Briefings in Bioinformatics, 00(00), 2018, 1–10

doi: 10.1093/bib/bby090
Advance Access Publication Date: 18 September 2018
Review Article

Sequence clustering in bioinformatics: an empirical study

Quan Zou, Gang Lin, Xingpeng Jiang, Xiangrong Liu and Xiangxiang Zeng

Corresponding authors: Xiangrong Liu and Xiangxiang Zeng, School of Information Science and Technology, Xiamen University, Xiamen 361005, China. E-mail: xrliu@xmu.edu.cn; xzeng@xmu.edu.cn

Abstract

Sequence clustering is a basic bioinformatics task that is attracting renewed attention with the development of metagenomics and microbiomics. The latest sequencing techniques have decreased costs and as a result, massive amounts of DNA/RNA sequences are being produced. The challenge is to cluster the sequence data using stable, quick and accurate methods. For microbiome sequencing data, 16S ribosomal RNA operational taxonomic units are typically used. However, there is often a gap between algorithm developers and bioinformatics users. Different software tools can produce diverse results and users can find them difficult to analyze. Understanding the different clustering mechanisms is crucial to understanding the results that they produce. In this review, we selected several popular clustering tools, briefly explained the key computing principles, analyzed their characters and compared them using two independent benchmark datasets. Our aim is to assist bioinformatics users in employing suitable clustering tools effectively to analyze big sequencing data. Related data, codes and software tools were accessible at the link http://lab.malab.cn/~lg/clustering/.

Key words: operational taxonomic unit; 16S ribosomal RNA; microbiome; sequence clustering; sequence redundancy removal

Introduction

Classification and clustering are two main tasks in machine learning research. There are two different types of machine learning tasks, supervised and unsupervised learning, according to known or unknown labels in the training set. Machine learning research focuses on vectors, which are also called features or attributions of samples. Most vectors are numeric, but sequences need to be classified or clustered. Although there are some tools that can transform DNA/RNA/protein sequences to numeric vectors [1–4], clustering sequences directly is still the preferred option. In this paper, we review bioinformatics molecular sequence-clustering algorithms and their applications.

Clustering is not a new topic in bioinformatics. In the analysis of gene expression data, genes obtained from microarray data are clustered and genes in the same cluster are considered to trigger the same function. Several advanced microarray clustering algorithms have been proposed [5], including hierarchical clustering [6] and ensemble clustering [7]. To avoid the noise in microarray data, bi-clustering [8] has been employed where feature selection and sample selection are performed at the same time. Single-cell sequencing data work similar to gene expression data. Different gene expressions were clustered according to different cell types [9, 10]. However, these methods focus on gene expression values rather than gene sequences.

Quan Zou is a professor at the Tianjin University and University of Electronic Science and Technology of China. He is a senior member of Institute of Electrical and Electronics Engineers (IEEE) and Association for Computing Machinery (ACM). He majors in bioinformatics, machine learning and algorithms. His email is zouquan@nclab.net.
Gang Lin is an undergraduate student at the Tianjin University. His research interest is sequence-clustering algorithm.
Xingpeng Jiang is a professor at the Central China Normal University. His research interests are microbiomics and machine learning.
Xiangrong Liu is a professor at the Xiamen University. His research interests are bioinformatics and DNA computing. His email is xrliu@xmu.edu.cn.
Xiangxiang Zeng is an associated professor at the Xiamen University. His research interests are system biology, molecular computing and evolutionary computing. His email is xzeng@xmu.edu.cn.
Submitted: 3 July 2018; **Received (in revised form):** 18 August 2018

The development of next-generation sequencing (NGS) has produced growing amounts of sequence data and a growing need for sequence-clustering algorithms that can process large-scale sequencing data. Two problems that require solving are particularly appealing, namely, minimizing redundancy in machine learning preprocessing and operational taxonomic unit (OTU) clustering in microbiomics research.

Several machine learning methods have been used to identify required protein or DNA/RNA sequences in large datasets. Positive training datasets may contain homologous sequences, for example, similar gene sequences from different species. Similar sequences can be used to improve the performance in the cross-validation step. The CD-HIT (UCSC, San Diego, USA) program [11, 12] has been widely used to remove sequence redundancy and improve sequence analyses, including identifying crotonylation sites [13], predicting ion channels and their types [14], organelle localization of noncoding RNAs [15, 16], conotoxin classification [17] and enzyme family prediction [18, 19].

