A New Unsupervised Binning Approach for Metagenomic Sequences Based on N-grams and Automatic Feature Weighting

Journal:	*IEEE/ACM Transactions on Computational Biology and Bioinformatics*
Manuscript ID:	TCBBSI-2013-10-0293
Manuscript Type:	SI: GIW 2013
Keywords:	I.5.3 Clustering < I.5 Pattern Recognition < I Computing Methodologies, J.3.a Biology and genetics < J.3 Life and Medical Sciences < J Computer Applications, Metagenomics, Binning

A New Unsupervised Binning Approach for Metagenomic Sequences Based on N-grams and Automatic Feature Weighting

Ruiqi Liao, Ruichang Zhang, Jihong Guan, and Shuigeng Zhou, *Member, IEEE*

Abstract—The rapid development of high-throughput technologies enables researchers to sequence the whole metagenome of a microbial community sampled directly from the environment. The assignment of these sequence reads into different species or taxonomical classes is a crucial step for metagenomic analysis, which is referred to as *binning* of metagenomic data. Most traditional binning methods rely on known reference genomes for accurate assignment of the sequence reads, therefore cannot classify reads from unknown species without the help of close references. To overcome this drawback, unsupervised learning based approaches have been proposed, which need not any known species' reference genome for help. In this paper, we introduce a novel unsupervised method called MCluster for binning metagenomic sequences. This method uses N-grams to extract sequence features and automatic feature weighting to improve the performance of the basic K-means clustering algorithm. We evaluate MCluster on a variety of simulated datasets and a real dataset, and compare it with three latest binning methods: AbundanceBin, MetaCluster 3.0 and MetaCluster 5.0. Experimental results show that MCluster achieves obviously better overall performance (F-measure) than AbundanceBin and MetaCluster 3.0 on long metagenomic reads ($\geq$ 800bp); while compared with MetaCluster 5.0, MCluster obtains a larger *sensitivity*, and a comparable yet more stable F-measure on short metagenomic reads ($<$ 300bp). This suggests that MCluster can serve as a promising tool for effectively binning metagenomic sequences.

Index Terms—Metagenomics; Binning; N-grams; Feature weighting; Algorithms.

1 Background

As a rapidly developing research area, metagenomics [1] refers to the genomic analysis of microbial communities sampled directly from their natural environments without prior culturing. It provides valuable insights into the identities, composition, dynamics, functions and interactions of diverse microbial communities, especially those cannot be cultured in the laboratory. For example, the metagenomics research of human gut microbial communities revealed the association between gut microbial composition and human health [2]; soil metagenomics researches discovered the influences of different environments on microbe communities [3]. With the development of high-throughput Next Generation Sequencing (NGS) technologies [4], researchers are able to directly sequence the genomes of multiple microorganisms obtained from an environmental sample, which greatly facilitates metagenomics researches in many areas [2], [5], [6], [7], [8]. Among the sequencing technologies, 454 Roche Pyrosequencing has been the most widely used one in metagenomics research for its ability to generate much longer reads than other technologies [9]. 454 technology can output reads of about 1000bp while Illumina and SOLID mainly output reads of less than 300bp.

Metagenomic data generated by Next Generation Sequencing contain a large number of short sequences (*i.e.* reads) from multiple species. To analyze these data, a crucial step is to group reads of the same species or taxonomic class together in order to get the taxonomic composition of the microbial community, which is also called binning [10]. Most existing binning methods can be roughly classified into two categories: similarity based methods and composition based methods.

Similarity based methods such as MEGAN [11] first align sequence reads to known reference genomes, then group the reads based on the alignment result. Reads aligned to the same genome or taxonomic class are grouped together. However, the successful application

- *R.Q. Liao, R.C. Zhang, and S.G. Zhou (corresponding author) are with the School of Computer Science, and Shanghai Key Lab of Intelligent Information Processing, Fudan University, 220 Handan Road, Shanghai 200433, China. E-mail: {rqliao, rczhang, sgzhou}@fudan.edu.cn.*
- *J.H. Guan is with the Department of Computer Science and Technology, Tongji University, 4800 Caoan Road, Shanghai 201804, China. E-mail: jhguan@tongji.edu.cn.*

Manuscript received XXX XX, 2013; revised XXX XX, 2013.

of similarity-based methods relies heavily on the availability of known microorganism genomes. As a matter of fact, up to 99% of bacteria found in environmental samples are unknown or cannot be cultured and separated in laboratories [12], therefore may have no genome sequences available. In a coral metagenomic dataset, only 12% of the reads can be aligned to known genomes [13]. It was reported that the accuracies of similarity based methods drop sharply when related known genomes are not provided [10], which becomes a major bottleneck for applying this kind of methods to the rapidly increasing amount of metagenomic data. On the other hand, composition based methods use supervised or unsupervised techniques to assign reads to different groups. The features are directly extracted from the nucleotide sequences, which include oligonucleotide frequencies, GC-content, codon usage etc.

For supervised composition based methods, the sequence features are used to train classifiers for a certain number of species or taxonomic classes. For example, existing methods used SVM [14], Naïve Bayes [15], KNN [16], Interpolated Markov Model [17] to train classifiers for taxonomic assignment of metagenomic sequences. However, the performance of these methods still relies substantially on the availability of known genomes, used as training samples.

To resolve or alleviate the problem of reference or training genome unavailability, in the past years composition based binning methods using unsupervised or semi-supervised techniques were proposed to deal with metagenomic data from unknown species. This kind of methods often uses k-mers (also called N-grams in natural language processing area) to generate the features of sequences for unsupervised or semi-supervised binning. For instance, Abe et al. [18], [19] showed the feasibility to classify environmental genomic fragments with minimal length of 5 Kbp using a self-organizing map (SOM). Chan et al. [20] developed a semi-supervised method to cluster metagenomic sequences by a seeded growing self-organizing map (S-GSOM).

Recently, Wu et al. [21] proposed the Abundancebin method that extracts k-mers from sequence reads and models the distribution of reads from each species by Poisson distribution, which can effectively separate the reads from species with different abundance ratios. However, Abundancebin does not work well when the datasets consist of reads from different species with identical abundance ratio. Leung et al. [12] developed the MetaCluster 3.0 method that uses 4-mers to build the feature vectors, and clusters them using the classical K-median algorithm, then merges close clusters. MetaCluster 3.0 achieved better performance than Abundancebin in both evenly and unevenly distributed datasets with read length of 1000bp. Later, Wang et al. introduced two improved versions of MetaCluster 3.0, which are MetaCluster 4.0 [22] and MetaCluster 5.0 [23]. MetaCluster 4.0 can deal with short reads by employing a preprocessing stage to concatenate short reads to longer ones based on sequence overlapping of the short reads. MetaCluster 5.0 advances MetaCluster 4.0 to handle short reads from species with different abundance ratios. The series of MetaCluster algorithms stand for the state of the art unsupervised binning techniques. An outstanding feature of the MetaClusters is that they can automatically determine the number of species hidden in the sequence reads. However, our experiments (refer to Sec. 4 for the detail) show that the number of species output by the MetaCluster algorithms is often inaccurate when the real number of species hidden in the sequences is relatively large (≥ 3).

All the existing unsupervised methods for metagenomic sequence bining take the weights of different k-mers equally in the clustering process. However, different k-mers may actually have divergent influences on the identification of each species according to a previous research [24]. The incorporation of the k-mer preference information can improve the performance of sequence clustering, which has been validated by one of our previous works on grouping miRNA sequences [25]. In this paper, we develop a new unsupervised binning approach called *MCluster* for metagenomic sequences, which is based on the N-grams representation of sequence reads and an improved version of the classical K-means algorithm with an automatic feature weighting mechanism. When applied to 31 simulated datasets and a real dataset sampled from Acid Mine Drainage, MCluster achieves better overall performance than Abundance-Bin and MetaCluster 3.0 on long metagenomic reads ($\geq$ 800bp); while compared with MetaCluster 5.0, MCluster obtains a larger *sensitivity*, and a comparable yet more stable F-measure on short metagenomic reads ($<$ 300bp). This demonstrates that MCluster is a promising method for effectively binning metagenomic sequences.

The rest of this paper is organized as follows. Section 2 presents the MCluster method. Section 3 gives the evaluation results. Section 4 discusses the proposed method and the empirical results as well as future work. Section 5 concludes the paper.

2 Method

Metagenomic sequence binning using the MCluster approach consists of three main phases: (1) each metagenomic sequence is represented by a feature vector using the N-grams scheme; (2) sequences are grouped using a clustering algorithm with automatic feature weighting; (3) the clustering result is evaluated by three metrics, *precision*, *sensitivity* and *F-measure*. The pipeline of MCluster is shown in Fig. 1. In what follows, we present the implementation techniques of MCluster in detail.

