IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 5, 2022, accepted May 18, 2022, date of publication May 23, 2022, date of current version May 26, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3176954

A Deep Embedded Clustering Algorithm for the Binning of Metagenomic Sequences

HUYNH QUANG BAO[1,2], LE VAN VINH[3], AND TRAN VAN HOAI[1,2]

[1]Faculty of Computer Science and Engineering, Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh City 70000, Vietnam
[2]Vietnam National University Ho Chi Minh City, Ho Chi Minh City 70000, Vietnam
[3]Faculty of Information Technology, HCMC University of Technology and Education, Ho Chi Minh City 70000, Vietnam

Corresponding author: Le Van Vinh (vinhlv@hcmute.edu.vn)

This work was supported by the Vietnam National University Ho Chi Minh City (VNU-HCM), under Grant B2019-20-06.

ABSTRACT The study of metagenomic sequences brings a deep understanding of microbial communities. One of the crucial steps in metagenomic projects is to classify sequences into different organisms, named the binning problem. In the emerging methods for classification, deep learning is a potential technology to be applicable with high accuracy. However, it is well-known that reference databases, which are highly required by deep learning based methods, are not always available. As a result, some existing binning solutions have applied unsupervised learning processes, but utilizing the strength of deep learning in an unsupervised model is still a challenging problem. This work proposes a binning algorithm for metagenomic sequences, called MetaDEC, which applies a deep unsupervised learning approach. By following the two-phase paradigm, the algorithm firstly divides sequences into groups of overlapping sequences. The groups are then classified into clusters using an adversarial deep embedded clustering technique. Experimental results show that MetaDEC achieves competitive performance compared to existing methods on both simulated and real metagenomic data.

INDEX TERMS Algorithm, clustering, deep learning, metagenomics, DNA sequence.

I. INTRODUCTION

Metagenomics is the study of genetic content collected directly from the environment, bypassing the need for culturing and isolating in laboratories. The field offers opportunities to get a deep inside into microbial communities that are infeasible with traditional methods of single-genome sequencing technologies. There are many fields of applications from metagenomic studies such as biomedical science, biotechnology, energy, and agriculture [1].

Some initial metagenomic projects, e.g. Acid Mine Drainage, and Sargasso Sea are based on Sanger sequencing technology [2], [3]. The technology produces quite long sequences which provide meaningful information for the binning process. Unfortunately, because it requires high costs and is very time-consuming, the technology is impractical for most current projects, which require analyzing a huge amount of data. Thanks to the development of Next-Generation Sequencing approaches such as Illumina, and 454 pyrosequencing [4] which are appropriate for nowadays metagenomic projects. The technologies are able to process millions of reads per run in a short time with low costs. However, one of the limits of the sequencing technologies is that they produce sequences with short lengths, and thus brings research challenges for communities because of the lack of information in short sequences.

Due to the fact that DNA fragments in metagenomic samples are from multiple genomes, one of the major steps in metagenomic projects is to classify sequences into groups of closely related genomes. The step is referred to as *binning* problem. Binning results are beneficial to reconstructing genomes by assembly approaches or used for analyzing sequences directly from every single genome.

Some binning approaches such as TIPP2 [5], and mOTUs2 [6] use marker genes, e.g. 16S rRNA, recA for profiling metagenomic sequences. The strength of those methods is that they require low computing costs because of not needing to analyze the whole genomes of organisms. However, the classification quality of the methods is affected by the fact that many different species contain the same marker genes, and others may have a small ratio of 16S rRNA genes [7], [8]. This work focuses on whole-genome sequencing approaches which analyzing all genome sequences of microbial organisms. Those approaches

The associate editor coordinating the review of this manuscript and approving it for publication was Wentao Fan.

can be classified into groups of *supervised*, and *unsupervised* methods.

Supervised methods are based on homology or composition information from an available reference database to support the classification process. MEGAN CE [9], and MEGAN-LR [10] use homology search tools such as DIAMOND [11], or LAST [12] to determine the similarity between input sequences with reference sequences. The algorithms then assign the sequences into groups of known organisms. One of the drawbacks of the methods is that they are very time-consuming. Kraken2 [13] deal with the challenge by extracting long k-mers from sequences and comparing them with reference databases. On the other hand, DeepMicrobes [14] is composition-based supervised methods that utilizes genomic signatures extracted from sequences to classify the metagenomic data. The method applies a deep learning-based computational framework for its classification task.

Due to the limitation on reference databases, some binning algorithms apply an unsupervised learning process for classifying sequences. MetaCluster 2.0 [15] and MetaCRS [16] aim to cluster metagenomic data of long sequences or contigs. Both of the algorithms use composition features and apply k-means algorithm in their process. Focusing on analyzing short sequences, MetaCluter 5.0 [17] applies a two-round binning approach that classifies effectively samples with low-abundance species. Another binning algorithm for short reads, AbundanceBin [18] is only based on abundance levels of species in metagenome, and thus it does not work well for samples of species with similar abundance levels.BiMeta [19] and MetaProb [20] are other clustering algorithms that utilize sequence overlapping information and k-mer frequency extracted from groups of reads. While BiMeta uses Euclidean distance of k-mer frequencies between data points in its second phase, MetaProb applies probabilistic signatures to get better classification quality. MetaProb 2 [21] is an improvement of MetaProb which requires an assembly task on the group of reads and a graph clustering algorithm for classifying sequences.

Recent studies on unsupervised deep learning-based clustering approach achieve significant improvement [22], [23] and are applied in a few metagenomic binning algorithms. Among them, the binning algorithm of Isis *et al* [24] uses an autoencoder architecture to compress composition-based representation of sequences and applied k-Means++ method on compressed representation to classify data. On the other hand, VAMB algorithm [25] only focuses on analyzing metagenomic contigs. It uses a variational autoencoder architecture to compress data into a latent posterior distribution (multivariate Gaussian distribution) in the clustering process.

This study proposes a novel unsupervised binning method for metagenomic sequences called MetaDEC (i.e., **Meta**genomic binning with **D**eep **E**mbedded **C**lustering). The proposed approach follows the paradigm of two phases as used by previous studies [19], [20] in which the first phase does preprocessing works to groups sequences using the sequence overlapping information. However, different from previous works, MetaDEC then uses ADEC [23] - a novel unsupervised deep learning approach - to classify the sequence groups in its second phase. The method utilizes an autoencoder architecture with latent space interpolation factor to learn the underlying representation of data and optimize the learned representation toward the clustering objective.

The next section presents the details of the proposed method. The Experiments and Results section shows the strength of MetaDEC comparing with available binning approaches. Some conclusions are presented in the final section.

II. METHODS

In order to overcome the lack of phylogenetic information in short sequences, the proposed method applies a two-phase paradigm in the clustering process (Fig. 1). Phase 1 does preprocessing work to build groups of sequences using overlapping information between them. Clustering features are then extracted from the sequence groups. Phase 2 uses the features to continue classifying the groups into clusters of close organisms.

