularity is maximum at 16 dimensions. This discrepancy between NMI and modularity reinforces once more the observation that the pure modularity optimization of models such as DMoN does not necessarily lead to superior quality. Finally, the results for other datasets follow a similar trend, confirming the robustness of UCoDe.

Figure 5 reports the overlapping threshold p_1 for the Fb-686 dataset as an example of a dataset with overlapping communities; we observe similar results in other datasets. The results indicate that there is a relatively broad range of values within [0, 40] in which our method performs well. A threshold > 22 misses relevant community assignments, while a low value assigns every node to all communities. The choice 21 corresponds to the earlier discussed setting p_1 (Fig. 5, red line).

5.5 Ablation study

Table 2 in Sect. 4.1.2 shows the results of the ablation study on three datasets for non-overlapping and four datasets for overlapping community detection. We experiment with a variant of our loss function in Eq. 4 only with intra-cluster similarity (modularity), only with inter-cluster similarity (row-permuted modularity) and UCoDe's loss. In non-overlapping community detection, the intra-cluster similarity produces noisy communities. Yet, the results improve significantly with a combination of the two modularity scores, as the objective drives the model to discriminate true communities from noise. In overlapping community detection, the effect of the row-permuted modularity is more tangible and vindicates the choice of our contrastive loss showing a sensitive increase in performance when both similarities are introduced. Furthermore, the introduction of overlapping community probabilities in UCoDe effectively encourages the model to discover nodes belonging to multiple communities. The results show that the combination of the intra-cluster and the inter-cluster similarity brings the largest benefit.

6 Conclusion

We propose UCoDe, a new Graph Neural Network method for community detection in attributed graphs. UCoDe performs both overlapping and non-overlapping community detection, by virtue of a novel contrastive loss that maximizes a soft version of network modularity. Our experimental assessment confirms that our method is expressive and overall superior in both overlapping and non-overlapping community detection tasks, exhibiting competitive performance in comparison with state-of-the-art methods designed for either one of the tasks.

A Additional material

Here we introduce additional material for reproducing the experiments and support further the analyses in the paper.

A.1 Baselines

A.1.1 Non-overlapping

We evaluate our method against the following established non-overlapping community detection methods:

- *k*-**means** clusters node attributes with the *k*-means++ algorithm (Arthur & Vassilvitskii, 2007); we use the implementation in the scikit-learn package[2].
- **Louvain** (Blondel et al., 2008) is a heuristic method for modularity maximization; we use the implementation in the NetworkX library[3]

[2] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeanshtml..

[3] https://networkx.org/.

- **DGI^k**: Deep Graph Infomax (DGI) (Veličković et al., 2018) is an unsupervised GNN model. After obtaining DGI node representations, *k*-meansclusters these representations; we use the implementation from the authors[4]
- **DMoN** (Tsitsulin et al., 2020) is a state-of-the-art community detection model that trains a shallow GCN to exclusively optimize graph modularity; we use the implementation from the authors[5]
- **MinCut** (Bianchi et al., 2020) is a graph pooling technique that trains a GNN with a min-cut loss similar to spectral clustering (Shi & Malik, 2000); we use the pytorch implementation from DMoN (Tsitsulin et al., 2020).
- **DCRN** (Liu et al., 2022) is the most recent GNN for non-overlapping community detection. DCRN employs a combined objective and requires a pre-trained DFCN (Tu et al., 2021) network to initialize the model embeddings; the communities are *k*-means clusters of the output embeddings. We downloaded the implementation from the authors[6] including the pre-trained networks. We tried to reproduce the experiments in the best of our capacity using the same version of the libraries, hyperparameters, and code, but the results were inconsistent with the ones reported in (Liu et al., 2022). After a thorough investigation, we realized that the reported values must be the maximum NMI across epochs. However, in an unsupervised task, the ground-truth communities are unknown, hence the maximum NMI is unknown as well. Even considering the maximum value, the method does not attain the results declared in the paper. We, therefore, report results obtained using standard evaluation methodology, i.e., NMI for a fixed number of training epochs, consistent with the remainder of our experiments.