2.1 Feature extraction using N-grams

Metagenomic data contain a large number of sequence reads coming from different species. In our method, we

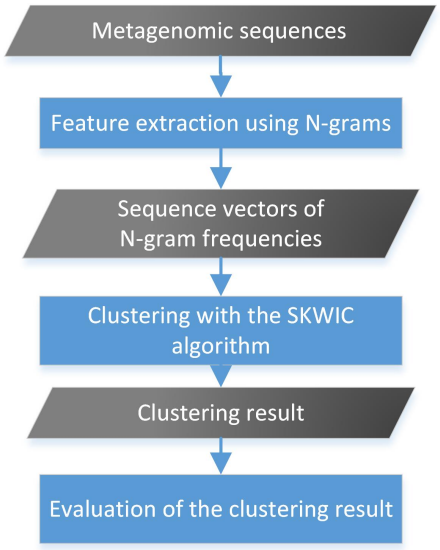

Fig. 1: The pipeline of MCluster. Here, the parallelograms indicate input/output modules, the rectangles stands for functional modules.

use the N-grams scheme to represent each read as a feature vector. An N-gram (also called a k-mer in the literature) is a subsequence consisting of N spatially consecutive characters extracted from a given sequence. In the context of DNA sequence, each character can be one of the four bases: A, T, C or G. Therefore, to represent DNA sequences, the total number of N-grams is 4^N. Concretely, we use the sliding window approach to count the frequency of each N-gram in the whole sequence [12]. Assume that f_w is the number of occurrences of an N-gram w, then the total number of all N-grams in the sequence is $\sum f_w$=M-N+1, where M is the length of the sequence. Here, for each N-gram, its complementary sequence in the other chain of the DNA sequence is also counted. Some existing works (*e.g.* the MetaCluster methods) use one N-gram to stand for each pair of complementary N-grams in the feature vectors, we give a discussion on this issue in Section 4.

According to the works of Chor et al. [26] and Zhou et al. [27], 4-gram is the best choice to extract features from metagenomic sequences. Therefore, in our method, we choose 4-grams to represent each sequence read as a 256-dimension vector used for clustering.

2.2 Clustering by Automatic Feature Weighting

Clustering is the process of automatically grouping a set of data objects into different groups (*i.e.* clusters), without any prior knowledge to which group each data object belongs. The target is to assign the data objects from the same category into the same cluster. In the context of metagenomic sequence binning, the data objects refer to short sequence reads and the task is to assign them into different clusters (*i.e.* the genomes of bacteria) without knowing the true taxonomic class of each read.

There are various clustering methods proposed for different applications. Among them, K-means is one of the most-widely used, which is an effective method to automatically group a set of data objects into clusters in an iterative manner. The basic algorithm of K-means is as follows: first, specify the number K of clusters to be obtained and select K initial centroids (the centers of clusters); after that, iteratively distribute data objects to the clusters whose centroids are nearest to them, and update the centroids according to the current data assignments. Such a process is performed iteratively until the centroids do not change or the amount of changes is under a specified threshold.

The basic K-means algorithm treats each dimension or feature as equally relevant to each cluster. However, actually in many circumstances, different clusters differ largely in their best feature sets, and the relationships between clusters and their respective feature sets need to be discovered in the clustering process. To solve this problem, an improved version of the K-means algorithm—the SKWIC clustering algorithm—was proposed by Frigui *et al.* [28] for clustering text documents with different weight for each word. The main advantage of the SKWIC algorithm over the basic K-means algorithm is that the former tunes the weight of each feature in each cluster when doing clustering. The SKWIC algorithm achieves considerable better performance than the traditional K-means algorithm when applied to text document clustering.

According to the research of Karlin *et al.* [24], N-grams (or k-mers) have different frequencies in different species, therefore they should have specific weights in defining the cluster consisting of reads from a specific species. By assigning different weights to different N-grams in different cluster during the clustering process, we implement the SKWIC algorithm and integrat it into MCluster to carry out metagenomic sequence binning.

As an improved version of K-means, SKWIC tries to minimize the following objective function [28] (the following equations are mainly from [28] after correcting some typos and errors in the original formulae. we present them here so that the readers can understand the SKWIC algorithm and the MCluster method well):

$$J(K,V;\chi)=\sum_{i=1}^{K}\sum_{x_j\in\chi_i}\sum_{k=1}^{n}v_{ik}D^k_{wc_{ij}}+\sum_{i=1}^{K}\delta_i\sum_{k=1}^{n}v_{ik}^2 \quad (1)$$

subject to

$$v_{ik}\in[0,1]\quad\forall i,k\quad and\quad\sum_{k=1}^{n}v_{ik}=1\quad\forall i \quad (2)$$

where K is the number of clusters, n is the number of dimensions; χ=$\bigcup_{i=1}^{K}\chi_i$ and χ_i indicates the set of data items in cluster i; V is a $K\times n$ matrix, V=[V_1, V_2, $\cdots$, V_K]^T and V_i=[v_{i1},v_{i2}, $\cdots$, v_{in}], v_{ik} is the weight of dimension k for cluster i; $D^k_{wc_{ij}}$ means the distance between

data item j and the center of cluster i along dimension k. This objective function is different from that of the classical K-means algorithm. Specifically, the first component is very much like the objective function of K-means except that its distances along individual dimensions are weighted with a positive value. Dimensions with larger weights are more relevant to that cluster than those with smaller weights. The second component in the objective function is a weighted sum of squares of all weights. δ_i is for balancing the two components.

The first component in Eq. (1) is the sum of weighted distances between data points (read vectors here) and their corresponding cluster centroids, which is used to obtain compact clusters. It is minimized when only one dimension in a cluster is very relevant and all the other dimensions are irrelevant. The second component in Eq. (1) is to control the weights v_{ik}. It is minimized when all dimensions are equally weighted. By combining these two components and selecting an appropriate parameter δ_i, the resulting clusters will have their within-cluster weighted distances minimized, while the feature weights of each cluster are optimized.

Given a set of centroids and a partitioning, the Lagrange multiplier method is adopted to solve the constrained optimization problem about J with respect to dimension weight v_{ik}. The objective function Eq. (1) and the constraint Eq. (2) can be turned into the following Lagrange function:

$$J(\Lambda, V) = \sum_{i=1}^{K}\sum_{x_j\in\chi_i}\sum_{k=1}^{n} v_{ik} D^k_{wc_{ij}} + \sum_{i=1}^{K}\delta_i\sum_{k=1}^{n} v_{ik}^2 - \sum_{i=1}^{K}\lambda_i(\sum_{k=1}^{n} v_{ik} - 1) \quad (3)$$

where Λ=[λ_1, λ_2, $\cdots$, λ_K] is the Lagrange multipliers. Since the rows of V is independent of each other, we can reduce the optimization problem into K independent sub-problems:

$$J_i(\lambda_i, V_i) = \sum_{x_j\in\chi_i}\sum_{k=1}^{n} v_{ik} D^k_{wc_{ij}} + \delta_i\sum_{k=1}^{n} v_{ik}^2 - \lambda_i(\sum_{k=1}^{n} v_{ik} - 1) \quad (4)$$

where V_i is the i-th row of V. Evaluating the gradients of J_i with regard to v_{ik} and λ_i, and set the gradients to zero, we obtain

$$\begin{cases} \frac{\partial J_i(\lambda_i,V_i)}{\partial v_{ik}} = \sum_{x_j\in\chi_i} D^k_{wc_{ij}} + 2\delta_i v_{ik} - \lambda_i = 0; \\ \frac{\partial J_i(\lambda_i,V_i)}{\partial \lambda_i} = (\sum_{k=1}^{n} v_{ik} - 1) = 0. \end{cases} \quad (5)$$

Solving the above group of equations for v_{ik}, we obtain

$$v_{ik} = \frac{1}{n} + \frac{1}{2\delta_i}\sum_{x_j\in\chi_i}[\frac{\sum_{l=1}^{n} D^l_{wc_{ij}}}{n} - D^k_{wc_{ij}}]. \quad (6)$$