A. PHASE 1: GROUPING SEQUENCES AND BUILDING SEEDS

Derived from the work in [19], the phase classifies reads that share sufficient long l-mer into the same groups and builds group representative. Based on an observation that the l-mers are unique in genomes [17], [26]. MetaDEC firstly builds a graph whose vertices are reads, and each edge is the connection between two reads that have sufficient substring overlapping. While previous methods [19], [20] used a greedy approach to build groups, this work applies a multilevel partitioning algorithm [27] to create a coarsened graph and acquires groups from connected components.

An observation in [19] revealed that genomics signatures of k-mer nucleotide frequency are also preserved in a group of non-overlapping short reads as in long sequences. Thus, MetaDEC selects a subgroup of non-overlapping reads in each group, called seed, as a representative of the group. The technique does not only reduce noises in features extracted from sequence groups that have unbalanced coverage but also saves computation costs [19]. Next, k-mer frequency distribution of each seed is computed as follows.

Given $S = \{r_1, r_2, \ldots r_n\}$ is a seed, where n is the number of reads in seed S. Let $|r_i|$, $i \in [1..n]$ be the length of read r_i. In order to find k-mers of each read, MetaDEC uses a sliding window method with a window size of k. There is $|r_i|-k+1$ k-mers in each read. Thus, the total number of k-mer of seed S, denoted $|S|$, is $\sum_{i=1}^{n}(|r_i|-k+1)$.

In other words, there are at most 4^k different contents of k-mers because each k-mer is a combination of 4 kinds of nucleotides (A, T, G, C corresponding to Adenine, Thymine, Cytosine, Guanine, respectively). However, either of the DNA strands can be obtained from their reversed complement

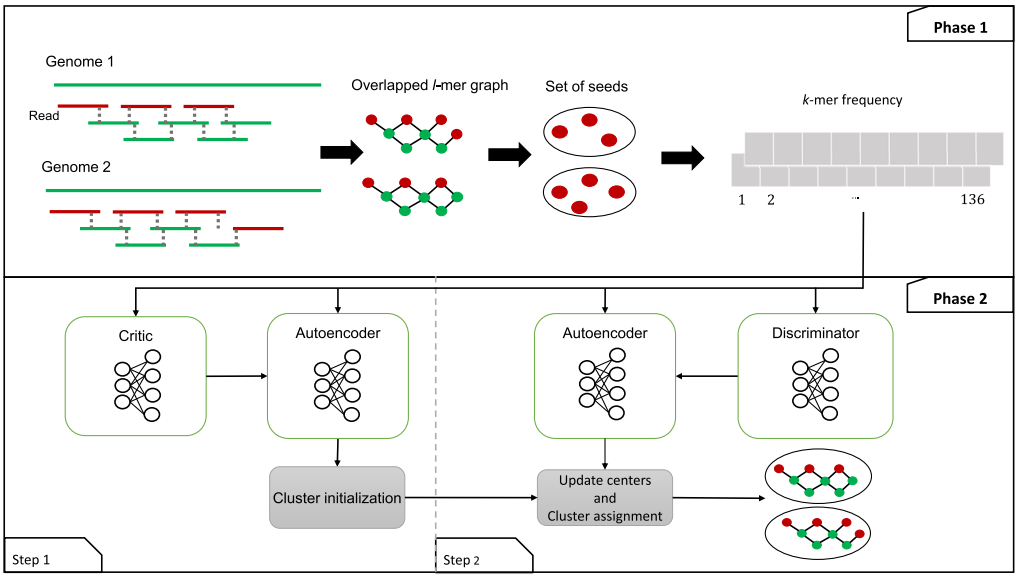


FIGURE 1. Process of MetaDEC. Phase 1: Grouping sequences and building seeds. Phase 2: Classifying sequence groups using deep clustering.

strands (e.g., ATTT - TAAA, GCGC - CGCG). The frequency of a k-mer and its reversed complement are the same. As a result, the number of total k-mer values could be reduced by half, from 4^k to $4^k/2$ if k is odd, $(4^k + 4^{k/2})/2$ if k is even. Studies in [15], [28], and [19] showed that value $k = 4$ in computing k-mer frequency is the best choice for extracting compositional features from DNA sequences or contigs. Thus, we also choose $k = 4$ in this work. It means that there are 136 k-mers in total.

Let $f^S = \{f_1^S, f_2^S, \ldots, f_{136}^S\}$ is the set of k-mer frequency representation of seed S. f^S is normalized by dividing each entry by $|S|$. It is then normalized a second time into a normal distribution x^S with mean $\mu = 0$ and variance $\sigma = 1$. As a result, the final representation of seed S is $x^S = \{x_1^S, x_2^S, \ldots, x_{136}^S\}$, in which each x_i^S is in the range of $[-1, 1]$.

B. PHASE 2: CLASSIFYING SEQUENCE GROUPS USING DEEP CLUSTERING

Given n groups of sequences which are represented by a set of n normalized frequencies of seeds constructed in phase 1 $X = \{x^{S_1}, x^{S_2}, \ldots, x^{S_n}\}$, in which $S_i, i \in [1..n]$ be seeds of the groups. In this phase, MetaDEC classifies X into m clusters with m centroids $C = \{c_1, c_2, \ldots c_m\}$ applying a deep clustering method called ADEC (Adversarial Deep Embedded Clustering) [23]. The phase consists of two steps: Cluster initialization, and Clustering optimization.

1) STEP 1: CLUSTERS INITIALIZATION

The first step aims to find a sufficient initialization of clusters (presented in Fig. 1 and Algorithm 1). Instead of using similarity measurement directly on X, the proposed method uses an autoencoder to learn the latent space representation of X. In detail, the autoencoder composes of two parts, encoder, and decoder. The encoder is a non-linear mapping function $E_\theta : X \longrightarrow Z$ that transforms X space into a smaller dimensionality latent feature space $Z = \{z^{S_1}, z^{S_2}, \ldots, z^{S_n}\}$, in which each z^{S_i} corresponds to encoded $x^{S_i}, i \in [1..n]$. The decoder is another non-linear mapping function $D_\phi : Z \longrightarrow \hat{X}$ that transforms latent space Z back to the original data space of k-mer frequency representation. In our work, both encoder and decoder are modeled by neural networks. MetaDEC also adds interpolation factor on latent space based on a framework proposed from ACAI [29] by using another neural network called *critic* C_ψ to improve the quality of latent space. The information from the *critic* is used as a regularization term in the autoencoder optimization process.

In this step, autoencoder (*encoder* E_θ, *decoder* D_ϕ) and *critic* C_ψ are trained in predefined iterations. Each iteration updates C_ψ using a loss function L_C (1) and E_θ, D_ϕ using a loss function $L_{E,D}$ (2).

$$L_C = ||C_\psi(\hat{x}_\alpha^S) - \alpha||^2 + ||C_\psi(\gamma x^{S_i} + (1-\gamma)D_\phi(E_\theta(x^{S_i}))||^2 \quad (1)$$

$$L_{E,D} = ||x^{S_i} - D_\phi(E_\theta(x^{S_i}))||_2^2 + \lambda||C_\psi(\hat{x}_\alpha^S)||^2 \quad (2)$$

in which $\hat{x}_\alpha^S = D_\phi(\alpha E_\theta(x^{S_1}) + (1-\alpha)E_\theta(x^{S_2}))$. x^{S_1}, x^{S_2} are randomly picked data points in X and α, λ are randomly sampled in range [0, 1]. It is noted that instead of using square root function for each term in L_C and $L_{E,D}$ as the work in [23], MetaDEC uses square function for a better model training.

