RepBin: Constraint-Based Graph Representation Learning for Metagenomic Binning

Hansheng Xue,[1] Vijini Mallawaarachchi,[1] Yujia Zhang,[1] Vaibhav Rajan,[2] Yu Lin[1]*

[1]School of Computing, The Australian National University, Canberra, Australia
[2]School of Computing, National University of Singapore, Singapore
{hansheng.xue, vijini.mallawaarachchi, yujia.zhang, yu.lin}@anu.edu.au, vaibhav.rajan@nus.edu.sg

Abstract

Mixed communities of organisms are found in many environments – from the human gut to marine ecosystems – and can have profound impact on human health and the environment. Metagenomics studies the genomic material of such communities through high-throughput sequencing that yields DNA subsequences for subsequent analysis. A fundamental problem in the standard workflow, called binning, is to discover clusters, of genomic subsequences, associated with the unknown constituent organisms. Inherent noise in the subsequences, various biological constraints that need to be imposed on them and the skewed cluster size distribution exacerbate the difficulty of this unsupervised learning problem. In this paper, we present a new formulation using a graph where the nodes are subsequences and edges represent homophily information. In addition, we model biological constraints providing heterophilous signal about nodes that cannot be clustered together. We solve the binning problem by developing new algorithms for (i) graph representation learning that preserves both homophily relations and heterophily constraints (ii) constraint-based graph clustering method that addresses the problems of skewed cluster size distribution. Extensive experiments, on real and synthetic datasets, demonstrate that our approach, called RepBin, outperforms a wide variety of competing methods. Our constraint-based graph representation learning and clustering methods, that may be useful in other domains as well, advance the state-of-the-art in both metagenomics binning and graph representation learning.

Introduction

The field of *metagenomics* has paved the way to study entire microbial communities obtained from natural environments (Kaeberlein, Lewis, and Epstein 2002). Large scale studies such as the Human Microbiome Project (Turnbaugh et al. 2007) have leveraged metagenomics analyses to gain valuable insights about the complex microbial communities found in the human body, and study the relationships of these microbial communities with health conditions and diseases such as pregnancy and preterm birth (PTB), inflammatory bowel diseases (IBD), stressors affecting prediabetes (Integrative et al. 2019), influence of diet on metabolism (Pasolli et al. 2020; Asnicar et al. 2021) and disease association of microbial species found in the human gut (Nayfach et al. 2019).

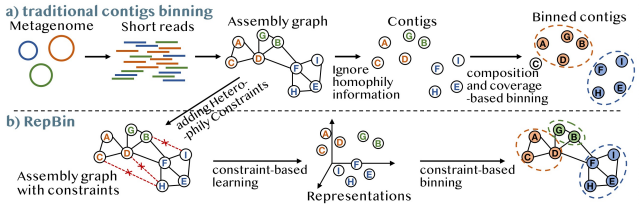

Figure 1: The pipeline of traditional metagenomic contigs binning and our proposed method, RepBin.

In a typical metagenomics workflow, genetic material from a microbial community is collected and first sequenced using high-throughput sequencing (HTS) platforms. The output at this stage contains short sequences of DNA called *reads* of all the constituent microorganisms. Note that since the genetic material of all the microorganisms are mixed to begin with, it is unknown which species each read belongs to; further, it is possible that multiple species may contain the same DNA sequence in their reads. A fundamental problem, for any subsequent downstream analysis, is to identify the species involved in the input sample.

Popular metagenomics approaches assemble these short reads into longer non-overlapping DNA sequences called *contigs* using *assembly graphs*, where each vertex represents a contig and each edge represents the homophily information between two contigs (Nurk et al. 2017). Then, these assembled contigs are clustered into specific bins corresponding to the constituent genomes in the microbial communities (referred as *metagenomic binning*). Contigs connected to each other in the assembly graph are more likely to belong to the same species (Barnum et al. 2018). Additional biological information can be used to incorporate constraints on contigs that are likely to be in different bins. For instance, single-copy marker genes are those that appear just once in each species and so, if two contigs contain the same single-copy marker gene, they are most likely to belong to different species. Such constraints can provide heterophilous information and could be utilized to improve binning.

Most metagenomics binning tools ignore the homophily

*Corresponding author.

information in assembly graphs and use the composition and coverage information of contigs for binning (refer to Figure 1a for a traditional pipeline for binning). Moreover, these tools have to discard many short contigs and thus suffer from low recall values because the composition and coverage features become unreliable for short contigs. To use the assembly graph directly, an alternative approach would be to obtain node embeddings (Cui et al. 2019) from the assembly graph, and cluster them to find bins. However, existing graph embedding techniques mainly utilize homophily information and do not model heterophilous relations.

In this paper, we propose a constraint-based graph representation learning model, called *Constraint-based Learning*, which can capture the structural information of the graph while respecting the prior constraints. The *Constraint-based Learning* model is a contrastive graph learning framework with diffusion convolutional operators as basic components, and optimizes prior constraints through a penalty term in the objective function. The learned representations can be used for downstream tasks. For metaganomic contig binning, we propose a GCN-based label annotation model, called *Constraint-based Binning*, with constrained nodes as initial labels. The combined model is called ***RepBin*** and is illustrated in Figure 1b. Our contributions in this paper are:

- To the best of our knowledge, this is the first use of graph representation learning in the important area of metagenomic contig binning.
- We design a novel constraint-based graph representation learning model, called *Constraint-based Learning*, which can capture structural information of the graph while respecting heterophilous constraints.
- We design a novel *Constraint-based Binning* strategy which uses Graph Convolutional Networks to annotate unknown contigs with labels, using constrained contigs to obtain initial labels.
- We benchmark the performance of our method on contigs binning, against state-of-the-art methods for metagenomics binning, graph representation learning and graph clustering. Results show that *RepBin* significantly outperforms them on both simulated and real-world datasets.
- Both algorithms, Constraint-based Learning and Binning, are general-purpose graph representation learning and clustering methods, and can be used in other domains as well, where constraints need to be incorporated in these tasks.