Through this equation above, the dimension weights of clusters can be updated to reflect the current dimension relevance to each cluster, given a set of centroids, a set of weights of the last iteration, and a partitioning based on the centroids and weights. The first part of Eq. (6) is $\frac{1}{n}$, which is the default weight if all dimensions are treated equally in a cluster. The second part, which is the sum of differences between the average of individual dimension distances and the individual distances of dimension k, is the bias that takes into account the differences between dimensions. This part can be either positive or negative. A positive value increases that weight, which means that the corresponding dimension is associated with the cluster more closely, for the sum of individual distances of dimension k is less than the sum of the average of all individual distances. Similarly, a negative value of that part means less relevant to the cluster for a dimension. The parameter δ_i in the above equations is important because it is used to weight the relative importance of the second component in Eq. (1). If δ_i is too small, then the contribution of the second part in Eq. (1) will be negligible, and one dimension in cluster i will have a relatively larger weight compared to the other dimensions, which would have a quite small weight or even a zero weight. On the other hand, if δ_i is too large, then almost all dimensions in cluster i will be equally weighted by $\frac{1}{n}$ approximately. Consequently, δ_i is updated iteratively as follows:

$$\delta_i^{(t)} = K_\delta \frac{\sum_{x_j\in\chi_i^{(t-1)}}\sum_{k=1}^{n} v_{ik}^{(t-1)}(D^k_{wc_{ij}})^{(t-1)}}{\sum_{k=1}^{n}(v_{ik}^{(t-1)})^2} \quad (7)$$

where the superscripts (t) and (t-1) are used to indicate the values of the current iteration t and the previous iteration (t-1), respectively, and K_δ is a constant. After updating, we can calculate the weighted aggregate distance between data point x_j and the center of class i, denoted as $\widetilde{D}_{wc_{ij}}$, and assign the data point to a nearest cluster. Subsequently, the updated clusters can be defined as:

$$\chi_i = \{x_j | \widetilde{D}_{wc_{ij}} \leqslant \widetilde{D}_{wc_{kj}} \forall k \neq i\}. \quad (8)$$

After partitioning, a centroid-updating step is carried out, as in the classical K-means. In SKWIC, this is done through the following equation:

$$c_{ik} = \begin{cases} 0 & if \quad v_{ik} = 0 \\ \frac{\sum_{x_j\in\chi_i} x_{jk}}{|\chi_i|} & if \quad v_{ik} > 0 \end{cases} \quad (9)$$

where c_{ik} is the value in the k-th dimension of the new centroid C_i of cluster i.

2.3 Distance Measures

When using SKWIC in document clustering, Frigui *et al.* concluded that Cosine distance is the most suitable distance measure. However, since documents and DNA

sequences have some different characteristics, we try three distance measures, Manhattan distance, Euclidean distance and Cosine distance for MCluster. Definitions of these three distance measures are as follows:

Manhattan distance:

$$D^k_{wc_{ij}} = v_{ik} \times abs(x_{jk} - c_{ik}). \quad (10)$$

where abs() is the function that returns the absolute value.

Euclidean distance:

$$D^k_{wc_{ij}} = v_{ik} \times (x_{jk} - c_{ik})(x_{jk} - c_{ik}). \quad (11)$$

Cosine distance:

$$D^k_{wc_{ij}} = v_{ik} \times (\frac{1}{n} - c_{ik} * x_{jk}). \quad (12)$$

2.4 Performance Evaluation Metrics

To evaluate the clustering results, we consider three performance metrics, namely, *precision*, *sensitivity* and *F-measure*. Assume that a metagenomic dataset comes from N_s species, and finally is grouped into K clusters, R_{ij} represents the number of reads in the i-th cluster that are from species j.

Precision is defined as [23]:

$$precision = \frac{\sum_{i=1}^{K} max_j(R_{ij})}{\sum_{i=1}^{K}\sum_{j=1}^{N_s} R_{ij}}. \quad (13)$$

Sensitivity is defined as [23]

$$sensitivity = \frac{\sum_{j=1}^{N_s} max_i(R_{ij})}{\sum_{i=1}^{K}\sum_{j=1}^{N_s} R_{ij} + the\ number\quad of\quad unclassified\quad reads} \quad (14)$$

where "unclassified reads" denotes the outliers that are excluded from the final clustering result by the clustering algorithm.

F-measure is defined as [29]:

$$F - measure = \frac{2 * precision * sensitivity}{precision + sensitivity}. \quad (15)$$

Precision represents the purity of each cluster; *sensitivity* means the concentration of the reads from each species, while *F-measure* gives the overall performance of the clustering method. There are two extreme cases of the clustering result: one is that all reads are grouped into one single cluster, another is that each read form a single cluster. In the first occasion, *sensitivity* is 1 while *precision* is very small; in the second occasion, *precision* is 1 while *sensitivity* is very small. So neither precision nor sensitivity can be used solely to represent the performance of a clustering algorithm. On the other hand, only when the clustering algorithm perfectly classifies reads from each species exactly into a unique cluster, *F-measure* is 1. In this sense, *F-measure* is used as a comprehensive measure to compare the performances of different clustering algorithms, and it is independent of the number of output clusters. Therefore, in our experiments, we use *F-measure* to measure the overall performances of different clustering algorithms.

2.5 Summary of the MCluster Method

We summarize the MCluster method in Algorithm 1. In Algorithm 1, Line 1 is for vectorizing the reads; Lines 2 – 21 describe the clustering process; Line 22 is for evaluating the clustering result.

Algorithm 1 the Metagenome Clustering algorithm MCluster

Input:
N_r: the number of reads; N_s: the number of species hidden in the sequences; K: the number of clusters; n: the number of dimensions;

Output:
The cluster centroids $\{C_i|i$=1 – $K\}$, reads partitions, and the values of 3 performance metrics;

1: transform each read into a vector of N-gram frequencies
2: initialize K centroids randomly;
3: initialize the partitions using Eq. (8), with all v_{ik} set to $\frac{1}{n}$;
4: **repeat**
5: **for** each $i \in [1..K]$ **do**
6: **for** each $j \in [1..N_r]$ **do**
7: **for** each $k \in [1..n]$ **do**
8: compute the k-th dimension distance $D^k_{wc_{ij}}$ using one of Eqs. (10) – (12);
9: **end for**
10: **end for**
11: **end for**
12: update every v_{ik} with Eq. (6);
13: **for** each $i \in [1..K]$ **do**
14: **for** each $j \in [1..n]$ **do**
15: update the weighted aggregate distance $\widetilde{D}_{wc_{ij}}$;
16: **end for**
17: **end for**
18: update the cluster partitions using Eq. (8);
19: update the centroids using Eq. (9);
20: update δ_i using Eq. (7);
21: **until** (centroids stabilized)
22: Evaluate the performance with 3 performance metrics described in Eqs. (13) – (15)

3 Result

In this section, we evaluate the effectiveness of MCluster on both simulated and real datasets. We also compare our method with MetaCluster 3.0, AbundanceBin and the latest MetaCluster 5.0. For a comprehensive evaluation and a fair comparison, we use both long sequences (1000bp on average) and short reads (128bp on average).

3.1 Datasets

3.1.1 Simulated Datasets

Since there are no commonly used benchmark datasets for NGS metagenomic sequence binning so far, in order to evaluate the performance of our algorithm, we simulate 31 datasets using MetaSim [30] — a tool designed for metagenomic sequences simulation. The 31 datasets are selected to represent metagenomic datasets

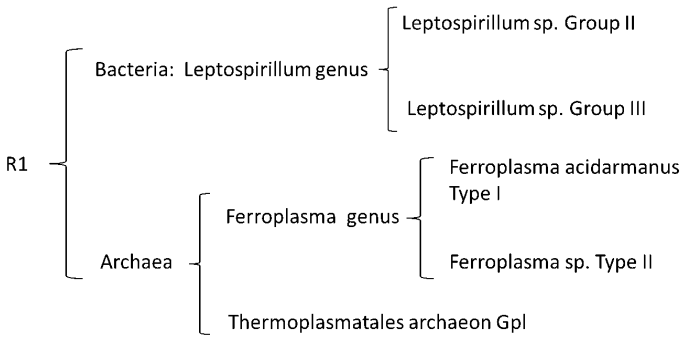

Fig. 2: The taxonomic classification of R1.

of different species numbers, different abundance ratios, different read numbers and lengths. Among them, 16 datasets, denoted by from D1 to D16, are long reads of low abundance, with an average length of 1000bp and read number of 5k and 50k; the other 15 datasets denoted by from S1 to S15 are of relatively-high abundance, where S1 - S10 contain long reads of 1000bp on average, their read numbers are 50k and 500k, S11 - S15 consist of short reads of 128bp on average, their read numbers are 8000k. More details of the 31 datasets are presented in Table 1 and Table 2, respectively.