Recently, microbiomics researchers have contributed to an explosion in the volumes of sequencing data that are produced [20, 21], and a lot of software has been developed to process the data produced by NGS platforms [22]. To analyze species diversity in a complex microbial community [23], sequence processing tools such as CD-HIT [11, 12], USEARCH (Robert C. Edgar) [24] and VSEARCH (Torbjørn Rogne, Oslo, Norway) [25] have been used to cluster sequences into OTUs and to remove redundancy. In the microbiomics researches, sequencing reads are required to be categorized into different microorganisms. According to different microorganism appearance and abundance, sequencing samples could be distinguished. Researchers could pay attention to the relationship between sample phenotypes and microorganism differences. Due to lack of genome information of microorganism, sequencing reads were always clustered into different OTUs. OTU clustering was frequently employed in gut, oil and sea environment researches.

However, different sequence-clustering software tools often produced quite different results. Besides parameters tuning, the algorithm mechanisms are the key points for the differences and none of the tools notably outperform all the others. For inexperienced users, it can be difficult to choose the best tools with the proper parameters, so a lot of time and resources can be wasted in analyzing the results. In this paper, we provide guidelines for clustering biological sequences in different cases.

Sequence redundancy removal

Problems

Sequence redundancy removal is an essential preprocess in several bioinformatics tasks, including machine learning, cross-validation and metagenomics analyses. Redundant sequences not only use up computational resources, but also influence the prediction precision. In machine learning tasks, the precision of the cross-validation process can be improved by using similar training and testing datasets. However, when there are a lot of redundant sequences, some will be in the training dataset and some will be in the testing dataset, and those in the testing dataset will be predicted easily. Redundant sequences produce better performance with cross-validation but poor performance with independent testing datasets, indicating that the robustness and generalization of the predictions are weak. Therefore, it is essential to remove redundant sequences in training and testing datasets.

The redundancy-removal process is shown in Figure 1. Firstly, training and testing sequences were cleaned. It means that sequence redundancy removal was carried out. Then DNA/protein sequences were transferred to vectors by numeric feature extraction. Sometimes feature vectors should be shortened by dimensionality reduction process. Lastly, classifiers (such as support vector machine [26], random forest [27], ensemble classifiers [28], etc.) were employed to distinguishing the different samples.

High-throughput NGS technologies can propel the expansion of metagenomics research [29]. Generally, the data produced by NGS platform are filtered and clustered, so the datasets can be considered as non-redundant datasets, which saves computational resources in subsequent analyses such as OTU clustering, microbial diversity or multiple sequence alignment [30]. The filtering process removes replicated sequences or sequences with high similarity [31].

CD-HIT and USEARCH are ultrafast tools based in greedy heuristic algorithms for processing DNA and protein sequences [11]. Both have options to set crucial parameters, including identity thresholds, threads and memory limitation. A major advantage of these methods is their speed in processing datasets. USEARCH and CD-HIT performed similarly as the other tools tested with different identity thresholds, but their results were different. Redundancy removal methods are different from OTU clustering. The selection of sequences from a rough dataset that are different at a specified identity threshold and the ability to pick rational OTUs are weighed differently in biological problems [32]. Although redundancy removal and OTU clustering methods both use similar greedy algorithm, understanding the detailed mechanisms and selecting proper parameters in different cases is important.

Software tools

In this section, we introduce and compare software tools that can be used to deal with redundant sequences efficiently and accurately.

CD-HIT

CD-HIT is an open source software that was developed based on a greedy incremental algorithm [11, 12] that can separately process nucleotide and peptide sequences. CD-HIT first sorts the input sequences into descending order according to length. The first sequence is commonly considered as trustworthy and representative [30, 33] sequence against which the remaining sequences are compared. Then, CD-HIT calculates an identity value as

$$Identity = (number\ of\ alignment\ columns\ containing\ matched\ residues)/(length\ of\ shorter\ sequence).$$

The identity value for two sequences is compared with the identity threshold. If the identity is higher than the threshold, the sequence is removed as redundancy; otherwise it is added to dataset as a new representative sequence.

Many machine learning-related works employed CD-HIT [34, 35]. But the parameters are set differently. This is partly due to the imbalanced datasets. Negative dataset was always much larger than positive dataset in bioinformatics researches [36–38]. So strict parameters were employed in the negative data while relaxed ones were used in the positive data. Training data would come to balance via tuning the parameters of CD-HIT.

USEARCH

USEARCH is a closed-source software tool based on a greedy incremental algorithm called UCLUST (Robert C. Edgar) algorithm [39]. The 32-bit version of USEARCH is freely available [39].