Overlapping

We compare UCoDe against the following baselines and state-of-the-art methods for overlapping community detection, including DMoN:

- **NOCD** (Shchur & Günnemann, 2019) is a GCN based on the BigClam objective; we use the implementation from the authors[7]
- **COPRA** (Gregory, 2010) discovers an arbitrary number of overlapping communities via label propagation; we use the implementation from the authors[8]
- **CDE** (Li et al., 2018) and **SNMF** (Wang et al., 2010) employ non-negative matrix factorization to detect communities; results are from (Shchur & Günnemann, 2019)
- **BigClam** (Yang and Leskovec, 2013) finds overlapping communities optimizing the parameters of a Bernoulli-Poisson model; results are from (Shchur & Günnemann, 2019).

[4] https://github.com/PetarV-/DGI.

[5] https://github.com/google-research/google-research/tree/master/graph_embedding/dmon.

[6] https://github.com/yueliu1999/DCRN.

[7] https://github.com/shchur/overlapping-community-detection.

[8] https://gregory.org/research/networks/copra/.

Table 8 NMI for non-overlapping community detection

Dataset	k-means	Louvain	DCRN	DGI^k	MinCut	NOCD	DMoN	UCoDe^k	UCoDe
Cora	15.0	45.60	33.3	55.4	37.1	45.8	46.3	55.7	**57.4**
Citeseer	22.0	32.7	23.0	42.5	23.1	23.4	31.4	**44.4**	41.0
Pubmed	**31.0**	20.1	0.0	30.0	23.6	23.7	25.1	23.5	25.0
Amz-Pho	13.6	65.2	–	15.6	–	60.6	55.0	61.0	**67.8**
Amz-PC	12.0	**52.3**	–	11.8	–	46.8	44.3	38.5	44.4
CoA-CS	33.8	57.3	63.3	67.5	68.1	73.6	67.7	**78.4**	77.0
CoA-Phy	20.9	45.7	54.7	51.0	45.9	52.8	49.8	55.3	**57.5**

The best performer is highlighted in bold, and the second best is underlined

Table 9 F1-result scores for non-overlapping community detection results on the seven real world data sets as summarized in Fig. 2

Dataset	k-means	Louvain	DCRN	DGI^k	MinCut	NOCD	DMoN	UCoDe^k	UCoDe
Cora	44.1	39.1	47.8	**63.4**	40.0	40.6	49.6	61.1	47.5
Citeseer	33.8	20.0	48.1	55.5	30.7	27.1	44.2	**55.9**	52.7
Pubmed	47.9	21.1	22.3	**50.3**	37.2	18.5	39.1	49.3	36.1
Amz-Pho	23.4	63.2	–	41.2	–	56.7	57.9	52.9	**66.2**
Amz-PC	20.8	38.8	–	42.7	–	38.2	46.4	36.0	**49.8**
CoA-CS	41.2	48.4	42.3	59.3	58.8	60.2	58.0	**79.7**	79.0
CoA-Phy	41.2	38.6	56.6	30.6	47.7	31.4	47.6	48.5	**49.0**

The best performer is highlighted in bold, and the second best is underlined

Table 10 Results of the t-tests using Normalized Mutual Information (NMI)

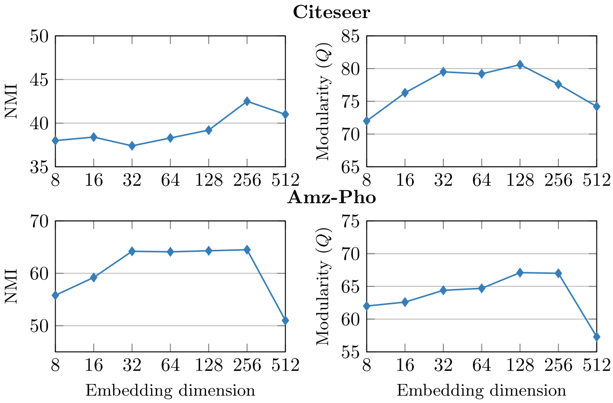
Dataset	Cora	Citeseer	Pubmed	Amz-Pho	Amz-PC	CoA-CS	CoA-Phy
k-means	↑	↑	↓	↑	↑	↑	↑
Louvain	↑	↑	↑	↑	↓	↑	↑
DGI^k	↑	↓	↓	↑	↑	↑	↑
MinCut	↑	↑	↑	↑	↑	↑	↑
NOCD	↑	↑	↑	↑	↓	↑	↑
DMoN	↑	↑	–	↑	–	↑	↑

p-value < 0.05; the arrows indicate statistical significance; ↑ indicates that UCoDe is significantly better than the competitor

Complete results for non-overlapping communities

For completeness, Table 8 reports the NMI values and Table 9 the F1 values for each method reported in Fig. 2.