Related Work

Metagenomic Binning. Although contigs are assembled from short reads using assembly graphs in metagenomic samples, most existing binning tools ignore the homophily information in assembly graphs. Instead, these binning tools use the composition (normalised oligonucleotide, *i.e.*, short string of length k, frequencies) and coverage (average number of reads aligning to each position of the contig) information to bin contigs, *e.g.*, in MetaWatt (Strous et al. 2012), CONCOCT (Alneberg and et al. 2014), and MaxBin2 (Wu and et al. 2015). MetaBAT2 (Kang and et al. 2019) employs a graph clustering approach, where the graph is constructed by composition information of contigs. SolidBin (Wang et al. 2019) uses semi-supervised spectral clustering that incorporates additional biological knowledge. VAMB (Nissen and et al. 2021) uses deep variational autoencoders to integrate both composition and coverage information and clusters the resulting latent representation of contigs into bins. BusyBee Web (Laczny et al. 2017) is a web-based application which makes use of a bootstrapped supervised binning approach to bin contigs. However, all these tools have to discard short contigs (*e.g.*, shorter than 1000bp) because the composition and coverage features become unreliable for short contigs, and thus suffer from low recall values. To recover these short contigs, recently published bin-refinement tools (Mallawaarachchi and et al. 2020a,b) have introduced the use of assembly graphs from which contigs are derived.

Graph Representation Learning. Existing unsupervised graph representation learning models are mainly grouped into three categories, random walk-based (Grover and Leskovec 2016; Perozzi, Al-Rfou, and Skiena 2014), matrix factorization-based (Qiu et al. 2018; Liu et al. 2019), and deep learning-based methods (Kipf and Welling 2016; Veličković et al. 2019). Generative and contrastive models are two typical frameworks for deep learning-based unsupervised graph learning models. Generative learning models, like VGAE (Kipf and Welling 2016), capture the graph structure by minimizing the error between the constructed and original graph in an encoder-decoder architecture. Contrastive learning, such as DGI (Veličković et al. 2019), utilizes discriminator to differentiate nodes from input graph and negative samples. The node embeddings are optimized via maximizing mutual information with graph summary.

Graph Clustering Graph clustering aims to use the graph structure to group nodes in graph into several disjoint clusters, like spectral clustering (Von Luxburg 2007). Many deep graph clustering methods have been proposed (Bianchi, Grattarola, and Alippi 2020; Tsitsulin et al. 2020). These methods mainly use deep learning or GNN models to capture the graph structure and then use clustering algorithms to cluster these learned features (Cao, Lu, and Xu 2016; Fan et al. 2020; Zhang et al. 2019). Methods such as Constraint-based spectral clustering (Wang, Qian, and Davidson 2014) and Deep constrained clustering (Zhang, Basu, and Davidson 2019) are designed to integrate constraints in clustering.

Preliminaries

Consider an assembly graph $G = (V, E)$ from a metagenomic assembler, where each node $v \in V$ represents a contig and each edge $e \in E$ denotes that two corresponding contigs (nodes) are connected in the assembly graph. Let $\mathcal{M}$ be the set of all pairwise heterophily constraints induced by *Single-copy marker genes* which are conserved and appear only once in most of the bacterial genomes (Albertsen et al. 2013). We use the tools FragGeneScan (Rho, Tang, and Ye 2010) and HMMER (Eddy 2011) to identify contigs containing each of the 107 single-copy marker genes. Then, we generate pairwise constraints between contigs containing each single-copy marker gene, *e.g.*, there exists a constraint $m(u, v)$ between u and v if the same single-copy marker

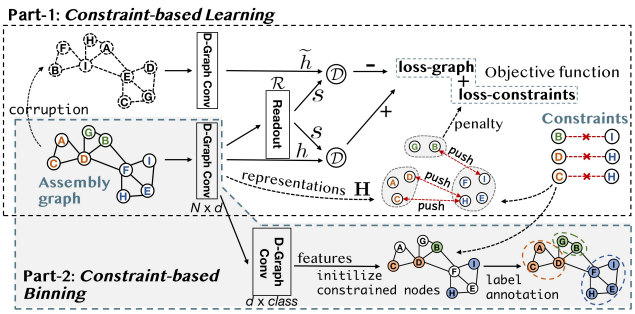

Figure 2: The framework of our proposed RepBin model.

gene appears in both u and v (which indicates that u and v are not expected to be in the same species).

The task of binning metagenomic contigs is to cluster contigs into bins that correspond to different species. Different from other models that directly use k-mer composition and contig coverage to bin contigs, we make use of the structure of assembly graph and label contigs based on the low-dimensional features learned from the assembly graph. The constraint-based representation of a contig from an assembly graph can be defined as:

Definition. Given an assembly graph $G = (V, E)$ and its constraints $\mathcal{M}$, the constraint-based embedding of a node (contig) in the assembly graph is a d-dimensional feature $H \in R^{|V| \times d}$, where $d \ll |V|$, that considers both the topological structure of the assembly graph and the constraints of $\mathcal{M}$ simultaneously.

Methodology

The proposed framework mainly contains two components, *Constraint-based Learning* and *Constraint-based Binning*. Figure 2 shows an overview of the entire RepBin model.

Part 1: *Constraint-based Learning*

In the *Constraint-based Learning* module, we aim to model the homophily graph structure and heterophily constraints. We first describe the contrastive graph learning framework and graph diffusion convolution, then explain how to integrate constraints to obtain the final node embedding matrix.

Contrastive graph learning framework. Here we introduce a contrastive graph representation learning model which tries to obtain representations for nodes that capture the global structure of the entire graph by maximizing the mutual information between node-level (local) and graph-level (global) features. The overall framework of the *Constraint-based Learning* is shown in the part 1 of Figure 2, which is similar to DGI. The proposed model mainly contains graph encoder, negative graph sampling, readout function, discriminator model, and constraints optimization.

The one-layer graph diffusion convolutional operator is used in the *Constraint-based Learning* model as the graph encoder module. We will explain diffusion convolution module in the next subsection. By the diffusion convolution operator in Equation 2, we can obtain the representations of all nodes $\mathbf{H} = \{\mathbf{h_1}, \mathbf{h_2}, ..., \mathbf{h_n}\}$.