After the optimization of the autoencoder and critic, latent space $Z = E_\theta(X)$ is computed, and centroids C are initialized by applying k-means algorithm on Z.

2) STEP 2: CLUSTERING OPTIMIZATION

In this step, MetaDEC continues optimizing the clustering result from step 1. The process performs two sub-steps

iteratively, which computes soft clustering assignment and learns from high confidence assignment. It runs until convergence, or a threshold met.

Firstly, soft cluster assignment for latent space Z is computed. This study uses the Student's t-distribution in calculating the distance from z^{S_i}, $i \in [1..n]$ and cluster's centroid c_j, $j \in [1..m]$. The probability $q_{ij} \in Q$ that data point i^{th} belongs to cluster j^{th} is computed as the following formulation.

Algorithm 1: Cluster Initialization

Input: Input groups: X; max iterations for pretraining autoencoder and critic: *max_iter_ae*; learning rate: δ; number of clusters: m.

Output: Clusters' centroids: $C = \{c_1, c_2, \ldots c_m\}$

Initialize $\theta, \phi, \psi, \omega$;

// Pretraining autoencoder and critic

1: **for** $i \leftarrow 1$ to *max_iter_ae* **do**

2: Compute $L_C, L_{E,D}$ using (1) and (2)

// Update parameter

3: $\theta \leftarrow \theta - \delta \Delta L_{E,D}$; $\phi \leftarrow \phi - \delta \Delta L_{E,D}$;

4: $\psi \leftarrow \psi - \delta \Delta L_C$

5: **end for**

// Initialize clusters' centroids

6: $Z = E_\theta(X)$

7: Initialize C by applying k-means algorithm on Z

$$q_{ij} = \frac{(1 + \|z_i - c_j\|^2/\eta)^{-\frac{\eta+1}{2}}}{\sum_{j'=1}^{m}(1 + \|z_i - c_{j'}\|^2/\eta)^{-\frac{\eta+1}{2}}} \tag{3}$$

in which, $z^{S_i} = E_\theta(x^{S_i}) \in Z$, η is degree of freedom in t-student distribution. Assignment of data point i is computed as $y^i_{pred} = argmax_j(q_{ij})$.

Secondly, auxiliary target distribution $P = p_{ij}$, $i \in [1..n], j \in [1..m]$, is computed by deriving from high confidence assignment as follows.

$$p_{ij} = \frac{q^2_{ij}/freq_j}{\sum_{j'=1}^{m} q^2_{ij'}/freq_{j'}} \tag{4}$$

in which $freq_{j'} = \Sigma_i q_{ij'}$ is the frequency of soft assignment. An objective here is to minimize the difference between soft cluster assignment distribution Q and auxiliary target distribution P. MetaDEC then minimizes Kullback-Leibler divergence (KLD) between P and Q to optimize E_θ and clusters' centroids $c_i \in C$.

Beside optimizing KLD loss function, MetaDEC adds a neural network into the clustering optimization step, called *discriminator* G_ω, to help improve the quality of latent space produced by encoder E_θ. The discriminator network is optimized to be able to distinguish real groups and reconstructed groups by minimizing loss function L_G (equation 5). The information produced by discriminator G_ω is used as a regularization term in optimizing E_θ and clusters' centroids C. The regularization term penalizes the encoder when it generates meaningless latent space for decoding. Equation 6 describes the final loss function L_{cls} used for optimizing E_θ and clusters' centroids C.

The proposed method keeps optimizing decoder D_ϕ in the training process. The decoder in this step plays a role as a monitor for assuring the quality of the latent space. Thus, it needs to be trained to catch up with the changes from encoder E_θ by optimizing reconstruction loss L_D in (7).

$$\begin{aligned} L_G &= \mathbb{E}_{x\sim p(x)}[log(G_\omega(x)] \\ &\quad + \mathbb{E}_{x\sim p(x)}[log(1 - G_\omega(D_\phi(E_\theta(x))))] \end{aligned} \tag{5}$$

$$\begin{aligned} L_{cls} &= KLD(P||Q) \\ &\quad + \mathbb{E}_{x\sim p(x)}[log(1 - G_\omega(D_\phi(E_\theta(x))))] \end{aligned} \tag{6}$$

$$L_D = ||x - D_\phi(x)||_2^2 \tag{7}$$

Algorithm 2 presents the details of clustering optimization step. It firstly does pretraining discriminator before finding assignment results for elements in X to clusters.

C. PERFORMANCE METRICS

The quality performance of binning algorithms are evaluated using three metrics *precision*, *recall* and *F-measure* (also used in [17], [20], and [19]). While *precision* measures the ratio of reads classified into a cluster that comes from the same species, *recall* presents the percentage of reads that belong to the same species are clustered into the same clusters. Because each of the two metrics cannot fully reflect the performance of a binning algorithm, *F-measure* is used as a harmonic metric between them. The three metrics are defined as follows.

$$precision = \frac{\sum_{i=1}^{k} max_j A_{ij}}{\sum_{i=1}^{k}\sum_{j=1}^{m} A_{ij}} \tag{8}$$

$$recall = \frac{\sum_{j=1}^{m} max_i A_{ij}}{\sum_{i=1}^{k}\sum_{j=1}^{m} A_{ij} + \neq \text{unassigned reads}} \tag{9}$$

$$F - measure = 2 * \frac{precision * recall}{precision + recall} \tag{10}$$

in which A_{ij} is the number of reads that come from species j classified into cluster i, m is the number of species in the metagenome, and k is the number of clusters revealed by binning algorithms.

III. EXPERIMENTS AND RESULTS

A. DATASETS

The proposed method is tested on 25 simulated datasets which were used in previous studies [19]. Among them, there are 16 datasets following Illumina sequencing technology, named from S1 to S10, and from L1 to L6, which are paired-end short reads with the length of 80bp. Datasets

Algorithm 2: Cluster Optimization

Input: Input groups: X; maximum clustering iterations: max_iter; auxiliary iterations: aux_iter; maximum iterations for pretraining discriminator: max_iter_dis; discriminator learning rate: δ_{dis}; cluster learning rate: δ,clusters' centroids: C, target update iterations: $targ\_iter$; tolerance threshold: tol.

Output: Assignment results to element in X to specific cluster.