Fig. 6 Impact of the embedding dimension on NMI and modularity for non-overlapping community detection; Amz-Pho and Citeseer datasets



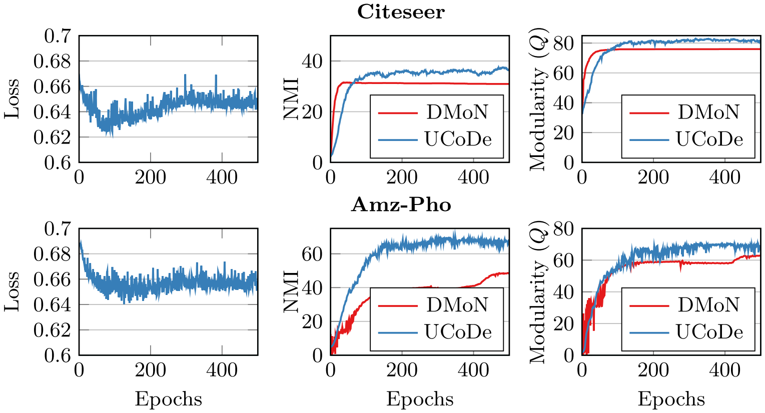


Fig. 7 Training UCoDe that quickly minimizes the loss (left); NMI increases steadily and achieves 9% (19% in Amz-Pho) higher value than DMoN (center); UCoDe gradually outperforms DMoN's modularity (right)

Statistical significance test

We perform an individual two-sided t-test using NMI to compare each model with UCoDe. The arrows in Table 10 indicate a statistically significant difference (with p-value < 0.05) compared to UCoDe. The results demonstrate that in 85% of the cases, UCoDe is significantly better than the competitors.

Extended sensitivity analysis

Figure 6 extends the analysis in Sect. 5.4 to the Citeseerand Amz-Pho datasets. The results consistently indicate 256 as an optimal embedding dimension for the intermediate layer.

We additionally present the analysis of the loss function for Citeseer and Amz-Pho in Fig. 7. The loss exhibits a steady increasing behaviour, stabilizing around 100 epochs as experienced in Sect. 5.4. Interestingly, DMoN's performance is more fluctuating in Amz-Pho as opposed to other datasets.

Author Contributions Atefeh Moradan contributed to the concept, the experiments, the writing, and the algorithms. Andrew Draganov contributed to the theory, part of the experiments, and writing. Davide Mottin and Ira Assent contributed to the supervision, the writing, and the correction of the paper.

Funding Open access funding provided by Royal Danish Library, Aarhus University Library. Atefeh Moradan is supported by the Innovationsfonden Denmark under the Grand Solutions project Hospital@Night.

Data availability The data and the code are available at https://github.com/AU-DIS/UCODE.

Code availability The data and the code are available at https://github.com/AU-DIS/UCODE.

Declarations

Conflicts of interest The authors have conflicts with Aarhus university (au.dk). Juelich research center (fz-juelich.de). Ira Assent: Co-author. laria Bordino: Recent collaborator. rancesco Gullo: Recent collaborator. Panagiotis Karras: Colleague. Thomas Seidl: PhD advisor.

Ethical approval Not applicable.

Consent to participate The authors provide the appropriate consent to participate.

Consent for publication The authors provide the consent to publish the images in the manuscript. The data used in the publication is publicly available. We provide respective citations for each of the data sources.

References

Arthur, D., Vassilvitskii, S. (2007). k-means++ the advantages of careful seeding. In: SODA, pp. 1027–1035

Battaglia, P.W., Hamrick, J.B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. (2018). Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261

Bianchi, F.M., Grattarola, D., Alippi, C. (2020). Spectral clustering with graph neural networks for graph pooling. In: International Conference on Machine Learning, pp. 874–883. PMLR

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment, 2008*(10), 10008.