The negative graph sampling generates negative graphs by applying a corruption function (*e.g.*, row-wise shuffling on the original attribute matrix, same as the one used in the DGI model) on the original assembly graph, $(\tilde{X}, \tilde{A}) = \mathcal{C}(X, A)$. Both the original and negative graphs are typed into the graph encoder model and generate latent node-level representations $\mathbf{H}$ and $\tilde{\mathbf{H}}$ respectively.

The readout function $\mathbf{s} = \mathcal{R}(\mathbf{H}) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{h}_i$ is a function to obtain a global representation of the whole assembly graph by aggregating all node-level representations. The discriminator model $\mathcal{D}$ is introduced to discriminate the true samples, *i.e.*, $(\mathbf{h_i}, \mathbf{s})$, from its negative counterparts, *i.e.*, $(\tilde{\mathbf{h}}_\mathbf{j}, \mathbf{s})$, by maximizing the mutual information.

Finally, we use a standard binary cross-entropy (BCE) loss between the samples from the joint (positive examples) and the product of marginals (negative examples) to optimize the mutual information (MI) between $\mathbf{h_i}$ and $\mathbf{s}$, same as DGI and DMGI (Park et al. 2020).

$$\mathcal{L}_g = -\frac{1}{2n}[\sum_{i=1}^{n} \log \mathcal{D}(\mathbf{h_i}, \mathbf{s}) + \sum_{j=1}^{n} \log(1 - \mathcal{D}(\tilde{\mathbf{h}}_\mathbf{j}, \mathbf{s}))] \quad (1)$$

Graph diffusion convolution. The main characteristic of the assembly graph which is different from arbitrary graphs is the high homophily score (see Table 1). It indicates that the majority of linked contigs belong to the same species. Most message-passing neural networks (*i.e.*, GCN) mainly aggregate information from one-hop neighbors in each layer and are limited in exploring higher-order neighborhoods, especially in graphs with high homophily. In contrast, diffusion has been found to be superior in graphs with high homophily (Klicpera and et al. 2019a,b). So, in our model, we use the Graph Diffusion Convolution (GDC) operator to capture the high-order structure of the graph.

One successful example of the graph diffusion is the personalized PageRank (PPR) (Page et al. 1999). Assuming that one node i and its teleport vector x_i, the adaptation of the PPR for node i can be calculated by the recurrent equation $\text{PPR}(x_i) = (1-\alpha)\hat{\mathbf{A}}\text{PPR}(x_i) + \alpha x_i$. Parameter $\alpha \in (0, 1]$ denotes the probability of teleporting to another state. We can obtain the transitions state for node i using $\text{PPR}(x_i) = \alpha(\mathbf{I}_N - (1-\alpha)\hat{\mathbf{A}})^{-1}x_i$. Thus, the normalized graph diffusion matrix for graph G can be formulated as:

$$\mathbf{T}_{S_{ij}} = \frac{\mathbf{S}_{ij}}{\sum_i \mathbf{S}_{ij}} \text{ with } \quad \mathbf{S} = \alpha[\mathbf{I}_N - (1-\alpha)\mathbf{T}]^{-1} \quad (2)$$

where $\mathbf{T} = \hat{\mathbf{D}}^{-1/2}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-1/2}$ is the symmetric transition matrix (approximately the spectral graph convolutional operator), $\hat{\mathbf{A}}$ is the adjacency matrix of the graph with self-loops $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$, $\mathbf{I}_N$ is the unit matrix, and $\hat{\mathbf{D}}$ denotes the diagonal degree matrix, *i.e.*, $\hat{\mathbf{D}}_{ii} = \sum_{j=1}^{N} \hat{\mathbf{A}}_{ij}$. The calculated diffusion matrix is dense and computationally inefficient, we can optionally add a sparsity operator (truncating small values less than ϵ: $\mathbf{S}_{ij}(\epsilon) = \mathbf{S}_{ij}$ if $\mathbf{S}_{ij} \geq \epsilon$ else 0) before the normalization.

We use graph diffusion convolution in the contrastive learning model to capture the graph structure. We formulate

the l+1-layer diffusion convolutional operator following the layer-wise propagation rule as:

$$\mathbf{H} = \sigma(\mathbf{T}_S \cdot \mathbf{X}\mathbf{\Theta}) \tag{3}$$

where $\sigma(\cdot)$ denotes a non-linear activation function (*e.g.*, ReLU), $\mathbf{\Theta} \in \mathbb{R}^{N \times d}$ is a trainable transformation matrix, d is the feature dimension, and $\mathbf{X}$ is the initial feature matrix.

Constraints optimization. In metagenomics, side information is given in an implicit way of negative constraints, which indicate pairwise contigs that belong to distinct species, and not in an explicit way (such as known labels for contigs). In our *constraint-based learning* model, we penalize the constrained nodes with respect to their similarity as described below.

In the previous graph diffusion convolution operator, we can obtain latent representations for each node $\mathbf{H} = \{\mathbf{h_1}, \mathbf{h_2}, ..., \mathbf{h_n}\}$. These learned features capture the structural information of the assembly graph and ignore the pairwise heterophily constraints $\mathcal{M}$. Each pairwise constraint $m(i, j) \in \mathcal{M}$ indicates that node i and j should belong to two different species. We calculate the distance $(\mathbf{h}_i, \mathbf{h}_j)$ between pairs of nodes i and j with a constraint $m(i, j)$, and the following objective function can be used to measure how all pairwise constraints are respected:

$$\mathcal{L}_c = \frac{1}{|\mathcal{M}|} \sum_{m(i,j) \in \mathcal{M}} \exp^{-||\mathbf{h}_i - \mathbf{h}_j||_2} \tag{4}$$

where $||.||_2 = \sqrt{\sum_{i=1}^{d} |._i|^2}$ is the Euclidean Norm to calculate the Euclidean distance, $\mathcal{M}$ is the set of heterophily constraints, and d is the dimension of representations. The exponential loss (Friedman et al. 2001) is used to penalize the distance between pairwise constrained nodes.