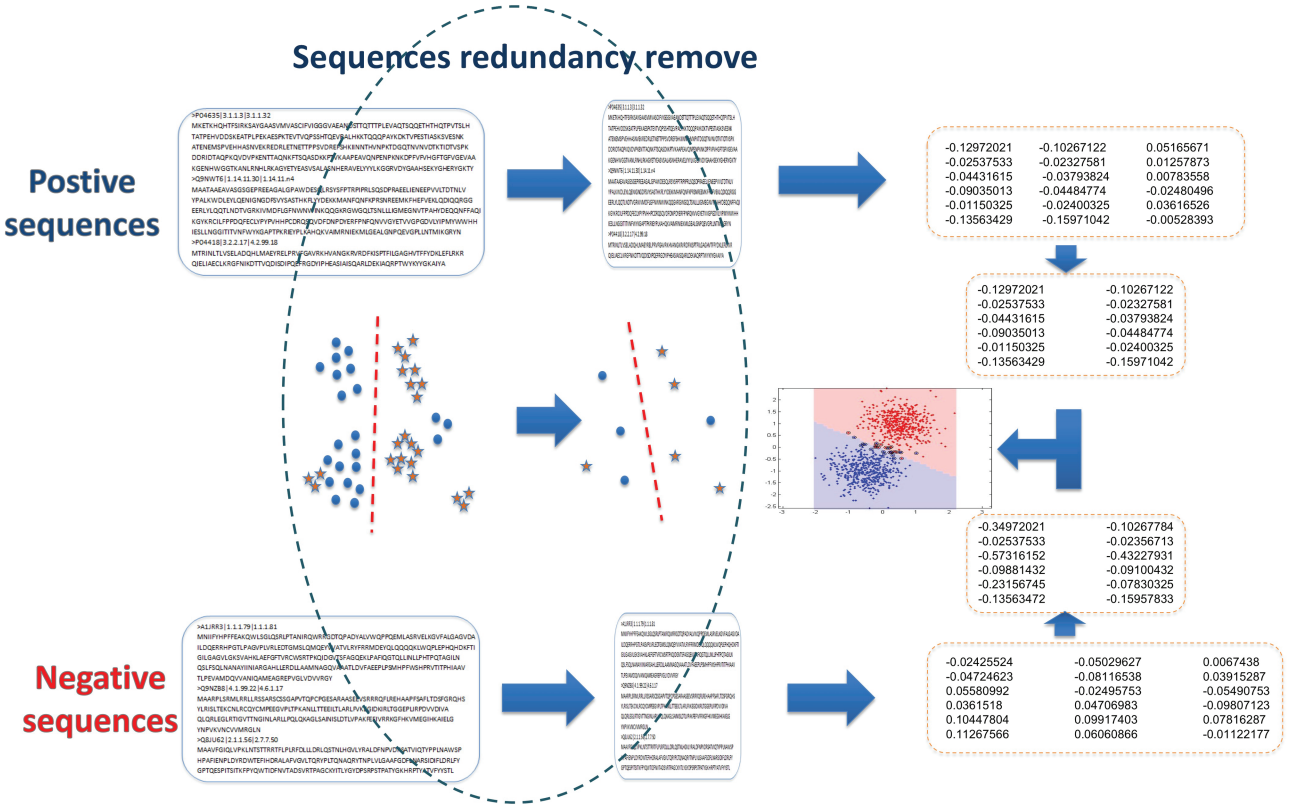

Figure 1. Sequence redundancy-removal process for machine learning tasks.

The options in USEARCH are similar to those in CD-HIT. However, in the latest version of USEARCH, the definition of identity is different from that in CD-HIT and the same as that in BLAST (Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ, USA) [40].

$$Identity = \frac{(number\ of\ alignment\ columns\ containing\ matched\ residues)}{(length\ of\ columns\ after\ alignment)}$$

After alignment, gaps in sequences are represented by '-' and similarity is calculated using the above formula. According to results shown previously on Edgar's website (https://drive5.com/usearch/cdhit_usearch.html), USEARCH v5 (Robert C. Edgar) seemed to be more efficient than CD-HIT v4 (UCSC, San Diego, USA) [40].

VSEARCH

VSEARCH, a free 64-bit and open-source software comparing to USEARCH, was developed by Rognes and others [25]. However, VSEARCH does not support amino acid sequences analysis. VSEARCH uses most of the USEARCH commands for nucleotide sequences analysis, and the clustering method supports pre-sorting by abundance and length. With the limited documentation and websites offered by USEARCH, VSEARCH is an alternative that implements most of the functions of USEARCH using a full dynamic programming method instead of the heuristic greedy algorithm used by USEARCH. VSEARCH uses a different strategy to find centroids with a certain identity threshold. Hence, it is meaningful to compare the performance of VSEARCH with USEARCH for redundancy removal.