3.1.2 Real Dataset

We download an Acid Mine Drainage metagenomic dataset from NCBI (http://www.ncbi.nlm.nih.gov/books/NBK6860/), denoted as R1, to test the performance of MCluster. R1 incorporates 2534 contigs with an average length of 5000bp, which are assembled from 103,462 high-quality trimmed reads [8]. The dataset includes annotated sequences from 5 known species: Leptospirillum sp. Group II, Leptospirillum sp. Group III, Ferroplasma acidarmanus Type I, Ferroplasma sp. Type II and Thermoplasmatales archaeon Gpl, as well as some sequences from unknown species. The taxonomic classification of the five species is showed in Fig. 2, which can be classified into two superkingdoms and three genera. Since the original reads do not have species annotations, we use the 2534 annotated contigs to test the clustering performances of our method and two existing methods (AbundanceBin and MetaCluster 3.0).

3.2 Effect of Distance Measure

Since distance measure may affect the performance of clustering algorithms, before performing clustering evaluation, we conduct experiments to select an appropriate distance measure for MCluster. We test the performance of MCluster on D9, D10, D12 and D13 datasets using different distance measures, and the results are presented in Fig. 3. The results show that Manhattan distance achieves the best overall performance among the three distance measures. Although Cosine distance performs very well when applied to document clustering using the original SKWIC algorithm [28], and achieves the best sensitivity in our experiments, its precision and overall performance in metagenomic sequence binning is the worst among the three distance measures. Therefore, in the following experiments, we use Manhattan distance as the distance measure in MCluster to cluster metagenomic reads.

3.3 Experimental Results on Simulated datasets

For a comparative evaluation, we compare our method with three state of the art unsupervised binning methods: AbundanceBin, MetaCluster 3.0 and MetaCluster 5.0. MetaCluster 3.0 and AbundanceBin work well with only long reads. In addition, AbundanceBin works better with high-abundance datasets. MetaCluster 5.0 is the latest one of the series of MetaCluster methods, it was designed for binning short pair-end reads from species with different sequence abundance ratios. So we compare our method with AbundanceBin and MetaCluter-3.0 on long reads, and with MetaCluster 5.0 on short reads.

3.3.1 Experiments on Long Reads Datasets

We first test and compare the performances of MCluster and MetaCluster 3.0 on the 16 simulated datasets with long reads. Since both MCluster and MetaCluster 3.0 are based on the K-means algorithm, which randomly initiates the cluster centers, we repeat each experiment 50 times and compute the average performance.

To evaluate the performances of MCluster and MetaCluster 3.0 on datasets with balanced abundance ratio, we compare their performances on four evenly distributed datasets: D1, D8, D11 and D13. Experimental results are shown in Fig. 4. Each of the four datasets contains the same number of reads from different species. As showed in Fig. 4, when applied to the 4 evenly distributed datasets with different numbers of species ranging from 2 to 10, MCluster achieves larger precision and better overall performance than MetaCluster 3.0 in all the four datasets. It can also be observed that the performances of MetaCluster 3.0 and MCluster are influenced by the number of species in the datasets.

Unevenly distributed datasets with different abundance ratios pose a serious challenge to metagenomic sequence clustering, because algorithms such as K-means tend to group the data into similar-size clusters [12]. MetaCluster 3.0 tries to solve the problem by first setting a large K value and then merging similar small clusters. In order to evaluate MCluster's ability to deal with unevenly distributed data, we compare MCluster with MetaCluster 3.0 by using datasets from D1 to D7 with abundance ratios of 1:1, 1:2, 1:4, 1:6, 1:8, 1:10 and 1:12, where the minority genome's DNA fragments are about from 8% to 50% of the total sequences. The results are showed in Fig. 5. On these 7 datasets,

TABLE 1: Simulated long-read datasets of low abundance

Dataset	#Reads	Read length	#Species	Abundance ratio	Species makeup
D1	5k	1000bp	2	1:1	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
D2	5k	1000bp	2	1:2	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
D3	5k	1000bp	2	1:4	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
D4	5k	1000bp	2	1:6	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
D5	5k	1000bp	2	1:8	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
D6	5k	1000bp	2	1:10	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
D7	5k	1000bp	2	1:12	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
D8	5k	1000bp	3	1:1:1	*Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Francisella_tularensis_subsp._tularensis_SCHU_S4*
D9	5k	1000bp	3	1:3:9	*Pseudomonas_aeruginosa_PAO1, Legionella_pneumophila_str._Lens, Cycloclasticus_sp._P1*
D10	5k	1000bp	4	1:3:3:9	*Pseudomonas_aeruginosa_PAO1, Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1*
D11	5k	1000bp	5	1:1:1:1:1	*Pseudomonas_aeruginosa_PAO1, Shewanella_baltica_OS155, Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Francisella_tularensis_subsp._tularensis_SCHU_S4*
D12	5k	1000bp	5	1:1:3:3:9	*Pseudomonas_aeruginosa_PAO1, Marinobacter_sp._BSs20148, Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1*
D13	5k	1000bp	10	1:1:1:1:1:1:1:1:1:1	*Pseudomonas_aeruginosa_PAO1, Marinobacter_sp._BSs20148, Chromohalobacter_salexigens_DSM_3043, Legionella_pneumophila_str, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1, Salmonella_typhimurium_LT2, Xanthomonas_oryzae_pv._oryzae_KACC10331, Aeromonas_salmonicida_subsp._salmonicida_A449, Vibrio_cholerae_O395*
D14	50k	1000bp	3	1:3:9	*Pseudomonas_aeruginosa_PAO1, Legionella_pneumophila_str._Lens, Cycloclasticus_sp._P1*
D15	50k	1000bp	4	1:3:3:9	*Pseudomonas_aeruginosa_PAO1, Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1*
D16	50k	1000bp	5	1:1:3:3:9	*Pseudomonas_aeruginosa_PAO1, Marinobacter_sp._BSs20148, Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1*

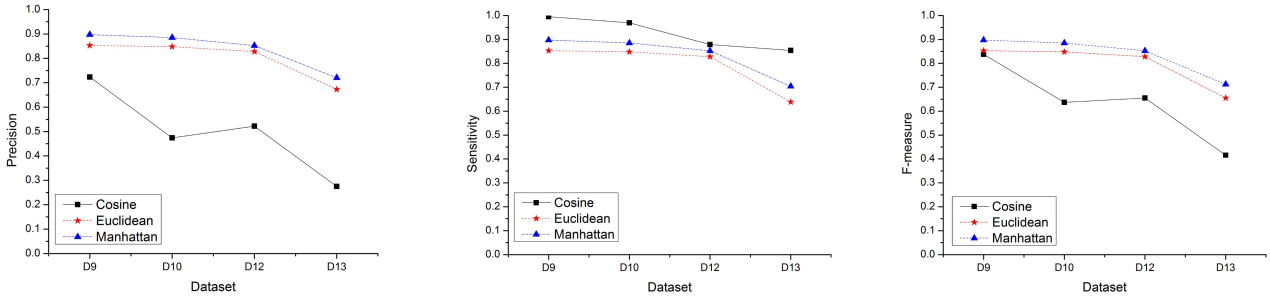

Fig. 3: The effect of distance measure on the performance of MCluster.

TABLE 2: Simulated datasets of relatively-high abundance

Dataset	#Reads	Read length	#Species	Abundance ratio	Species makeup
S1	50k	1000bp	2	1:2	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
S2	50k	1000bp	3	1:3:9	*Pseudomonas_aeruginosa_PAO1, Legionella_pneumophila_str._Lens Cycloclasticus_sp._P1*
S3	50k	1000bp	3	1:1:1	*Legionella_pneumophila_str._Lens,Nitrosococcus_oceani_ATCC_19707, Francisella_tularensis_subsp._tularensis_SCHU_S4*
S4	50k	1000bp	5	1:1:3:3:9	*Pseudomonas_aeruginosa_PAO1,Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1*
S5	50k	1000bp	10	1:1:1:1:1:1:1:1:1:1	*Pseudomonas_aeruginosa_PAO1, Marinobacter_sp._BSs20148, Chromohalobacter_salexigens_DSM_3043, Legionella_pneumophila_str, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1, Salmonella_typhimurium_LT2, Xanthomonas_oryzae_pv._oryzae_KACC10331, Aeromonas_salmonicida_subsp._salmonicida_A449, Vibrio_cholerae_O395*
S6	500k	1000bp	2	1:2	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
S7	500k	1000bp	3	1:3:9	*Pseudomonas_aeruginosa_PAO1, Legionella_pneumophila_str._Lens, Cycloclasticus_sp._P1*
S8	500k	1000bp	3	1:1:1	*Legionella_pneumophila_str._Lens,Nitrosococcus_oceani_ATCC_19707, Francisella_tularensis_subsp._tularensis_SCHU_S4*
S9	500k	1000bp	5	1:1:3:3:9	*Pseudomonas_aeruginosa_PAO1, Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1*
S10	500k	1000bp	10	1:1:1:1:1:1:1:1:1:1	*Pseudomonas_aeruginosa_PAO1, Marinobacter_sp._BSs20148, Chromohalobacter_salexigens_DSM_3043, Legionella_pneumophila_str, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1, Salmonella_typhimurium_LT2, Xanthomonas_oryzae_pv._oryzae_KACC10331, Aeromonas_salmonicida_subsp._salmonicida_A449, Vibrio_cholerae_O395*
S11	8000k	128bp	2	1:1	*Pseudomonas_aeruginosa_PAO1, Cycloclasticus_sp._P1*
S12	8000k	128bp	3	1:3:9	*Pseudomonas_aeruginosa_PAO1, Legionella_pneumophila_str._Lens, Cycloclasticus_sp._P1*
S13	8000k	128bp	3	1:1:1	*Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Francisella_tularensis_subsp._tularensis_SCHU_S4*
S14	8000k	128bp	5	1:1:3:3:9	*Pseudomonas_aeruginosa_PAO1, Legionella_pneumophila_str._Lens, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1*
S15	8000k	128bp	10	1:1:1:1:1:1:1:1:1:1	*Pseudomonas_aeruginosa_PAO1, Marinobacter_sp._BSs20148, Chromohalobacter_salexigens_DSM_3043, Legionella_pneumophila_str, Nitrosococcus_oceani_ATCC_19707, Cycloclasticus_sp._P1, Salmonella_typhimurium_LT2, Xanthomonas_oryzae_pv._oryzae_KACC10331, Aeromonas_salmonicida_subsp._salmonicida_A449, Vibrio_cholerae_O395*