Objective function. The *constraint-based learning* model mainly contains three components contrastive graph learning framework, graph diffusion convolution, and constraints optimization. It aims to model both the homophily information of the graph structure and pairwise heterophily constraints. The objective function of *constraint-based learning* model can be formulated as the combination of unsupervised graph learning $\mathcal{L}_g$ and constraints optimization $\mathcal{L}_c$:

$$\mathcal{L} = \mathcal{L}_g + \lambda \cdot \mathcal{L}_c \tag{5}$$

where the parameter $\lambda \in (0, 1)$ controls the importance of the heterophily constraints loss.

Part 2: *Constraint-based Binning*

After obtaining node-level latent representations $\mathbf{H}$ from our *Constraint-based Learning* model, we can directly employ clustering algorithms (K-Means) on these features to obtain final bins. However, two challenges remain to be addressed. First, the number of constraints with respect to each contig vary a lot (i,e., from 0 to dozens); Second, the numbers of contigs in distinct bins/species in metagenomic samples are imbalanced (*i.e.*, varying from a couple to hundreds).

To address the above challenges, we introduce a constraints-based label annotation strategy (*Constraint-based Binning*) instead of naive clustering algorithms which over rely on the initial centroids. The *Binning* method comprises of two steps, initializing labels for constraints and GCN-based label annotation (see Figure 2 part 2).

Algorithm 1: The RepBin algorithm

Input: Assembly Graph $G = (V, E)$, constraints $\mathcal{M}$, num. of bins k, initial parameters (*e.g.*, λ and α);
Output: Bins Y;

Model Construction:
Calculate graph diffusion convolution $\mathbf{T}_S$;
Construct *Constraint-based Learning* model:
    Corrupt negative graph $\mathcal{C}(X, A)$;
    Learn postive and negative graphs H and $\tilde{H}$;
    Capture global patterns by Readout function R;
    Maximize the mutual info. by a discriminator D;
Obtain latent features $\mathbf{H}$;
Initialize labels for constrained contigs Y_m;
GCN-based label annotation to obtain final bins Y;
Optimization:
Initialize Embeddings $\mathbf{X}$ with initial features;
Randomly sample negative graph by corruption $\mathcal{C}$;
Optimize loss (5) via gradient descent;
Optimize loss (6) via gradient descent;
return Bins Y;

Initializing labels for constraints. Note that not all contigs have pairwise constraints and contigs without any prior constraints are usually more challenging to bin correctly. We first run a clustering algorithm (K-Means in this work) on the representations of nodes with at least one constraint to obtain initial labels, $Y_m = \{y_1^m, y_2^m, ..., y_k^m\}$. Ideally, the number of bins K should be equal to the number of contigs that contain the same single-copy marker gene. As there are errors in both assemblies and alignments, we need a robust estimate and so, we set K as the third quartile value of the counts of contigs containing each of the marker genes.

GCN-based label annotation. After obtaining initial labels for nodes with at least one constraint, we treat the binning as a semi-supervised label annotation task and construct a two-layer graph diffusion convolutional operator to annotate unlabelled contigs (*i.e.*, nodes without any prior constraints). Similar to the layer-wise propagation rule defined in Equation 3, labels for all node are obtained simultaneously via $\mathbf{Z} = \sigma(\mathbf{T}_S \cdot \mathbf{H}\Theta^{'}) = \sigma(\mathbf{T}_S \cdot (\sigma(\mathbf{T}_S \cdot \mathbf{X}\Theta))\Theta^{'})$. We evaluate the cross-entropy error over all contigs with initial labels and use back propagation for optimization.

$$\mathcal{L} = -\sum_{l \in Y_m} \sum_{k=1}^{K} Y_{lk} \ln \mathbf{Z}_{lk} \tag{6}$$

where Y_m is the set of constrained nodes with initialized labels, K is the number of clusters. The pseudocode for RepBin is shown in Algorithm 1.

Experiments

Datasets. Three simulated and two real-world datasets are used in our experiments. Their detailed statistics are given in Table 1. **Sim-5G, Sim-10G, and Sim-20G** are simulated

Datasets	Nodes	Edges	$\mathcal{M}$	Bins	H
Sim-5G	519	2,488	810	5	0.97
Sim-10G	920	4,210	2,448	10	0.96
Sim-20G	1,452	6,531	10,734	20	0.91
CAMI	5,814	23,257	21,362	19	0.89
Sharon	20,743	102,918	2,988	12	0.95

Table 1: Statistics of datasets. $\mathcal{M}$ is the number of constraints and H is the node-homophily of the assembly graph.

based on the species found in the simMC+ dataset (Wu et al. 2014). **Sharon** is a preborn infant gut metagenome dataset (Sharon, Morowitz, and et al. 2013), and the corresponding NCBI accession number is *SRA052203*[1]. **CAMI** is a publicly available dataset extracted from the first CAMI challenge with low complexity of microbiomes (Sczyrba, Hofmann, and et al. 2017)[2]. All the data sets are assembled using metaSPAdes (Nurk et al. 2017).

Baselines. We compare RepBin with 4 unsupervised GNN models, 4 graph clustering methods, and 8 binning tools. **GNNs:** GraphSAGE (Hamilton, Ying, and Leskovec 2017), GAT (Veličković et al. 2018), DGI (Veličković et al. 2019), and VGAE (Kipf and Welling 2016). **GCs:** O2MAC (Fan et al. 2020), AGC (Zhang et al. 2019), Constrainted Spectral Clustering (CSC) (Wang, Qian, and Davidson 2014), and Deep Constrained Clustering (DCC) (Zhang, Basu, and Davidson 2019). **Binnings:** MetaWatt (Strous et al. 2012), CONCOCT (Alneberg and et al. 2014), MaxBin2 (Wu and et al. 2015), BusyBeeWeb (Laczny et al. 2017), MetaBAT2 (Kang and et al. 2019), SolidBin (Wang et al. 2019), VAMB (Nissen and et al. 2021), and GraphBin (Mallawaarachchi and et al. 2020a).