// Pretraining discriminator

for $i \leftarrow 1$ **to** max_iter_dis **do**
  Compute L_G using (3)
  // Update parameter
  $\omega \leftarrow \omega - \delta \Delta L_G$
end for
// Clustering optimization
for $i \leftarrow 1$ **to** max_iter **do**
  if i % $targ\_iter == 0$ **then**
    Compute Q using (3)
    // Save y_{pred} *result of previous step*
    $y_{pred\_old} \leftarrow y_{pred}$
    Compute y_{pred}
    Compute P using (4)

    if $\frac{1}{N}\Sigma(y_{pred} \neq y_{pred\_old}) < tol$ **then**
      break
    end if
  end if
  Compute L_G, L_D, L_{cls} using (5), (6), (7)
  if i % $aux\_iter <= (aux\_iter/2)$ **then**
    $\phi \leftarrow \phi - \delta \Delta L_D$
  else
    $\theta \leftarrow \theta - \delta \Delta L_{cls}$; $\phi \leftarrow \phi - \delta \Delta L_D$;
    $\omega \leftarrow \omega + \delta \Delta L_G$; $C \leftarrow C - \delta \Delta L_{cls}$
  end if
end for

from L1 to L6 contain sequences of species with different abundance levels. Besides, 9 datasets of Roche 454 single-end long reads named from R1 to R9, have the length of 700bp. The details of the datasets are presented in Appendix I. Furthermore, this study also evaluates the proposed algorithm on a real metagenome, Acid Mine Drainage (AMD) [2]. The dataset is downloaded from National Center for Biotechnology Information (NCBI) trace archive. This work tests MetaDEC, and compares it with published results of BiMeta, MetaCluster 2.0, AbundanceBin, MetaCluster 5.0 on those datasets [19].

B. EXPERIMENTAL SETTINGS

In the experiment, we use a specific machine for each phase of the proposed algorithm. Phase 1 is run on a machine with 500GB memory and Intel(R) Xeon(R) Gold 6152 processor. Phase 2, which includes the process of training neural networks, is run on a machine of NVIDIA Tesla P100 16GB GPU to accelerate the training time.

In the first phase of MetaDEC, two sequences are considered to be overlapped with each other if they share at least no_{olp} overlapping l-mer substrings. This work empirically set $no_{olp} = 5$ for short sequence datasets and $no_{olp} = 45$ for long sequence datasets. Besides, the value of l of l-mer to determine overlapping sequences in phase 1 of the proposed algorithm is set to 30 empirically.

The second phase of MetaDEC is performed with four types of architecture including *Tiny*, *Small*, *Large*, *Xlarge* which are presented in Appendix II. Experiments for simulated datasets in this work uses *Small* architecture.

In the other hands, the step of cluster initialization trains autoencoder and critic parts for 2000 iterations using Adam optimizer with a fixed learning rate at 0.001. Besides, in the cluster optimization step, discriminator is trained using the same optimizer with 200 iterations. The step continues to train the model using Adam optimizer with learning rate at 0.0001 with epsilon value 10e-8. The training will be stopped if the ratio of changing clustered data points after two consecutive iterations less than a threshold tol of 0.0001% or reaching the maximum number of 300 iterations. In addition, the degree of freedom η in t-student distribution used in step 2 of the phase is set as 1.

C. RESULTS ON SIMULATED DATASETS

1) RESULTS ON SHORT READ DATASETS

The proposed method is firstly compared with BiMeta, MetaCluster 5.0 on short-read datasets from S1 to S10. The bar charts in Fig. 2 present the precision and recall of the binning algorithms. It can be seen from the chart that MetaDEC gets higher precision values than BiMeta and MetaCluster 5.0 on 9/10 datasets. Besides, there are 6/10 cases in which the proposed method returns better recall values than the two remaining algorithms.

Furthermore, overall F-measures of the three methods are presented in Table 1. The table shows that MetaDEC outperforms both BiMeta and MetaCluster 5.0 for 6/10 samples. On average, MetaDEC gets higher 3.79% and 12.91% compared with the two remaining methods, respectively.

TABLE 1. F-measure performance of MetaCluster 5.0, BiMeta and MetaDEC for S1 to S10 datasets.

Dataset	MetaCluster 5.0 [19]	BiMeta [19]	MetaDEC
S1	67.11%	98.02%	**99.32%**
S2	88.68%	60.14%	**93.52%**
S3	71.98%	97.72%	**98.58%**
S4	77.2%	99.35%	**99.45%**
S5	80.08%	89.32%	**91.21%**
S6	88.74%	**99.29%**	99.24%
S7	**91.04%**	77.24%	84.19%
S8	57.94%	70.27%	**79.85%**
S9	67.56%	**77.01%**	69.70%
S10	52.17%	**65.37%**	56.51%
Average	74.25%	83.37%	**87.16%**

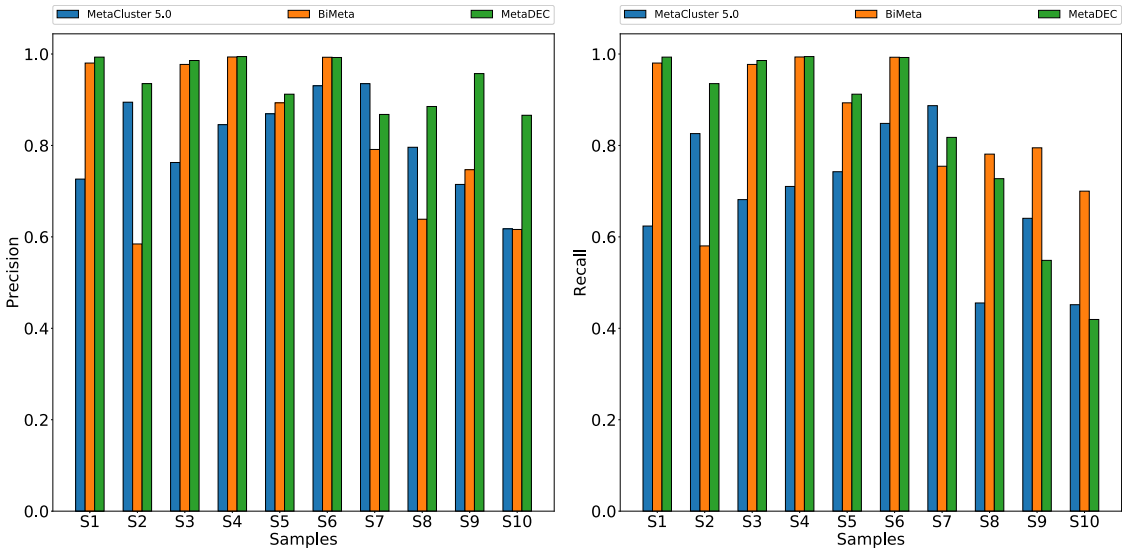


FIGURE 2. Precisions and recalls of MetaCluster 5.0 [19], BiMeta [19], and MetaDEC on datasets from S1 to S10.