MCluster achieves similar precision but obviously better sensitivity and overall performance, in comparison with MetaCluster 3.0. Moreover, MCluster performs stably for various abundance ratios, proving that it can be applied to datasets with both identical and biased abundance ratios.

We also test the performances of MCluster and MetaCluster 3.0 on multi-species unbalanced datasets: D9, D10 and D12, with abundance ratios of 1:3:9, 1:3:3:9 and 1:1:3:3:9, respectively. The results are illustrated in Fig. 6. As shown in Fig. 6, although the sensitivity of MetaCluster 3.0 is slightly larger, MCluster is better in precision and overall performance on all the three datasets. This proves that MCluster has the ability to effectively cluster multi-species unbalanced meta-genomic sequence data.

The number of sequencing reads in a dataset represents the coverage of the sequencing experiment, which also has considerable influence on clustering performance. Therefore, we test and compare the performances of MCluster and MetaCluster on 3 relatively-high coverage datasets with 50000 reads: D14, D15 and D16. Note that the only difference between these three datasets and the other three datasets (D9, D10 and D12) tested above lies in that the formers have 10 times of sequence

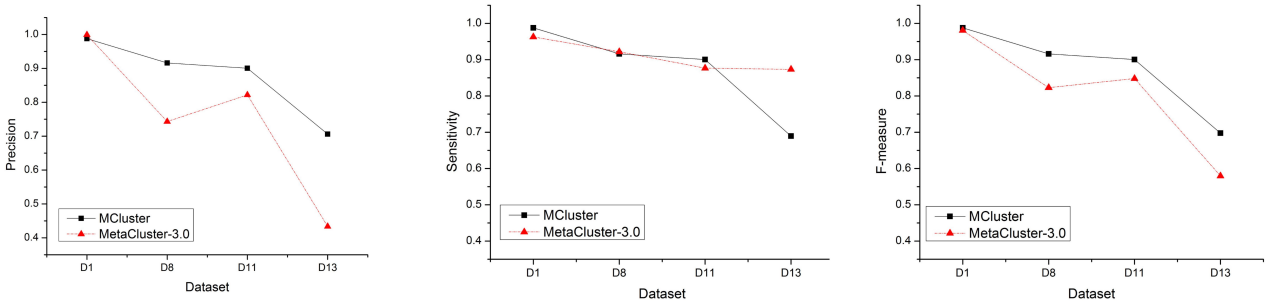

Fig. 4: The performances of MetaCluster 3.0 and MCluster on evenly distributed datasets.

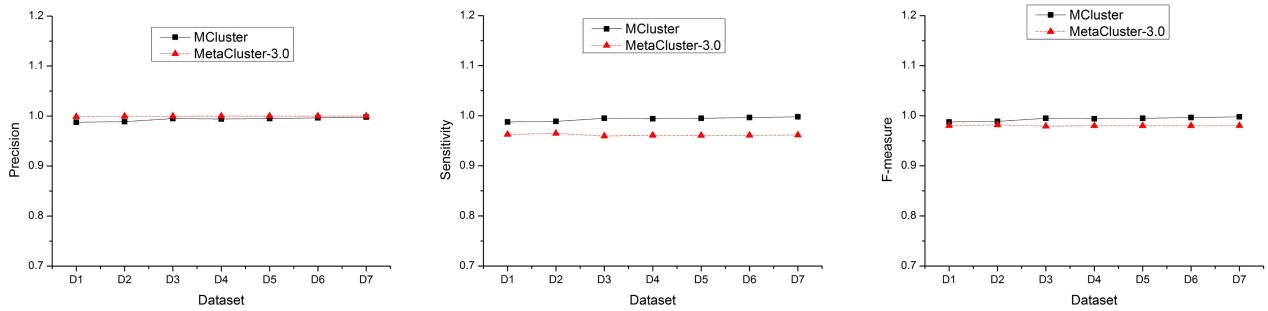

Fig. 5: The performances of MetaCluster 3.0 and MCluster on uneven distributed datasets with two species.

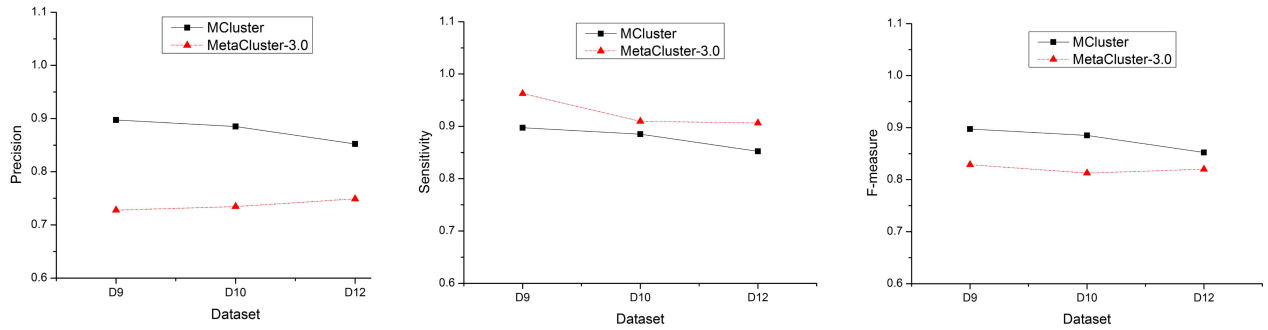

Fig. 6: The performances of MetaCluster 3.0 and MCluster on multi-species unbalanced datasets.

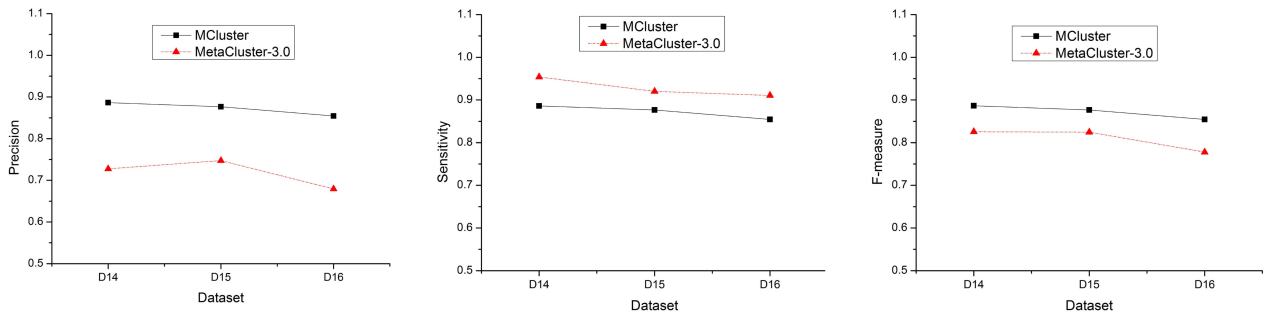

Fig. 7: The performances of MetaCluster 3.0 and MCluster on relatively-high coverage datasets.

reads contained in the latters. Experimental results are shown in Fig. 7. It is clear that the overall performance of MCluster is still better than that of MetaCluster 3.0 on relatively-high abundance datasets, which is consistent with the experimental results on low abundance datasets showed in Fig. 6.