Metrics and Experimental Settings. The entire procedure is repeated five times to avoid any experimental biases. Mean and standard deviation values of the binning evaluation metrics are reported. For the simulated and CAMI datasets, we map the contigs to the reference genomes to obtain the ground truth species. For the Sharon dataset, we map the contigs to the annotated results[3] to obtain the ground truth species. We use the F1, ARI, and NMI as the evaluation metrics for machine learning-based baselines to evaluate the performance of *Constrain-based Learning* (referred as RepBin-*Learning*). We use the Precision, Recall, and F1 scores to evaluate the performance of binning contigs. As existing binning tools typically discard many short contigs and thus cannot bin all contigs, ARI and NMI are not included to avoid possible biases towards binned contigs. We also report the number of bins identified by binning tools.

For RepBin, we use the adjacency matrix as the initial features for nodes. The representation dimensions are all empirically set to be 32. For all baseline methods, we optimize their models with different parameters and report the best performance scores. The RepBin model is freely available[4].

[1]https://www.ncbi.nlm.nih.gov/sra/?term=SRA052203

[2]https://data.cami-challenge.org/participate

[3]https://ggkbase.berkeley.edu/carrol/organisms

[4]https://github.com/xuehansheng/RepBin

Benchmarking against Machine Learning Models

Table 2 shows that Both RepBin-*Learnig* and RepBin significantly outperform machine learning-based baselines, which achieve the highest scores on all metrics and all three datasets. For Sim-20G, the metric scores obtained by RepBin-*Learning* are 91.8% for F1, 84.8% for ARI, and 92.1% for NMI respectively which is significantly higher than the highest scores achieved by GNNs (79.8% for F1, 65.4% for ARI, and 83.6% for NMI). Both GNNs and RepBin-*Learning* run K-Means algorithm to cluster nodes after obtaining representations for nodes. The gap between GNNs and RepBin-*Learning* demonstrates the superiority of our model in learning structure of graph and integrating prior constraints. Because of the label imbalance, graph clustering (GCs) methods also achieve lower metric scores than RepBin. The results of two constraint-based clustering algorithms are also inferior to RepBin, because these methods cannot capture structural information of the graph well.

Benchmarking against Metagenomic Binning Tools

Table 3 shows the results obtained by RepBin and stand-alone baselines on all five datasets. We also compare RepBin with refined binning results of baselines with GraphBin on Sim-20G and CAMI datasets.

Table 3 shows that RepBin significantly outperforms baselines including GraphBin refinements. RepBin achieves the highest Recall and F1 score on all simulated and real-world datasets. Take Sim-20G for example, the highest Recall and F1 score achieved among all baselines is SolidBin, 85.04% and 90.41%, which is much lower than the scores achieved by RepBin (96.98% for Recall and 97.15% for F1). The Precision score achieved by RepBin is 97.31%, which is slightly lower than VAMB (99.35%) and significantly higher than other baselines. In the publicly-available CAMI dataset, RepBin also achieves the highest Recall and F1 score (92.84% and 87.41%) against other baselines (the second highest score achieved by BusyBeeWeb, (53.15% for Recall and 63.05 % for F1). The significant improvement achieved by RepBin on the F1 score indicates that RepBin can tackle the label imbalance problem well and accurately bin as many contigs as possible. Moreover, Table 3 shows that RepBin also achieves better performance against the refined binning results of baselines+GraphBin. GraphBin is not a stand-alone binning tool. We have to first run one existing binning tool (e.g., MetaWatt) to derive the initial binning results and then run GraphBin to improve the initial binning results. Although GraphBin improves the binning results on all stand-alone baselines, the final evaluation matrices are still inferior to RepBin.

Visualization. To better understand the binning results, we use python-iGraph package to visualize the Sim-10G assembly graph with ground truth and binning results from distinct stand-alone binning tools (see Figure 3). Different colors denote distinct species and grey nodes indicate the nodes that are not identified or are discarded. Black edges represent homophily edges in the assembly graph and grey edges are heterophily constraints. RepBin achieves the most consistent labels with respect to the ground truth while other baselines suffering from missing or incorrect labels.

Methods		Sim-5G			Sim-10G			Sim-20G		
		F1	ARI	NMI	F1	ARI	NMI	F1	ARI	NMI
GNNs	GSAGE	88.0±0.6	72.9±0.9	81.6±0.8	76.6±0.2	59.3±0.7	75.9±0.6	77.4±0.6	61.5±1.8	81.5±0.3
	GAT	94.5±1.6	86.9±1.7	87.7±0.7	73.7±0.5	54.0±2.6	74.3±0.4	79.8±1.0	65.4±1.3	83.6±0.7
	DGI	79.9±3.4	54.6±6.8	68.2±3.3	68.1±1.9	39.3±2.4	61.8±2.4	63.9±2.5	40.1±2.3	65.8±2.5
	VGAE	85.7±1.4	70.1±2.6	80.1±3.2	71.4±1.9	46.0±2.6	68.9±1.1	72.2±1.7	54.8±2.1	75.9±1.2
RepBin-*Learning*		99.8±0.0	99.4±0.0	99.2±0.0	97.1±0.8	95.4±1.3	96.3±0.8	91.8±0.5	84.8±0.6	92.1±0.2
GCs	O2MAC	74.9±3.5	63.6±2.1	72.5±3.1	65.8±1.4	53.5±2.1	69.1±1.7	61.0±3.4	52.0±2.7	71.1±1.8
	AGC	80.9±0.4	92.7±0.8	90.4±0.5	78.3±0.3	87.9±0.3	89.6±0.7	67.0±0.2	75.9±1.1	82.0±0.5
	CSC	96.7±0.0	87.9±0.0	91.5±0.0	90.9±1.3	77.9±4.2	85.1±1.7	83.0±1.4	63.2±4.3	83.1±1.1
	DCC	90.9±0.0	94.0±1.0	88.6±1.1	92.1±2.9	83.3±3.0	77.3±2.3	82.1±1.9	65.3±2.8	75.1±3.3
RepBin		99.7±0.1	99.0±0.2	98.5±0.0	99.4±0.0	99.1±0.1	98.8±0.3	97.2±0.6	94.3±0.8	95.7±0.6

Table 2: The results of RepBin and machine leaning-based baselines on three simulated datasets for contigs binning (%).