Considering the ability to classify metagenomes of different abundance levels, MetaDEC is compared with BiMeta and AbundaceBin on datasets from L1 to L6. The datasets contain two species of *Eubacterium eligens* and *Lactobacillus amylovorus*, and have different abundance ratios. The bar chart in Fig. 3 shows that MetaDEC achieves higher F-measure than both BiMeta and AbundanceBin on all samples of low-abundance ratios (Datasets from L1 to L4 with ratios of 1:1, 1:2, 1:3, 1:4, respectively). Especially, MetaDEC outperforms BiMeta for all cases. Because the strength of AbundanceBin is based on abundance ratios of species in datasets to classify sequences, it gets better results than BiMeta and MetaDEC for samples L5, and L6 (with highly different abundance levels).

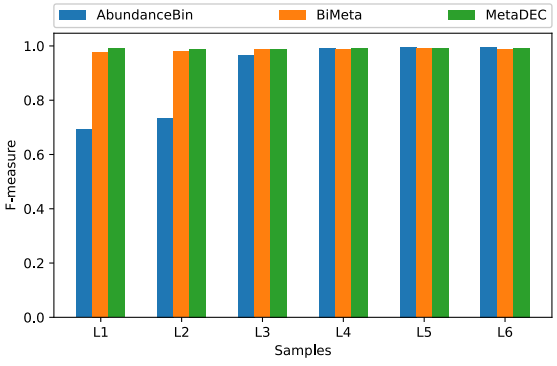


FIGURE 3. F-measures of AbundanceBin [19], BiMeta [19], and MetaDEC on datasets from L1 to L6.

2) RESULTS ON LONG READ DATASETS

The proposed method is also compared with MetaCluster 2.0 and BiMeta on long sequence datasets from R1 to R9. Table 2 shows that MetaDEC outperforms the two algorithms for 8/9 cases. The proposed method gets 3.44% and 11.44% higher F-measure on average than BiMeta and MetaCluster 2.0 on the samples, respectively. For more details, bar charts in Fig. 4 show that MetaDEC returns higher precision values than both the algorithms on 8/9 samples. Besides, there are 6/9 cases that the proposed method achieves better recall values than MetaCluster 2.0 and BiMeta.

TABLE 2. F-measure performance of MetaCluster 2.0, BiMeta and MetaDEC for R1 to R9 datasets.

Dataset	MetaCluster 2.0 [19]	BiMeta [19]	MetaDEC
R1	75.61%	**97.82%**	97.48%
R2	80.40%	87.19%	**95.66%**
R3	66.83%	77.59%	**90.98%**
R4	96.42%	98.94%	**99.39%**
R5	94.75%	98.97%	**99.67%**
R6	95.40%	96.09%	**99.1%**
R7	69.96%	91.63%	**93.81%**
R8	96.74%	97.92%	**99.22%**
R9	-	86.42%	**88.26%**
Average	84.51%	92.51%	**95.95%**

D. RESUTLS ON A REAL DATASET

The proposed method is also tested on a real metagenome. It is compared with BiMeta on the dataset of AMD. This dataset is revealed in previous studies that it contains five dominant species, including *Ferroplasma sp. Type II*, *Leptospirillum sp. Group II*, *Leptospirillum sp. Group III*, *Ferroplasma acidarmanus Type I*, and *Thermoplasmatalesarchaeon Gpl* with abundance ratios of 5:5:1:1:1, respectively [2]. In order to assess binning results, assembled scaffolds of the five species are downloaded from NCBI. Reads from each cluster in the output results are aligned with the scaffolds using the BLAST tool to calculate roughly classification accuracy.

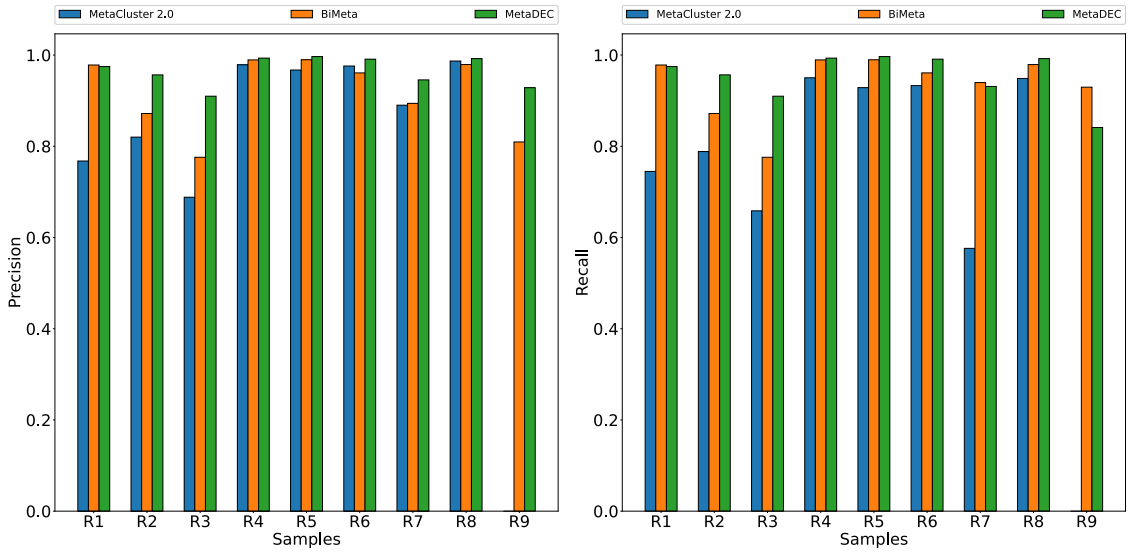


FIGURE 4. Precisions and recalls of MetaCluster 2.0 [19], BiMeta [19], and MetaDEC on datasets from R1 to R9.

The dataset is tested using two network architectures of *Small* and *Large* presented in Appendix II. Experimental results show that MetaDEC achieves F-measure values of 68.85% and 70.1% when architecture *Small* and *Large* applied, respectively. The results are very comparative to BiMeta which was reported that it got a F-measure value of 68.3 on the dataset [19].

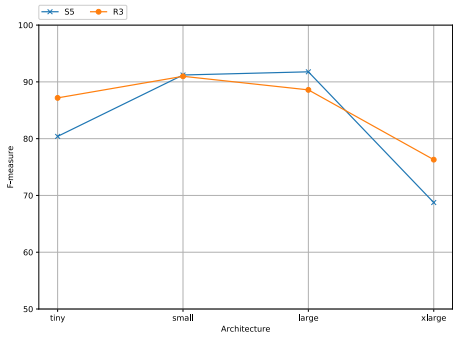


FIGURE 5. F-measures of MetaDEC on dataset S5 and R3 with different learning network architectures.

E. NETWORK EVALUATION

The section evaluates the performance and running time of MetaDEC on different sizes of model architectures. The proposed algorithm is performed on four architectures *Tiny*, *Small*, *Large*, and *Xlarge* presented in Appendix II on S5 and R3 datasets. It is noted that the architectures are assigned with different numbers of layers (*Tiny* is 3 layers, *Small* is 4 layers, *Large* is 5 layers, and *Xlarge* is 7 layers). Line chart in Fig. 5 shows that MetaDEC reaches the highest F-measures with architecture *Large* on sample S5 and with architecture *Small* on sample R3. However, it gets the lowest F-measures with architecture *Xlarge* on both datasets.