We then evaluate and compare the performances of MCluster and AbundanceBin [21]. As AbundanceBin can only deal with high-abundance datasets, we test their performances on ten datasets, in which five datasets (S1 - S5) contain 50k reads and the other five datasets (S6 - S10) contain 500k reads. The results are shown in Fig. 8 and Fig. 9, respectively. We can see that for both high-abundance datasets (500k reads) and relatively-high abundance datasets (50k reads), our method achieves better precision and F-measure than AbundanceBin. But the sensitivity of MCluster is smaller than that of AbundanceBin on four of the ten tested datasets, which may be attributed to the fact that the number of nonempty clusters output by AbundanceBin is less than the real species number, even when we set the input number of clusters for AbudanceBin to the real number of species.

3.3.2 Experiments on Short Reads Datasets

We also compare the performance of MCluster with that of MetaCluster 5.0 [23]. As MetaCluster 5.0 works only for short reads, we use the reads about 128bp (datasets S11 - S15) to test them. The results are shown in Fig. 10. We can see that the MCluster achieves a larger sensitivity than MetaCluster 5.0. This is possibly because MetaCluster 5.0 classifies many reads as extremely-low abundance reads and abandons them during the clustering process. However, MetaCluster 5.0 has a higher precision than MCluster. As a result of tradeoff between precision and sensitivity, our method obtains a larger F-measure than MetaCluster 5.0 on two of the five datasets. It is interesting to notice that MetaCluster 5.0 performs badly on the dataset S15 that has the largest number of species, while our method MCluster has a relatively stable F-measure on the five tested datasets. In summary, compared with MetaCluster 5.0, the experimental results suggest that our method is able to achieve comparable yet more stable overall performance in binning short reads.

3.4 Experimental Results on A Real Dataset

Here we present the results on a real dataset R1 described in Sec. 3.1.2. We predefine the number of clusters according to the input sequences for AbundanceBin. Since MetaCluster 3.0 has a bottom-up merging step, the final number of clusters output by it can not be predefined. In our experiment, MetaCluster 3.0 groups the dataset into three clusters. This can be explained as a result of clustering in genus level, as showed in Fig. 2. Since sequences in R1 dataset can also be classified into two superkingdoms (Bacteria and Archaea) or five species, we set the number of clusters to 2, 3 and 5 for MCluster, to cluster the dataset at superkingdom, genus and species levels, respectively. The clustering performances of the three methods are summarized in Table 3.

TABLE 3: The performances of MCluster, MetaCluster-3.0 and AbundanceBin on the real dataset R1

Method	#Clusters	Precision	Sensitivity	F-measure
MetaCluster-3.0	3	0.7054	0.7403	0.7224
MCluster	2	0.6748	0.9562	0.7912
MCluster	3	0.676	0.923	0.7804
MCluster	5	0.6819	0.7833	0.7291
AbundanceBin	2	0.3733	0.9838	0.5412

As showed in Table 3, when clustering at superkingdom and genus levels, MCluster significantly outperforms MetaCluster 3.0 in sensitivity and F-measure, with only a slightly smaller precision. At species level, MCluster still achieves slightly better sensitivity and overall performance than MetaCluster 3.0. While comparing with AbundanceBin for the case of two clusters, our method achieves much larger precision and F-measure, but slightly smaller sensitivity.

The clustering performance comparison at different levels reveals that the clustering level impacts the clustering performance. In our experiment, clustering at superkingdom level achieves the best overall performance among all the three taxonomic levels, and clearly separates the sequences of Bacteria from that of Archaea. The success of superkingdom level clustering may be attributed to the specific characteristics of R1 dataset. As illustrated in Fig. 2, the two Bacteria in R1 belongs to the same genus, while the other three Archaea all belongs to the same order. Since distance within the same genus or the same order is much smaller than the distance between superkingdoms, it is reasonable to cluster the dataset at superkingdom level rather than at lower levels.

Moreover, as pointed out in Sec. 3.1.2, there are also a few sequences from unknown species in dataset R1. These sequences are unclassified because there is not enough evidence to classify them into any known species [8]. However, Tyson *et al*. mentioned that the sample seems to contain sequences from 3 bacteria. Our experimental results in Table 4 validates this judgment: 95% of the unclassified sequences are grouped into Cluster 1 that is dominated by "bacteria" sequences, which suggests that these sequences might belong to some unknown bacterium in the sample. In this sense, our method provides valuable insights into the real taxonomic classification of unknown sequences.

4 Discussion

With all the experiments conducted in this study, we find that the incorporation of automatic feature

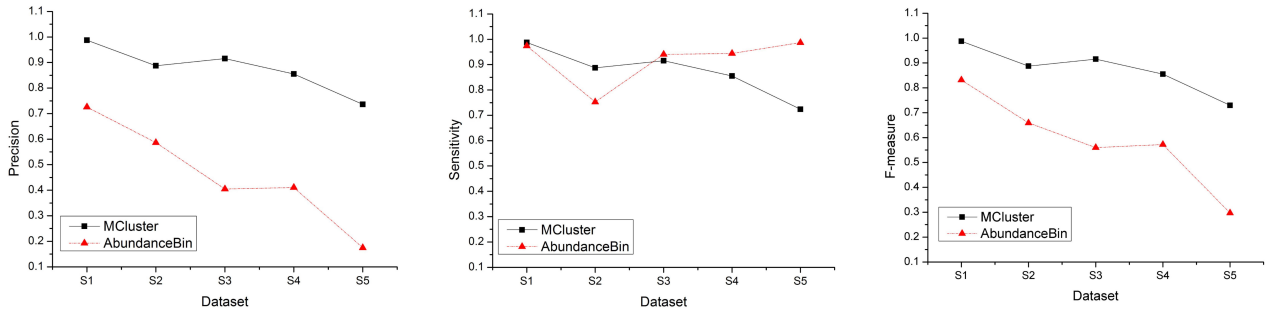

Fig. 8: The performances of MCluster and AbundanceBin on relatively-high abundance datasets of 50k reads.

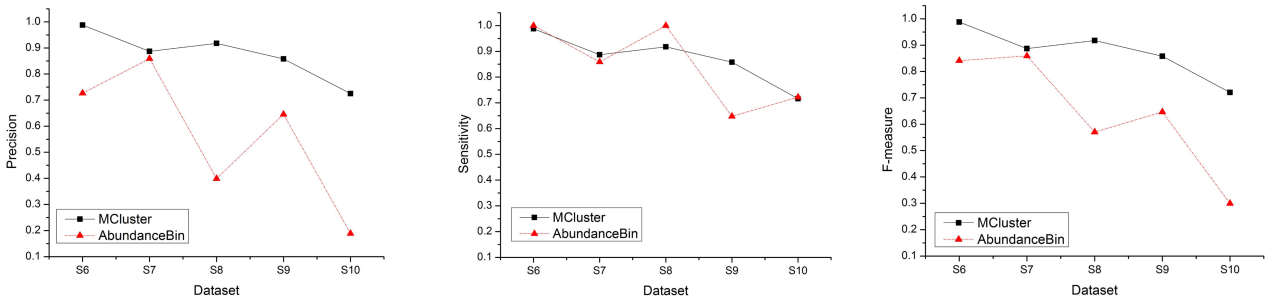

Fig. 9: The performances of MCluster and AbundanceBin on high abundance datasets of 500k reads.

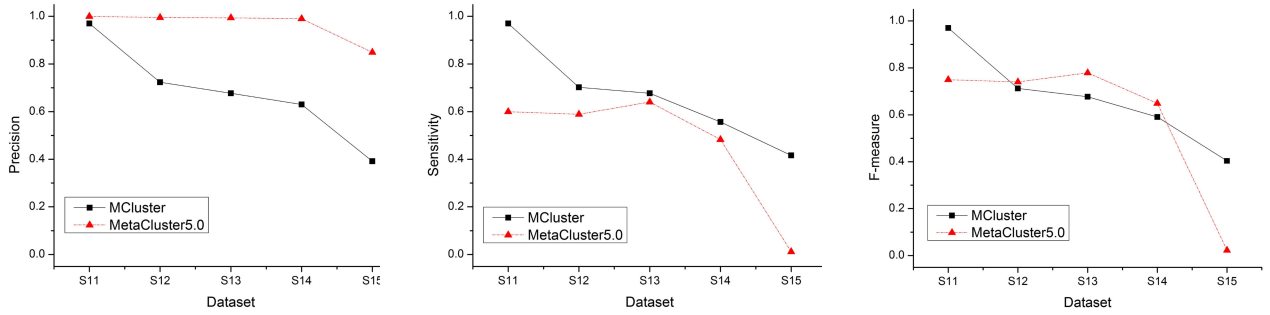

Fig. 10: The performances of MCluster and MetaCluster 5.0 on high abundance datasets of 8000k reads about 128bp.