Bo, D., Wang, X., Shi, C., Zhu, M., Lu, E., Cui, P. (2020). Structural deep clustering network. In: Proceedings of The Web Conference 2020, pp. 1400–1410

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., & Vandergheynst, P. (2017). Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine, 34*(4), 18–42.

Bruna, J., Zaremba, W., Szlam, A., LeCun, Y. (2014). Spectral networks and deep locally connected networks on graphs. In: ICLR

Cai, H., Zheng, V. W., & Chang, K.C.-C. (2018). A comprehensive survey of graph embedding: Problems, techniques, and applications. *TKDE, 30*(9), 1616–1637.

Chami, I., Abu-El-Haija, S., Perozzi, B., Ré, C., Murphy, K. (2020). Machine learning on graphs: A model and comprehensive taxonomy. arXiv preprint arXiv:2005.03675

Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. *NeurIPS, 29*, 3844–3852.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological), 39*(1), 1–22.

Devi, J. C., & Poovammal, E. (2016). An analysis of overlapping community detection algorithms in social networks. *Procedia Computer Science, 89*, 349–358.

Dhillon, I. S., Guan, Y., & Kulis, B. (2007). Weighted graph cuts without eigenvectors a multilevel approach. *TPAMI, 29*(11), 1944–1957.

Fortunato, S. (2010). Community detection in graphs. *Physics reports, 486*(3–5), 75–174.

Fortunato, S., & Barthelemy, M. (2007). Resolution limit in community detection. *Proceedings of the national academy of sciences, 104*(1), 36–41.

Gao, H., Ji, S. (2019). Graph u-nets. In: ICML, pp. 2083–2092

Gao, H., Huang, H. (2018). Deep attributed network embedding. In: IJCAI

Girvan, M., & Newman, M. E. (2002). Community structure in social and biological networks. *PNAS, 99*(12), 7821–7826.

Gregory, S. (2010). Finding overlapping communities in networks by label propagation. *New journal of Physics, 12*(10), 103018.

Gregory, S. (2007). An algorithm to find overlapping community structure in networks. In: European Conference on Principles of Data Mining and Knowledge Discovery, pp. 91–102. Springer

Hamilton, W.L., Ying, R., Leskovec, J. (2017). Inductive representation learning on large graphs. In: NIPS, pp. 1025–1035

He, D., Song, Y., Jin, D., Feng, Z., Zhang, B., Yu, Z., Zhang, W.: Community-centric graph convolutional network for unsupervised community detection. In: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, pp. 3515–3521 (2021)

Kingma, D.P., Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980

Kipf, T.N., Welling, M. (2016). Variational graph auto-encoders. Bayesian Deep Learning Workshop at NIPS

Kipf, T.N., Welling, M. (2017) Semi-supervised classification with graph convolutional networks. In: ICLR. OpenReview.net

Lee, J., Lee, I., Kang, J. (2019). Self-attention graph pooling. In: ICML, pp. 3734–3743

Li, Y., Sha, C., Huang, X., Zhang, Y. (2018). Community detection in attributed graphs: An embedding approach. In: AAAI

Liu, Y., Tu, W., Zhou, S., Liu, X., Song, L., Yang, X., Zhu, E. (2022). Deep graph clustering via dual correlation reduction. In: Proc. of AAAI

Mall, R., Ullah, E., Kunji, K., Bensmail, H., Ceccarelli, M. (2017). An adaptive refinement for community detection methods for disease module identification in biological networks using novel metric based on connectivity, conductance & modularity. In: BIBM, pp. 2282–2284

Mcauley, J., & Leskovec, J. (2014). Discovering social circles in ego networks. *TKDD, 8*(1), 1–28.

Newman, M. E. (2006). Modularity and community structure in networks. *PNAS, 103*(23), 8577–8582.

Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W. (2016). Asymmetric transitivity preserving graph embedding. In: KDD, pp. 1105–1114

Park, N., Rossi, R., Koh, E., Burhanuddin, I.A., Kim, S., Du, F., Ahmed, N., Faloutsos, C. (2022). Cgc: Contrastive graph clustering forcommunity detection and tracking. In: Proceedings of the ACM Web Conference 2022, pp. 1115–1126

Perozzi, B., Al-Rfou, R., Skiena, S. (2014). Deepwalk: online learning of social representations. KDD

Qiu, J., Dong, Y., Ma, H., Li, J., Wang, K., Tang, J. (2018). Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In: WSDM, pp. 459–467

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., & Eliassi-Rad, T. (2008). Collective classification in network data. *AI magazine, 29*(3), 93–93.

Shchur, O., & Günnemann, S. (2019). *Overlapping community detection with graph neural networks*. KDD: Deep Learning on Graphs Workshop.

Shchur, O., Mumme, M., Bojchevski, A., Günnemann, S. (2018). Pitfalls of graph neural network evaluation. arXiv preprint arXiv:1811.05868

Shi, J., & Malik, J. (2000). *Normalized cuts and image segmentation. TPAMI, 22*(8), 888–905.

Sun, H., Li, Y., Lv, B., Yan, W., He, L., Qiao, S., & Huang, J. (2021). *Graph community infomax. TKDD, 16*(3), 1–21.

Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q. (2015). Line: Large-scale information network embedding. In: WWW, pp. 1067–1077

Tsitsulin, A., Palowitch, J., Perozzi, B., Müller, E. (2020). Graph clustering with graph neural networks. arXiv preprint arXiv:2006.16904

Tsitsulin, A., Mottin, D., Karras, P., Bronstein, A., Müller, E. (2019). Spectral graph complexity. In: Companion Proceedings of The Web Conf, pp. 308–309

Tu, W., Zhou, S., Liu, X., Guo, X., Cai, Z., Zhu, E., & Cheng, J. (2021). *Deep fusion clustering network. In: AAAI, 35*, 9978–9987.

Veličković, P., Fedus, W., Hamilton, W.L., Liò, P., Bengio, Y., Hjelm, R.D. (2018). Deep graph infomax. In: ICLR

Wang, F., Li, T., Wang, X., Zhu, S., & Ding, C. (2010). Community discovery using nonnegative matrix factorization. *Data Mining and Knowledge Discovery, 22*(3), 493–521. https://doi.org/10.1007/s10618-010-0181-y

Wang, D., Cui, P., Zhu, W. (2016). Structural deep network embedding. In: KDD, pp. 1225–1234

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Philip, S. Y. (2020). A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems, 32*(1), 4–24.

Xu, K., Hu, W., Leskovec, J., Jegelka, S. (2018). How powerful are graph neural networks? In: ICLR

Yang, J., & Leskovec, J. (2015). Defining and evaluating network communities based on ground-truth. *KIS, 42*(1), 181–213.

Yang, J., McAuley, J., Leskovec, J. (2013). Community detection in networks with node attributes. In: 2013 IEEE 13th International Conference on Data Mining, pp. 1151–1156. IEEE

Yang, J., Leskovec, J. (2012). Community-affiliation graph model for overlapping network community detection. In: ICDM, pp. 1170–1175

Yang, J., Leskovec, J. (2013). Overlapping community detection at scale: a nonnegative matrix factorization approach. In: WSDM, pp. 587–596

Yang, C., Liu, Z., Zhao, D., Sun, M., Chang, E. (2015). Network representation learning with rich text information. In: IJCAI

Ying, R., You, J., Morris, C., Ren, X., Hamilton, W.L., Leskovec, J. (2018). Hierarchical graph representation learning with differentiable pooling. In: NeurIPS, pp. 4805–4815

Zhang, X., Liu, H., Li, Q., Wu, X.M. (2019). Attributed graph clustering via adaptive graph convolution. In: IJCAI, pp. 4327–4333

Zhang, T., Xiong, Y., Zhang, J., Zhang, Y., Jiao, Y., Zhu, Y. (2020). Commdgi: Community detection oriented deep graph infomax. In: CIKM, pp. 1843–1852

Zhao, H., Yang, X., Wang, Z., Yang, E., Deng, C.: Graph debiased contrastive learning with joint representation clustering. In: IJCAI, pp. 3434–3440 (2021)

Zhou, M. (2015). Infinite Edge Partition Models for Overlapping Community Detection and Link Prediction