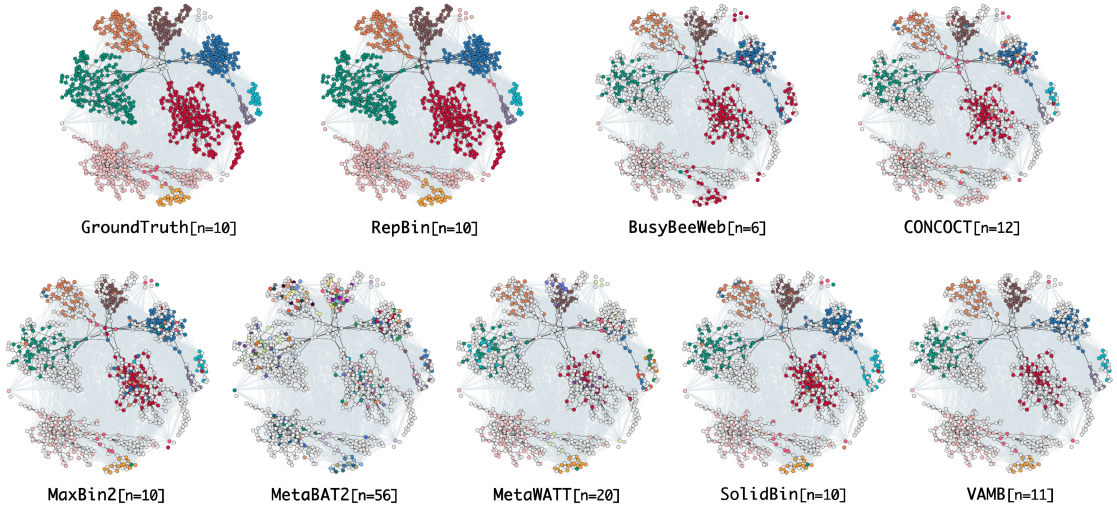

Figure 3: Visualization of the Sim-10G assembly graph with ground truth and different binning results.

Ablation Study

Effects of Constraints in Learning and Binning. To study the effectiveness of incorporating constraints of our model, we conduct an ablation study by examining variants of RepBin. *RepBin without constraint-based learning* represents RepBin discards the information of constraints in the Learning part and only captures the structure of the assembly graph. *RepBin without constraint-based binning* denotes RepBin without constraint-based binning part (refer to the *Constraint-based Learning* model) and uses K-Means algorithm to obtain final binning results. Figure 4 shows that modelling constraints in RepBin improves metagenomic binning. For CAMI, the F1 values achieved by RepBin, *RepBin without constraint-based learning*, *RepBin without constraint-based binning* are 87.41%, 82.10%, and 77.84% respectively, which demonstrate the effectiveness of constraint-based learning and binning.

Besides, we also use t-SNE (van der Maaten and Hinton 2008) to visualize the latent embeddings of *RepBin*. Figure 5 shows the 2D-visualization of the embeddings from the CAMI dataset, where nodes with distinct colors represent different species. Visually, *RepBin-Learning with Constraint* gives the best separation between different clusters

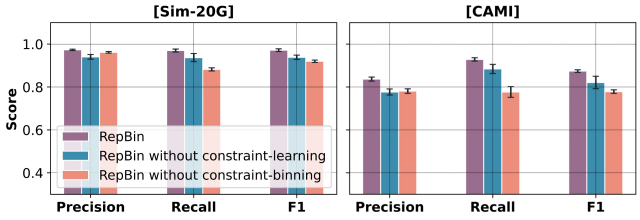

Figure 4: Results of RepBin and its variants on Sim-20G and CAMI datasets.

comparing with *RepBin-Learning without Constraint*.

Parameters Analysis. The parameter α, varying from 0.001 to 0.1, is used in diffusion to control the probability of teleporting to another state. The parameter λ, varying from 0.1 to 0.9, is used to balance the importance between the graph and constraints in the loss function of the *Constraint-based Learning* model. The parameter d, varying in $\{16, 32, 64, 128\}$, represents the dimension of the RepBin-*Learning*. Figure 6 shows that the performance of RepBin is not sensitive to the changes in above parameters.

Datasets		MetaWatt	CON COCT	MaxBin2	BusyBee Web	MetaBAT2	SolidBin	VAMB	RepBin
Sim-5G	Precision	100.00	91.60	91.13	86.57	100.00	90.00	100.00±0.00	99.69±0.10
	Recall	24.59	40.50	46.69	49.79	6.61	46.49	33.92±0.90	99.69±0.10
	F1	39.47	56.16	61.75	63.22	12.40	61.31	50.66±1.02	99.69±0.10
	Pred. bins	12	7	5	4	34	5	6	5
Sim-10G	Precision	99.29	86.99	89.43	84.47	100.00	91.58	99.93±0.15	99.20±0.00
	Recall	26.13	39.72	40.30	45.53	6.39	41.70	33.80±0.20	99.55±0.08
	F1	41.38	54.54	55.56	59.17	12.01	57.30	50.51±0.23	99.37±0.04
	Pred. bins	20	12	10	6	56	10	11	10
Sim-20G	Precision	96.85	84.02	88.25	77.39	96.77	96.51	99.35±0.10	97.31±0.31
	Recall	32.01	42.27	41.69	44.51	7.73	85.04	36.88±0.60	96.98±0.69
	F1	48.12	56.24	56.63	56.52	14.32	90.41	53.79±0.64	97.15±0.61
	Pred. bins	33	22	21	12	88	20	22	20
CAMI	Precision	86.29	92.83	85.33	77.47	95.41	80.65	98.82±0.41	83.67±0.93
	Recall	37.12	43.65	39.69	53.15	1.51	44.86	31.90±0.92	92.84±0.78
	F1	51.91	59.38	54.18	63.05	2.97	57.66	48.23±1.06	87.41±0.60
	Pred. bins	65	30	22	16	101	20	21	17
Sharon	Precision	79.08	74.03	82.33	68.68	76.89	70.04	97.41±0.21	73.60±0.72
	Recall	18.91	24.58	25.74	43.82	1.72	25.82	30.23±2.51	76.59±0.79
	F1	30.52	36.91	40.37	53.50	3.36	37.73	46.10±3.01	74.97±0.16
	Pred. bins	39	48	16	5	34	9	9	11
Stand-alone Binning Tools + GraphBin									
Sim-20G	Precision	98.02	92.96	96.77	90.27	96.77	96.51	98.15±0.04	97.31±0.31
	Recall	68.71	81.86	83.96	91.84	7.73	85.04	86.35±2.21	96.98±0.69
	F1	80.79	87.06	89.91	91.05	14.32	90.41	91.86±1.25	97.15±0.61
	Pred. bins	33	22	21	12	88	20	22	20
CAMI	Precision	91.88	96.44	89.18	85.16	92.44	86.86	95.05±0.26	83.67±0.93
	Recall	60.12	79.43	76.82	88.17	36.77	82.15	78.94±1.47	92.84±0.78
	F1	72.68	87.11	82.54	86.85	52.60	84.44	86.24±0.91	87.41±0.60
	Pred. bins	65	30	22	16	101	20	21	17