TABLE 4: The clustering result of R1 dataset at superkingdom level.

	Bacteria	Archaea	Unknown sequences	Total size
Cluster1	927	2	104	1033
Cluster2	111	1385	5	1501

weighting mechanism to the clustering algorithm can significantly improve the performance of metagenomic sequence clustering. While the basic K-means algorithm tends to separate data items into similar-size clusters [12], the feature weighting algorithm mitigates this drawback and is as effective as the bottom-up merging step used by MetaCluster 3.0 in dealing with unbalanced datasets.

The performance of MCluster is also related to an appropriate distance measure. Although Cosine distance performs best in document clustering and achieves good sensitivity in our experiments, we find that Manhattan distance is more suitable for metagenomic sequence binning.

While MetaCluster 3.0 and its improved versions [23] use a 136-dimension vector to represent each read, MCluster follows many existing binning methods [16], [20], [31] to represent each read as a 256-dimension vector. To evaluate the possible impact of vectorization scheme on clustering, we compare the clustering performances of the two different vectorization schemes with our method, the results are shown in Fig. 11. Obviously, it seems that the size of vector (136 or 256) has little impact on the final performance. Such a result is reasonable, because the two representation schemes keep

the same amount of information of the reads, though they use vectors of different lengths. However, shorter vectors benefit clustering efficiency.

We compare the performances of MCluster and MetaCluster 3.0 on 16 simulated datasets (D1 – D16). On all the datasets, MCluster achieves better overall performance than MetaCluster 3.0. And in most cases, MCluster also has better precision than MetaCluster 3.0. However, on some datasets, MetaCluster 3.0 achieves better sensitivity than MCluster. This may be attributed to its cluster-merging step. With this step, the cluster number output by MetaCluster 3.0 is often less than the real number of species hidden in reads, which leads to a larger sensitivity. Table 5 shows the output numbers of clusters by MetaCluster 3.0 on 16 datasets. We can see that when the datasets contain reads from more than 2 species, MetaCluster 3.0 often outputs a smaller number of clusters than the real number of species. For example, MetaCluster 3.0 detects only 4 of the 10 species in D13.

TABLE 5: The number of clusters output by MetaCluster 3.0 on 16 simulated datasets

Dataset	#Species	#output-clusters
D1	2	2
D2	2	2
D3	2	2
D4	2	2
D5	2	2
D6	2	2
D7	2	2
D8	3	2
D9	3	2
D10	4	3
D11	5	4
D12	5	4
D13	10	4
D14	3	3
D15	4	3
D16	5	3

When compared with AbundanceBin on 10 relatively-high abundance datasets with long reads (S1 – S10), our method achieves larger precision and better overall performance, which suggests that our method can accurately cluster both even-distributed and uneven-distributed data.

While compared with MetaCluster 5.0 on high-abundance datasets with short reads, our method also achieves better sensitivity on all five tested datasets (S11 – S15), and better overall performance on two of the five datasets. More importantly, MCluster performs more stably than MetaCluster 5.0 in overall. This result suggests that by using an automatic feature weighting scheme, MCluster can also effectively handle short reads.

The experiment on real dataset implies that the performance of clustering algorithms depends on the characteristics of the tested dataset. If the dataset contains data from closely related species, taking the related species as one class and do clustering at a higher level may be more appropriate, and thus achieves better performance.

Although MCluster achieves considerable good performance for metagenomic sequence binning, there is still much space for improvement in the future:

On the one hand, the cluster number has to be set before clustering, while in many cases the actual number of species in the dataset is unknown. The problem is more complicated when doing clustering at different taxonomic levels. Unfortunately, up to now, there is not any effective computational method to automatically and accurately determine the number of species in a sample without using reference genome information. This is a common challenge to all unsupervised binning methods. Although MetaCluster 3.0 tried to determine the cluster number automatically, as shown in Table 5, we find that its output cluster number is not correct in many cases. To solve the problem, before binning, experimental methods such as 16S ribosomal RNA gene clone library construction can be used to determine the species number in a sample [8], [32]. After the species number is determined, MCluster can be used to effectively determine the origins of short reads in the sample automatically without using reference genome information.

On the other hand, since 454 is the most widely used second-generation-sequencing platform in metagenomic study so far [9], and the upcoming generation of sequencing such as Pacific Bio and Oxford Nanopore Technologies will also output long reads [33], we have shown that MCluster performs well on metagenomic datasets with long reads. However, there are still many metagenomic datasets containing reads shorter than 200bp. The main challenge to solve the short reads binning problem is that for short reads, fewer N-grams can be extracted, which leads to sparse representations of reads. MetaCluster 4.0/5.0 uses a pregrouping step to solve this problem. In our method, though the feature weighting mechanism can mitigate this problem in some extent, in the future we plan to employ a pregrouping step as used by MetaCluster 4.0/5.0 to further improve the performance of MCluster for clustering short reads.

5 Conclusion

To summarize, in this paper, we aim at solving the binning problem of unknown metagenomic sequences without using reference genomes. Since traditional similarity-based and supervised composition-based methods cannot be applied to this problem, we present a new unsupervised composition-based method called MCluster to tackle this problem. The incorporation of automatic feature weighting mechanism enables MCluster to handle both balanced and unbalanced datasets with long or short reads. MCluster achieves clearly better overall performance than AbundanceBin and MetaCluster 3.0 on both simulated and real datasets, and comparable overall performance to MetaCluster 5.0 on five simulated datasets. The proposed method can

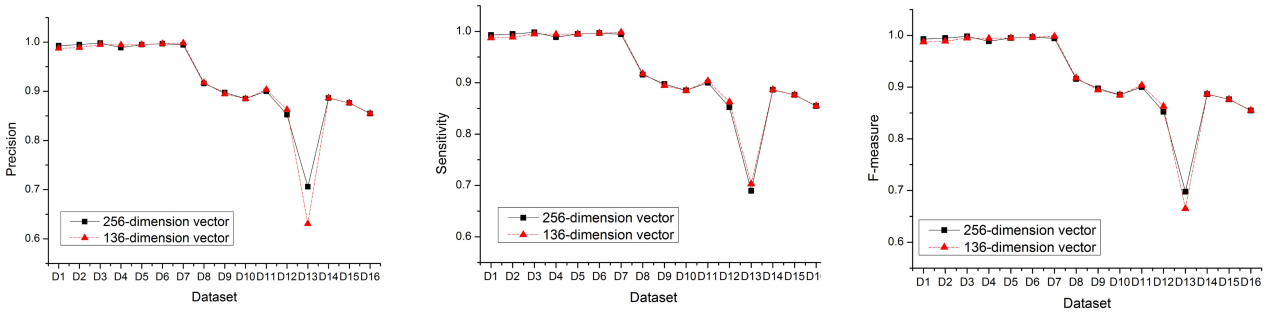

Fig. 11: The performance comparison between 256-dim vectors and 136-dim vectors with MCluster.

thus be used as a promising tool for characterizing the compositions of unknown microbial communities.

Acknowledgments

We appreciate Yuan Yi's help in implementing the MCluster method. We thank the authors of AbundanceBin and MetaCluster 3.0/5.0 for providing the tools to do comparison study. This work was partially supported by National Natural Science Foundation of China (NSFC) under grants No. 61173118 and No. 61272380.

References

[1] J. C. Wooley, A. Godzik, and I. Friedberg, "A primer on metagenomics," *PLoS Computational Biology*, 6(2): e1000667, 2010.

[2] J. Qin, R. Li, J. Raes, M. Arumugam, K. S. Burgdorf, C. Manichanh, T. Nielsen, N. Pons, F. Levenez, and T. Yamada, "A human gut microbial gene catalogue established by metagenomic sequencing," *Nature*, 464(7285): 59-65, 2010.

[3] R. Daniel, "The metagenomics of soil," *Nature Reviews Microbiology*, 3(6): 470-478, 2005.

[4] S. C. Schuster, "Next-generation sequencing transforms today's biology," *Nature*, 200(8): 16-18, 2008.

[5] E. K. Costello, C. L. Lauber, M. Hamady, N. Fierer, J. I. Gordon, and R. Knight, "Bacterial community variation in human body habitats across space and time," *Science*, 326(5960): 1694-1697, 2009.