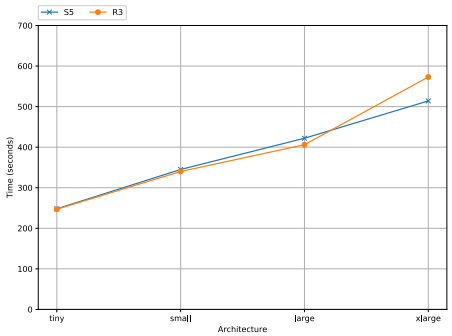


FIGURE 6. Running time of MetaDEC on dataset S5 and R3 with different learning network architectures.

Furthermore, with the shallow network *Tiny*, MetaDEC also returns lower F-measures on the datasets compared with architectures of *Small* and *Large*. The results reveal that if a model is too shallow (e.g. *Tiny* with 3 layers), it is not able to learn meaningful latent space that has the ability to describe original data space. In contrast, a model that is too deep (e.g. *Xlarge* with 7 layers) is prone to overfitting the data instead of extracting meaningful features from it. In both cases, latent spaces learned from these models would not result in a good clustering performance.

As can be seen from the visualization of latent spaces from results of cluster initialization for the both datasets (section II-B1 in Phase 2 of MetaDEC) presented in Appendix III, latent spaces of species from *Tiny* and *Xlarge* models tend to be mixed together. It can result in poor centroid initialization. In contrast, latent spaces of *Small* and *Large* models give a more convex shape for each specie, and result in a good centroid initialization.

In the aspect of processing time, the line chart in Fig. 6 shows that the running time of MetaDEC is proportional

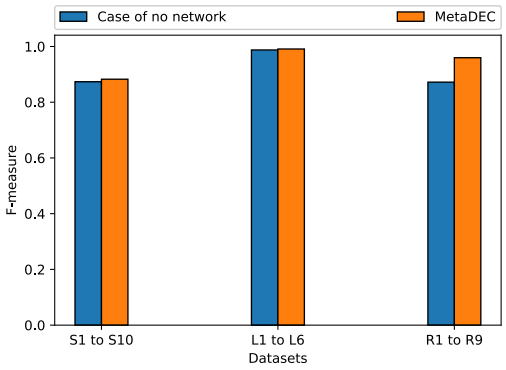


FIGURE 7. F-measure of the case with zero layer neural network and MetaDEC with architecture neural network *Small*. The F-measure values are the average F-measure on datasets S1-S10, L1-L6, R1-R9, respectively.

TABLE 3. Description of simulated datasets.

Samples	No. of species	Phylogenetic distance	Ratio	No. of reads
S1	2	Species	1:1	96367
S2	2	Species	1:1	195339
S3	2	Order	1:1	338725
S4	2	Phylum	1:1	375302
S5	3	Species and Family	1:1:1	325400
S6	3	Phylum and Kingdom	3:2:1	713388
S7	5	Order, Order, Genus, Order	1:1:1:4:4	1653550
S8	5	Genus, Order, Order, Order	3:5:7:9:11	456224
S9	15	Various distances	1:1:1:1:1: 2:2:2:2:2: 3:3:3:3:3	2234168
S10	30	Various distances	4:4:4:4:4: 6:6:6:6:6: 7:7:7:7:7: 8:8:8:8:8: 9:9:9:9:9: 10:10:10:10:10	4990632
L1	2	Class	1:1	176688
L2	2	Class	1:2	259568
L3	2	Class	1:3	342448
L4	2	Class	1:4	425328
L5	2	Class	1:5	508209
L6	2	Class	1:6	591089
R1	2	Species	1:1	82960
R2	2	Genus	1:1	77293
R3	2	Genus	1:1	93267
R4	2	Family	1:1	34457
R5	2	Family	1:1	40043
R6	2	Order	1:1	70550
R7	3	Family and Order	1:1:8	290473
R8	3	Order and Phylum	1:1:8	374830
R9	6	Species, Order, Family and Kingdom	1:1:1:1:2:14	588258

to the complexity level of model architectures. It can be understood because the larger size of the model is used, the higher computational resources are required.

Furthermore, this work also evaluates the performance of the case of using neural network with zero layer (by applying k-means directly on k-mer frequency representation). The experiment result is compared with MetaDEC using *Small* architecture on all of the simulated datasets. The chart in Fig. 7 shows that MetaDEC performs better than the case merely applying k-means on average. It means that applying

TABLE 4. Details of encoder part.

Name \ Layer	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7
Tiny	136	500	10				
Small	136	500	1000	10			
Large	136	500	500	2000	10		
Xlarge	136	500	500	1000	1000	2000	10

TABLE 5. Details of decoder part, which has the inverse order of encoder.

Name \ Layer	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7
Tiny	10	500	136				
Small	10	1000	500	136			
Large	10	2000	500	500	136		
Xlarge	10	2000	1000	1000	500	500	136

TABLE 6. Details of critic part. Last layer of the critic has 10 dimensions, which are averaged into a scalar.

Name \ Layer	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7
Tiny	136	500	1				
Small	136	500	1000	1			
Large	136	500	500	2000	1		
Xlarge	136	500	500	1000	1000	2000	1

TABLE 7. Details of discriminator part.

Name \ Layer	Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6	Layer 7
Tiny	136	500	1				
Small	136	500	1000	1			
Large	136	500	500	2000	1		
Xlarge	136	500	500	1000	1000	2000	1

ADEC method providing promising classification quality on metagenomic data.

IV. CONCLUSION

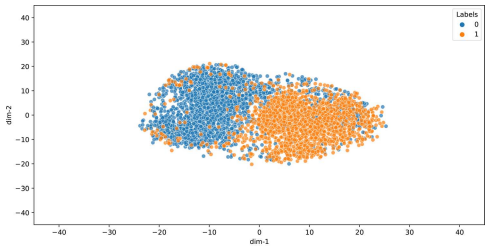
The complexity of the microbial community requires in-depth exploration strategies to yield practical benefits. This study takes advantage of deep learning techniques to effectively solve the binning problem of metagenomic data. The proposed method employs an adversarial deep embedding approach to automatically learn a latent space of k-mer frequency representation of sequence groups, and is demonstrated to outperform state-of-art binning algorithms. In future works, it is worth extending our method to estimate the number of potential clusters in order to be more suitable in the binning context. Besides, a deeper investigation to find a better measuring function for soft cluster assignment in the second phase of MetaDEC is one of the potential research directions. Source codes and datasets used in this work can be downloaded from *https://bioinfolab.fit.hcmute.edu.vn/MetaDEC*.

APPENDIX I. DATASETS

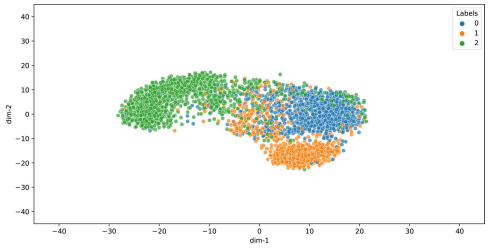
See Table 3.