Table 3: The experimental metrics of RepBin and baselines on five datasets for binning contigs (%).

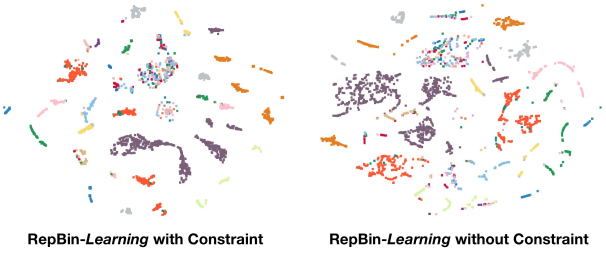

Figure 5: Visualization of representations on CAMI.

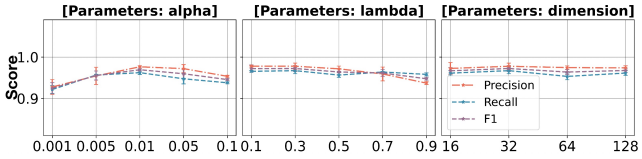

Figure 6: Parameters analysis of the RepBin model.

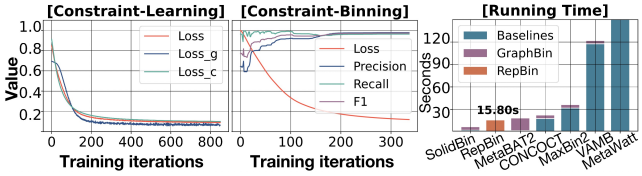

Figure 7: The training loss and running time of RepBin.

Training & Running Analysis. Figure 7 (a) and (b) show the process of optimizing Equation 3 and the changes of evaluation metrics in the *Constraint-based Binning* model of the RepBin. Figure 7 (c) shows the running time of RepBin against other baselines on Sim-20G. RepBin is the second fastest binning tool and only slightly slower than SolidBin.

Conclusion

To learn both the graph structure and prior heterophily information (represented as pairwise constraints), we propose a constraint-based graph representation learning model, *Constraint-based Learning*. It is a contrastive graph learning framework that uses a diffusion graph convolutional operator to model graph structure and respects prior constraints by adding a penalty with respect to constrained nodes. We also propose a *Constraint-based Binning* model which groups constrained nodes using k-means algorithm and then use a semi-supervised GCN model to annotate unlabeled nodes, which is robust to imbalance in the node constraints and the bin sizes. The proposed *RepBin* is implemented to solve the real-world task of metagenomic binning. Extensive experiments demonstrate the superiority of RepBin in binning.

References

Albertsen, M.; Hugenholtz, P.; Skarshewski, A.; Nielsen, K. L.; Tyson, G. W.; and Nielsen, P. H. 2013. Genome sequences of rare, uncultured bacteria obtained by differential coverage binning of multiple metagenomes. *Nature biotechnology*, 31(6): 533–538.

Alneberg, J.; and et al. 2014. Binning metagenomic contigs by coverage and composition. *Nature methods*, 1144–1146.

Asnicar, F.; Berry, S. E.; Valdes, A. M.; Nguyen, L. H.; Piccinno, G.; Drew, D. A.; Leeming, E.; Gibson, R.; Le Roy, C.; Al Khatib, H.; et al. 2021. Microbiome connections with host metabolism and habitual diet from 1,098 deeply phenotyped individuals. *Nature Medicine*, 27(2): 321–332.

Barnum, T. P.; Figueroa, I. A.; Carlström, C. I.; Lucas, L. N.; Engelbrektson, A. L.; and Coates, J. D. 2018. Genome-resolved metagenomics identifies genetic mobility, metabolic interactions, and unexpected diversity in perchlorate-reducing communities. *The ISME journal*, 12(6): 1568–1581.

Bianchi, F. M.; Grattarola, D.; and Alippi, C. 2020. Spectral Clustering with Graph Neural Networks for Graph Pooling. In *Proceedings of the 37th international conference on Machine learning*, 2729–2738. ACM.

Cao, S.; Lu, W.; and Xu, Q. 2016. Deep neural networks for learning graph representations. In *AAAI*.

Cui, P.; Wang, X.; Pei, J.; and Zhu, W. 2019. A Survey on Network Embedding. *TKDE*, 31: 833–852.

Eddy, S. R. 2011. Accelerated profile HMM searches. *PLoS computational biology*, 7(10): e1002195.

Fan, S.; Wang, X.; Shi, C.; Lu, E.; Lin, K.; and Wang, B. 2020. One2Multi Graph Autoencoder for Multi-View Graph Clustering. WWW '20, 3070–3076. Association for Computing Machinery.

Friedman, J.; Hastie, T.; Tibshirani, R.; et al. 2001. *The elements of statistical learning*, volume 1. Springer series in statistics New York.

Grover, A.; and Leskovec, J. 2016. Node2Vec: Scalable Feature Learning for Networks. In *KDD*.