[6] E. A. Grice, H. H. Kong, S. Conlan, C. B. Deming, J. Davis, A. C. Young, G. G. Bouffard, R. W. Blakesley, P. R. Murray, and E. D. Green, "Topographical and temporal diversity of the human skin microbiome," *Science*, 324(5931): 1190-1192, 2009.

[7] D. B. Rusch, A. L. Halpern, G. Sutton, K. B. Heidelberg, S. Williamson, S. Yooseph, D. Wu, J. A. Eisen, J. M. Hoffman, and K. Remington, "The Sorcerer II global ocean sampling expedition: northwest Atlantic through eastern tropical Pacific," *PLoS biology*, 5(3): e77, 2007.

[8] 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*, 428(6978): 37-43, 2004.

[9] H. Teeling, and F. O. Glöckner, "Current opportunities and challenges in microbial metagenome analysisła bioinformatic perspective," *Briefings in Bioinformatics*, 13(6): 728-742, 2012.

[10] J. Dröge, and A. C. McHardy, "Taxonomic binning of metagenome samples generated by next-generation sequencing technologies," *Briefings in Bioinformatics*, 13(6): 646-655, 2012.

[11] D. H. Huson, D. C. Richter, S. Mitra, A. F. Auch, and S. C. Schuster, "Methods for comparative metagenomics," *BMC Bioinformatics*, 10(S12), 2009.

[12] H. C. Leung, S. Yiu, B. Yang, Y. Peng, Y. Wang, Z. Liu, J. Chen, J. Qin, R. Li, and F. Y. Chin, "A robust and accurate binning algorithm for metagenomic sequences with arbitrary species abundance ratio," *Bioinformatics*, 27(11): 1489-1495, 2011.

[13] E. A. Dinsdale, O. Pantos, S. Smriga, R. A. Edwards, F. Angly, L. Wegley, M. Hatay, D. Hall, E. Brown, and M. Haynes, "Microbial ecology of four coral atolls in the Northern Line Islands," *PloS one*, 3(2): e1584, 2008.

[14] A. C. McHardy, H. G. Martin, A. Tsirigos, P. Hugenholtz, and I. Rigoutsos, "Accurate phylogenetic classification of variable-length DNA fragments," *Nature Methods*, 4(1): 63-72, 2006.

[15] M. Stark, S. Berger, A. Stamatakis, and C. von Mering, "MLTreeMap-accurate Maximum Likelihood placement of environmental DNA sequences into taxonomic and functional reference phylogenies," *BMC Genomics*, 11: 1, 2010.

[16] N. N. Diaz, L. Krause, A. Goesmann, K. Niehaus, and T. W. Nattkemper, "TACOA-Taxonomic classification of environmental genomic fragments using a kernelized nearest neighbor approach," *BMC Bioinformatics*, 10:1, 2009.

[17] A. Brady, and S. L. Salzberg, "Phymm and PhymmBL: metagenomic phylogenetic classification with interpolated Markov models," *Nature Methods*, 6(9): 673-676, 2009.

[18] T. Abe, H. Sugawara, M. Kinouchi, S. Kanaya and T. Ikemura, "Novel phylogenetic studies of genomic sequence fragments derived from uncultured microbe mixtures in environmental and clinical samples," *DNA Research*, 12: 281-290, 2005.

[19] T. Abe, H. Sugawara, S. kanaya, T. Ikemura, "A novel bioinformatics tool for phylogenetic classification of genomic sequence fragments derived from mixed genomes of uncultured environmental microbes," *Polar Biosci*, 20: 103-112, 2006.

[20] C. Chan, A. Hsu, S. Halgamuge, S. Tang, "Binning sequences using very sparse labels within a metagenome," *BMC Bioinformatics*, 9: 215, 2008.

[21] Y. W. Wu, and Y. Ye, "A novel abundance-based algorithm

for binning metagenomic sequences using l-tuples," *Journal of Computational Biology*, 18(3): 523C534, 2011.
[22] Y. Wang, H. C. Leung, S. Yiu, and F. Y. Chin, "MetaCluster 4.0: a novel binning algorithm for NGS reads and huge number of species," *Journal of Computational Biology*, 19(2): 241-249, 2012.
[23] Y. Wang, H. C. Leung, S. Yiu, and F. Y. Chin, "MetaCluster 5.0: a two-round binning approach for metagenomic data for low-abundance species in a noisy sample," *Bioinformatics*, 28(18): i356-i362, 2012.
[24] S. Karlin, J. Mrazek, and A. M. Campbell, "Compositional biases of bacterial genomes and evolutionary implications," *Journal of Bacteriology*, 179(12): 3899-3913, 1997.
[25] Y. Yi, J. Guan, and S. Zhou, "Effective clustering of microRNA sequences by N-grams and feature weighting," *Proceedings of IEEE 6th International Conference on System Biology (ISB'12)*, pp. 203-210, 2012.
[26] B. Chor, D. Horn, N. Goldman, Y. Levy, and T. Massingham, "Genomic DNA k-mer spectra: models and modalities," *Genome Biol.*, 10(10): R108, 2009.
[27] F. Zhou, V. Olman, and Y. Xu, "Barcodes for genomes and applications," *BMC Bioinformatics*, 9:1, 2008.
[28] H. Frigui, and O. Nasraoui, "Simultaneous clustering and dynamic keyword weighting for text documents," Survey of text mining, Michael Berry (Ed.), Springer, pp.45-70, 2004.
[29] B. Larsen, and C. Aone, "Fast and effective text mining using linear-time document clustering," *Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'99)*, pp. 16-22, 2009.
[30] D. C. Richter, F. Ott, A. F. Auch, R. Schmid, and D. H. Huson, "MetaSim-A sequencing simulator for genomics and metagenomics," *PloS one*, 3(10): e3373, 2008.
[31] H. Teeling, J. Waldmann, T. Lombardot, M. Bauer, F.O. Glöckner, "TETRA: a web-service and a stand-alone program for the analysis and comparison of tetranucleotide usage patterns in DNA sequences," *BMC Bioinformatics*, 5: 163, 2004.
[32] V. Kunin, A. Copeland, A. Lapidus, K. Mavromatis, and P. Hugenholtz, "A bioinformatician's guide to metagenomics," *Microbiology and Molecular Biology Reviews*, 72(4): 557-578, 2008.
[33] E. E. Schadt, S. Turner, and A. Kasarskis, "A window into third-generation sequencing," *Human molecular genetics*, vol. 19, no. R2, pp. R227-R240, 2010.

Liao Ruiqi received his Bachelor degree of Biology in 2010 from East China Normal University, Shanghai China, and is now a master student in School of Computer Science, Fudan University, Shanghai, China. His research interests include data mining, Bioinformatics and MapReduce based algorithms.

Ruichang Zhang received his Bachelor degree in Computer Science and Technology in 2012 from Shanghai University, China, and is now a master student in School of Computer Science, Fudan University, Shanghai, China. His research interests include data mining, machine learning and Bioinformatics.

Jihong Guan is now a professor at the Department of Computer Science Technology, Tongji University, Shanghai, China. She received his Bachelor degree from Huazhong Normal University in 1991, her Master degree from Wuhan Technical University of Surveying and Mapping (merged into Wuhan University since Aug. 2000) in 1991, and her PhD from Wuhan University in 2002. Before joining Tongji University, she served in the Department of Computer, Wuhan Technical University of Surveying and Mapping from 1991 to 1997, as an assistant professor and an associate professor (since August 2000) respectively. She was an associate professor (Aug. 2000-Oct. 2003) and a professor (Since Nov. 2003) in the School of Computer, Wuhan University. Her research interests include databases, data mining, distributed computing, Bioinformatics, and geographic information systems (GIS). She has published more than 100 papers in domestic and international journals and conferences.

Shuigeng Zhou is now a professor at the School of Computer Science, Fudan University, Shanghai, China. He received his Bachelor degree from Huazhong University of Science and Technology (HUST) in 1988, his Master degree from University of Electronic Science and Technology of China (UESTC) in 1991, and his PhD of Computer Science from Fudan University in 2000. He served in Shanghai Academy of Spaceflight Technology from 1991 to 1997, as an engineer and a senior engineer (since August 1995) respectively. He was a postdoctoral researcher in State Key Lab of Software Engineering, Wuhan University from 2000 to 2002. His research interests include data management, data mining and Bioinformatics. He has extensively published in domestic and international journals (including IEEE TKDE, IEEE TPDS, IEEE TCBB, IEEE TGRS, DKE and Bioinformatics etc.) and conferences (including SIGMOD, SIGKDD, SIGIR, VLDB, ICDE, IJCAI, SODA and RECOMB etc.). Currently he is a member of IEEE, ACM and IEICE.