APPENDIX II. ARCHITECTURE TYPES

In the trainning process for neural networks in phase 2 of MetaDEC, each architecture consists of four parts of encoder (θ), decoder (ϕ), critic (ψ), discriminator (ω). Each network is a multi layer perceptron. Each layer i of a network

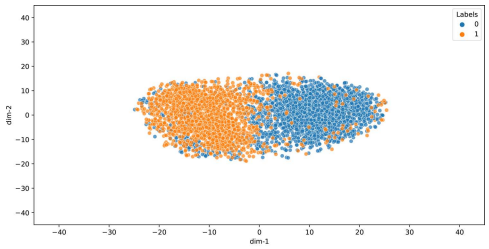


(a) Latent space of R3

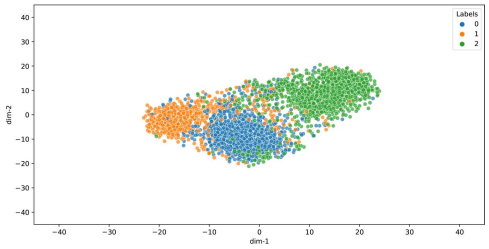


(b) Latent space of S5

FIGURE 8. Visualization of R3 and S5 on *Tiny* architecture.



(a) Latent space of R3



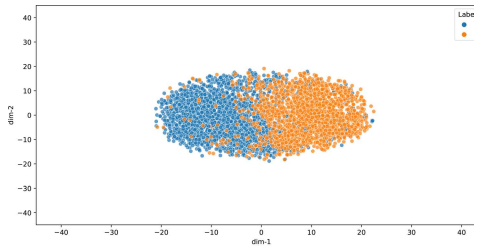
(b) Latent space of S5

FIGURE 9. Visualization of R3 and S5 on *Small* architecture.

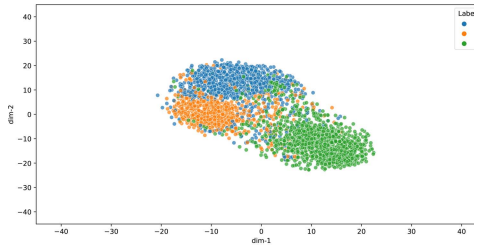
is defined as follows.

$$x_i = g_i(W_i x_{i-1} + b_i) \tag{11}$$

where W_i and b_i are layer's parameters, x_{i-1} is the output of previous layer, g_i is the non-linear activation function. MetaDEC uses ReLU for all layers' g_i, except the last layers of encoder, decoder and critic, which are set to be the identity function and the last layer of discriminator is set to be the sigmoid function. We initialized all layers' parameters to random numbers drawn from a zero-mean Gaussian distribution with a standard deviation of 0.01. Tables 4, 5, 6, 7 describe the architectures used in our experiments for encoder, decoder, critic, and discriminator, respectively.

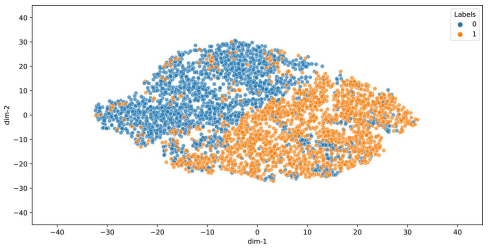


(a) Latent space of R3

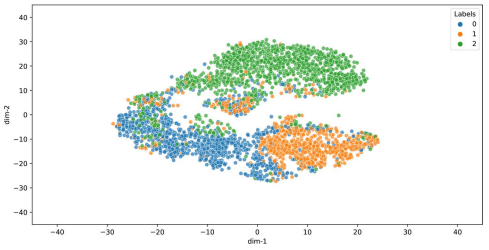


(b) Latent space of S5

FIGURE 10. Visualization of R3 and S5 on *Large* architecture.



(a) Latent space of R3



(b) Latent space of S5

FIGURE 11. Visualization of R3 and S5 on *Xlarge* architecture.

APPENDIX III. VISUALIZATION OF LATENT SPACE

The visualization of latent spaces from results of cluster initialization (section II-B1 in Phase 2 of MetaDEC) for datasets S5 and R3 on four architectures, which are *Tiny* (Fig. 8), *Small* (Fig. 9), *Large* (Fig. 10), and *Xlarge* (Fig. 11).

REFERENCES

[1] *National Research Council (US) Committee on Metagenomics: Challenges and Functional Applications, 'Why Metagenomics?' The New Science of Metagenomics: Revealing the Secrets of Our Microbial Planet*, NAP, Washington, DC, USA, 2007.

[2] G. W. Tyson, J. Chapman, P. Hugenholtz, E. E. Allen, R. J. Ram, P. M. Richardson, V. V. Solovyev, E. M. Rubin, D. S. Rokhsar, and J. F. Banfield, "Community structure and metabolism through reconstruction of microbial genomes from the environment," *Nature*, vol. 428, no. 6978, pp. 37–43, Mar. 2004.