Hamilton, W. L.; Ying, Z.; and Leskovec, J. 2017. Inductive Representation Learning on Large Graphs. In *NeurIPS*.

Integrative, H.; Proctor, L. M.; Creasy, H. H.; Fettweis, J. M.; Lloyd-Price, J.; Mahurkar, A.; Zhou, W.; Buck, G. A.; Snyder, M. P.; Strauss III, J. F.; et al. 2019. The integrative human microbiome project. *Nature*, 569(7758): 641–648.

Kaeberlein, T.; Lewis, K.; and Epstein, S. S. 2002. Isolating" uncultivable" microorganisms in pure culture in a simulated natural environment. *Science*, 296(5570): 1127–1129.

Kang, D. D.; and et al. 2019. MetaBAT 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies. *PeerJ*, 7: e7359.

Kipf, T. N.; and Welling, M. 2016. Variational Graph Auto-Encoders. In *NeurIPS Workshop on Bayesian Deep Learning*.

Klicpera, J.; and et al. 2019a. Diffusion Improves Graph Learning. In *NeurIPS*.

Klicpera, J.; and et al. 2019b. Predict then Propagate: Graph Neural Networks meet Personalized PageRank. In *ICLR*.

Laczny, C. C.; Kiefer, C.; Galata, V.; Fehlmann, T.; Backes, C.; and Keller, A. 2017. BusyBee Web: metagenomic data analysis by bootstrapped supervised binning and annotation. *Nucleic Acids Research*, W171–W179.

Liu, X.; Murata, T.; Kim, K.-S.; Kotarasu, C.; and Zhuang, C. 2019. A general view for network embedding as matrix factorization. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 375–383.

Mallawaarachchi, V.; and et al. 2020a. GraphBin: refined binning of metagenomic contigs using assembly graphs. *Bioinformatics*, 36(11): 3307–3313.

Mallawaarachchi, V. G.; and et al. 2020b. GraphBin2: Refined and Overlapped Binning of Metagenomic Contigs Using Assembly Graphs. In *WABI*.

Nayfach, S.; Shi, Z. J.; Seshadri, R.; Pollard, K. S.; and Kyrpides, N. C. 2019. New insights from uncultivated genomes of the global human gut microbiome. *Nature*, 568(7753): 505–510.

Nissen, J. N.; and et al. 2021. Improved metagenome binning and assembly using deep variational autoencoders. *Nature biotechnology*, 39(5): 555–560.

Nurk, S.; Meleshko, D.; Korobeynikov, A.; and Pevzner, P. A. 2017. metaSPAdes: a new versatile metagenomic assembler. *Genome research*, 27(5): 824–834.

Page, L.; Brin, S.; Motwani, R.; and Winograd, T. 1999. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Park, C.; Kim, D.; Han, J.; and Yu, H. 2020. Unsupervised Attributed Multiplex Network Embedding. In *AAAI*.

Pasolli, E.; De Filippis, F.; Mauriello, I. E.; Cumbo, F.; Walsh, A. M.; Leech, J.; Cotter, P. D.; Segata, N.; and Ercolini, D. 2020. Large-scale genome-wide analysis links lactic acid bacteria from food with the gut microbiome. *Nature communications*, 11(1): 1–12.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. DeepWalk: Online Learning of Social Representations. In *KDD*.

Qiu, J.; Dong, Y.; Ma, H.; Li, J.; Wang, K.; and Tang, J. 2018. Network Embedding as Matrix Factorization: Unifying DeepWalk, LINE, PTE, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 459–467. ACM.

Rho, M.; Tang, H.; and Ye, Y. 2010. FragGeneScan: predicting genes in short and error-prone reads. *Nucleic acids research*, 38(20): e191–e191.

Sczyrba, A.; Hofmann, P.; and et al. 2017. Critical Assessment of Metagenome Interpretation-a benchmark of metagenomics software. *Nature methods*, 1063—1071.

Sharon, I.; Morowitz, M. J.; and et al. 2013. Time series community genomics analysis reveals rapid shifts in bacterial species, strains, and phage during infant gut colonization. *Genome research*, 111—120.

Strous, M.; Kraft, B.; Bisdorf, R.; and Tegetmeyer, H. 2012. The Binning of Metagenomic Contigs for Microbial Physiology of Mixed Cultures. *Frontiers in Microbiology*, 3: 410.

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

Turnbaugh, P. J.; Ley, R. E.; Hamady, M.; Fraser-Liggett, C. M.; Knight, R.; and Gordon, J. I. 2007. The human microbiome project. *Nature*, 449(7164): 804–810.

van der Maaten, L.; and Hinton, G. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86): 2579–2605.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *ICLR*.

Veličković, P.; Fedus, W.; Hamilton, W. L.; Liò, P.; Bengio, Y.; and Hjelm, R. D. 2019. Deep Graph Infomax. In *ICLR*.

Von Luxburg, U. 2007. A tutorial on spectral clustering. *Statistics and computing*, 17(4): 395–416.

Wang, X.; Qian, B.; and Davidson, I. 2014. On constrained spectral clustering and its applications. *Data Mining and Knowledge Discovery*, 28(1): 1–30.

Wang, Z.; Wang, Z.; Lu, Y. Y.; Sun, F.; and Zhu, S. 2019. SolidBin: improving metagenome binning with semi-supervised normalized cut. *Bioinformatics*, 35(21): 4229–4238.

Wu, Y.-W.; and et al. 2015. MaxBin 2.0: an automated binning algorithm to recover genomes from multiple metagenomic datasets. *Bioinformatics*, 605–607.

Wu, Y.-W.; Tang, Y.-H.; Tringe, S. G.; Simmons, B. A.; and Singer, S. W. 2014. MaxBin: an automated binning method to recover individual genomes from metagenomes using an expectation-maximization algorithm. *Microbiome*.

Zhang, H.; Basu, S.; and Davidson, I. 2019. A framework for deep constrained clustering–algorithms and advances.

Zhang, X.; Liu, H.; Li, Q.; and Wu, X.-M. 2019. Attributed Graph Clustering via Adaptive Graph Convolution. In *IJCAI-19*.