[3] J. C. Venter *et al.*, "Environmental genome shotgun sequencing of the Sargasso sea," *Science*, vol. 304, no. 5667, pp. 66–74, Apr. 2004.
[4] J. Shendure and H. Ji, "Next-generation DNA sequencing," *Nature Biotechnol.*, vol. 26, no. 10, pp. 1135–1145, 2008.
[5] N. Shah, E. K. Molloy, M. Pop, and T. Warnow, "TIPP2: Metagenomic taxonomic profiling using phylogenetic markers," *Bioinformatics*, vol. 37, no. 13, pp. 1839–1845, Jul. 2021.
[6] A. Milanese, D. R. Mende, L. Paoli, G. Salazar, H.-J. Ruscheweyh, M. Cuenca, P. Hingamp, R. Alves, P. I. Costea, L. P. Coelho, T. S. B. Schmidt, A. Almeida, A. L. Mitchell, R. D. Finn, J. Huerta-Cepas, P. Bork, G. Zeller, and S. Sunagawa, "Microbial abundance, activity and population genomic profiling with mOTUs2," *Nature Commun.*, vol. 10, no. 1, pp. 1–11, Dec. 2019.
[7] R. J. Case, Y. Boucher, I. Dahllöf, C. Holmström, W. F. Doolittle, and S. Kjelleberg, "Use of 16S rRNA and rpoB genes as molecular markers for microbial ecology studies," *Appl. Environ. Microbiol.*, vol. 73, no. 1, pp. 278–288, Jan. 2007.
[8] H. G. Martin, N. Ivanova, V. Kunin, F. Warnecke, K. Barry, A. C. McHardy, C. Yeates, S. He, A. Salamov, E. Szeto, E. Dalin, N. Putnam, H. J. Shapiro, J. L. Pangilinan, I. Rigoutsos, N. C. Kyrpides, L. L. Blackall, K. D. McMahon, and P. Hugenholtz, "Metagenomic analysis of phosphorus removing sludge communities," *Genome Res.*, vol. 18, no. 2, pp. 293–297, 2008.
[9] D. H. Huson, S. Beier, I. Flade, A. Górska, M. El-Hadidi, S. Mitra, H.-J. Ruscheweyh, and R. Tappu, "MEGAN community edition–interactive exploration and analysis of large-scale microbiome sequencing data," *PLOS Comput. Biol.*, vol. 12, no. 6, Jun. 2016, Art. no. e1004957.
[10] D. H. Huson, B. Albrecht, C. Bağcı, I. Bessarab, A. Górska, D. Jolic, and R. B. H. Williams, "MEGAN-LR: New algorithms allow accurate binning and easy interactive exploration of metagenomic long reads and contigs," *Biol. Direct*, vol. 13, no. 1, pp. 1–17, Jan. 2018.
[11] B. Buchfink, C. Xie, and D. H. Huson, "Fast and sensitive protein alignment using DIAMOND," *Nature Methods*, vol. 12, no. 1, pp. 59–60, Jan. 2015.
[12] S. M. Kiełbasa, R. Wan, K. Sato, P. Horton, and M. C. Frith, "Adaptive seeds tame genomic sequence comparison," *Genome Res.*, vol. 21, no. 3, pp. 487–493, Mar. 2011.
[13] D. E. Wood, J. Lu, and B. Langmead, "Improved metagenomic analysis with Kraken 2," *Genome Biol.*, vol. 20, no. 1, pp. 1–13, Dec. 2019.
[14] Q. Liang, P. W. Bible, Y. Liu, B. Zou, and L. Wei, "DeepMicrobes: Taxonomic classification for metagenomics with deep learning," *NAR Genomics Bioinf.*, vol. 2, no. 1, Mar. 2020, Art. no. lqaa009.
[15] B. Yang, Y. Peng, H. C. M. Leung, S. M. Yiu, J. Qin, R. Li, and F. Y. L. Chin, "Metacluster: Unsupervised binning of environmental genomic fragments and taxonomic annotation," in *Proc. ACM-BCB*, New York, NY, USA, 2010, pp. 170–179.
[16] Z. Jiang, X. Li, and L. Guo, "MetaCRS: Unsupervised clustering of contigs with the recursive strategy of reducing metagenomic dataset's complexity," *BMC Bioinf.*, vol. 22, no. 12, pp. 1–17, Dec. 2021.
[17] Y. Wang, H. C. M. Leung, S. M. Yiu, and F. Y. L. Chin, "MetaCluster 5.0: A two-round binning approach for metagenomic data for low-abundance species in a noisy sample," *Bioinformatics*, vol. 28, no. 18, pp. i356–i362, Sep. 2012.
[18] Y.-W. Wu and Y. Ye, "A novel abundance-based algorithm for binning metagenomic sequences using *l*-tuples," *J. Comput. Biol.*, vol. 18, no. 3, pp. 523–534, 2011.
[19] L. V. Vinh, T. V. Lang, L. T. Binh, and T. V. Hoai, "A two-phase binning algorithm using *l*-mer frequency on groups of non-overlapping reads," *Algorithms Mol. Biol.*, vol. 10, no. 1, p. 2, Dec. 2015.
[20] S. Girotto, C. Pizzi, and M. Comin, "MetaProb: Accurate metagenomic reads binning based on probabilistic sequence signatures," *Bioinformatics*, vol. 32, no. 17, pp. i567–i575, Sep. 2016.
[21] F. Andreace, C. Pizzi, and M. Comin, "MetaProb 2: Metagenomic reads binning based on assembly using minimizers and *K*-mers statistics," *J. Comput. Biol.*, vol. 28, no. 11, pp. 1052–1062, Nov. 2021.
[22] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, vol. 37, 2017, pp. 478–487.
[23] N. Mrabah, M. Bouguessa, and R. Ksantini, "Adversarial deep embedded clustering: On a better trade-off between feature randomness and feature drift," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 4, pp. 1603–1617, Apr. 2022, doi: 10.1109/TKDE.2020.2997772.
[24] I. Bonet, A. Pena, C. Lochmuller, A. Patino, and M. Gongora, "Deep clustering for metagenomics," in *Proc. CIBB*, Bergamo, Italy, 2019, pp. 335–347.
[25] J. N. Nissen, J. Johansen, R. L. Allesøe, C. K. Sønderby, J. J. A. Armenteros, C. H. Grønbech, L. J. Jensen, H. B. Nielsen, T. N. Petersen, O. Winther, and S. Rasmussen, "Improved metagenome binning and assembly using deep variational autoencoders," *Nature Biotechnol.*, vol. 39, no. 5, pp. 555–560, May 2021, doi: 10.1038/s41587-020-00777-4.
[26] Y. Wang, H. C. M. Leung, S. M. Yiu, and F. Y. L. Chin, "MetaCluster 4.0: A novel binning algorithm for NGS reads and huge number of species," *J. Comput. Biol.*, vol. 19, no. 2, pp. 241–249, Feb. 2012.
[27] G. Karypis and V. Kumar, "METIS—Unstructured graph partitioning and sparse matrix ordering system, version 2.0," Dept. Comput. Sci., Univ. Minnesota, Minneapolis, MN, USA, Tech. Rep., 1995. [Online]. Available: https://dm.kaist.ac.kr/kse625/resources/metis.pdf
[28] B. Chor, D. Horn, N. Goldman, Y. Levy, and T. Massingham, "Genomic DNA *k*-mer spectra: Models and modalitie," *Genome Biol.*, vol. 10, no. 10, pp. 1–10, 2009.
[29] D. Berthelot, C. Raffel, A. Roy, and I. Goodfellow, "Understanding and improving interpolation in autoencoders via an adversarial regularizer," in *Proc. ICLR*, New Orleans, LA, USA, 2019, pp. 1–19.

HUYNH QUANG BAO received the B.S. degree in computer engineering from the HCMC University of Technology (Vietnam National University, Ho Chi Minh City), in 2013, where he is currently pursuing the M.S. degree in computer science. His research interests include deep learning and metagenomic binning.

LE VAN VINH received the bachelor's degree in information technology and the M.Sc. degree in computer science from the University of Science (Vietnam National University, Ho Chi Minh City), in 2005 and 2009, respectively, and the Ph.D. degree in computer science from the HCMC University of Technology (Vietnam National University, Ho Chi Minh City) in 2017. Currently, he is working with the Faculty of Information Technology, HCMC University of Technology and Education, Vietnam. His research interests include bioinformatics, high-performance computing, data science, and deep learning.

TRAN VAN HOAI received the Ph.D. degree in applied mathematics from Heidelberg University, in 2005. His theoretical background is combinatorial optimization. After graduation, he has initiated and joined different projects to apply his knowledge on optimization to various fields. He is an Associate Professor with the HCMC University of Technology (Vietnam National University, Ho Chi Minh City). His research interests include metagenomic binning in bioinformatics, convexity in computational geometry, and practical optimization problems.