Datasets

To compare the performance of the sequence redundancy removal software, we prepared two datasets, namely, a protein dataset from the Swiss-Prot [41] database and a DNA dataset from the Greengenes [42] database. In previous tests, comparing USEARCH to CD-HIT, the benchmarks shown in the website (https://www.drive5.com/usearch/features_benchmarks.html) of Edgar indicated the coverage of identity threshold was from 0.7 to 0.97. The performance of these tools varied at different identity thresholds. For the protein dataset, we set the coverage of identity threshold from 0.5 to 1.0, as shown in Table 1. Because UCLUST and CD-HIT process protein and nucleotide [43] sequences differently, we set the threshold parameter ranges differently.

Comparisons

We used the nucleotide and protein datasets to reflect the features of the software tools in processing the different sequences. In USEARCH, the ability to process nucleotide and protein sequences has been integrated in one algorithm. In contrast, CD-HIT contains a number of tools and different commands are used to cluster either nucleotide or protein sequences. For example, the CD-HIT-EST command is used to process nucleotide sequences. The numbers of clusters obtained and the time cost on the two datasets using CD-HIT, USEARCH and VSEARCH with different identity thresholds are shown in Figure 2.

Table 1. Performance of USEARCH (Method: cluster_fast) and CD-HIT with the Swiss-Prot (260 MB) database. Coverage of identity thresholds was from 50–100%. CD-HIT (Version 4.6) is 64-bit without memory limitation. USEARCH (Version 10.0.240) is 32-bit with 4 GB memory limitation. Programs were run on a server with 56 cores CPU, 504 GB memory and Ubuntu 16.04, and set with 10 threads

Method	Threshold	Clusters	Time	Max memory
USEARCH	100%	467.0 k	6 min 37 s	1.8 GB
USEARCH	90%	327.9 k	1 min 35 s	1.4 GB
USEARCH	80%	266.2 k	1 min 40 s	1.3 GB
USEARCH	70%	214.8 k	2 min 10 s	1.2 GB
USEARCH	60%	169.6 k	3 min 18 s	1.0 GB
USEARCH	50%	129.6 k	3 min 59 s	951 MB
CD-HIT	100%	466.0 k	55 s	746 MB
CD-HIT	90%	323.5 k	63 s	711 MB
CD-HIT	80%	261.2 k	63 s	695 MB
CD-HIT	70%	210.3 k	67 s	687 MB
CD-HIT	60%	164.2 K	6 min	700 MB
CD-HIT	50%	122.6 k	5 h 44 min 36 s	626 MB

Note: cluster_fast is USEARCH command based on UCLUST algorithm.

Table 2. Performance of USEARCH (Method: cluster_fast), CD-HIT-EST and VSEARCH with the Greengenes (1.7 GB) database. Coverage of identity thresholds was from 70–100%. VSEARCH (Version 2.7.1) and CD-HIT (Version 4.6) are 64-bit without memory limitation. USEARCH (Version 10.0.240) is 32-bit. Programs were run on a server with 56 cores CPU, 504 GB memory and Ubuntu 16.04, and set with 10 threads.

Method	Threshold	Clusters	Time	Max memory
USEARCH	100%	fail	fail	fail
USEARCH	90%	18.3 K	7 min 53 s	2.2 GB
USEARCH	80%	1718	21 min 29 s	2.2 GB
USEARCH	70%	231	9 min 50 s	2.2 GB
USEARCH	60%	39	12 min 19 s	2.2 GB
USEARCH	50%	7	10 min 37 s	2.2 GB
CD-HIT-est	100%	fail	fail	2.2 GB
CD-HIT-est	90%	16.1 K	8 h 3 min 50 s	2.3 GB
CD-HIT-est	80%	1747	3 h 31 min 12 s	2.3 GB
VSEARCH	90%	18.1 k	27 min 48 s	5.7 GB
VSEARCH	80%	1915	29 min 18 s	5.6 GB
VSEARCH	70%	248	37 min 38 s	5.7 GB
VSEARCH	60%	37	35 min 19 s	5.7 GB
VSEARCH	50%	6	68 min 15 s	5.7 GB

First, we clustered the Greengenes database, which is about 1.7 GB and contains 1 262 986 sequences, at three different identity levels. Greengenes is a chimera-checked [42] 16S ribosomal RNA (rRNA) gene database without replicated sequences. Because CD-HIT-EST can only process at identity levels ≥0.8, the results were compared with those of USEARCH only at the higher levels.

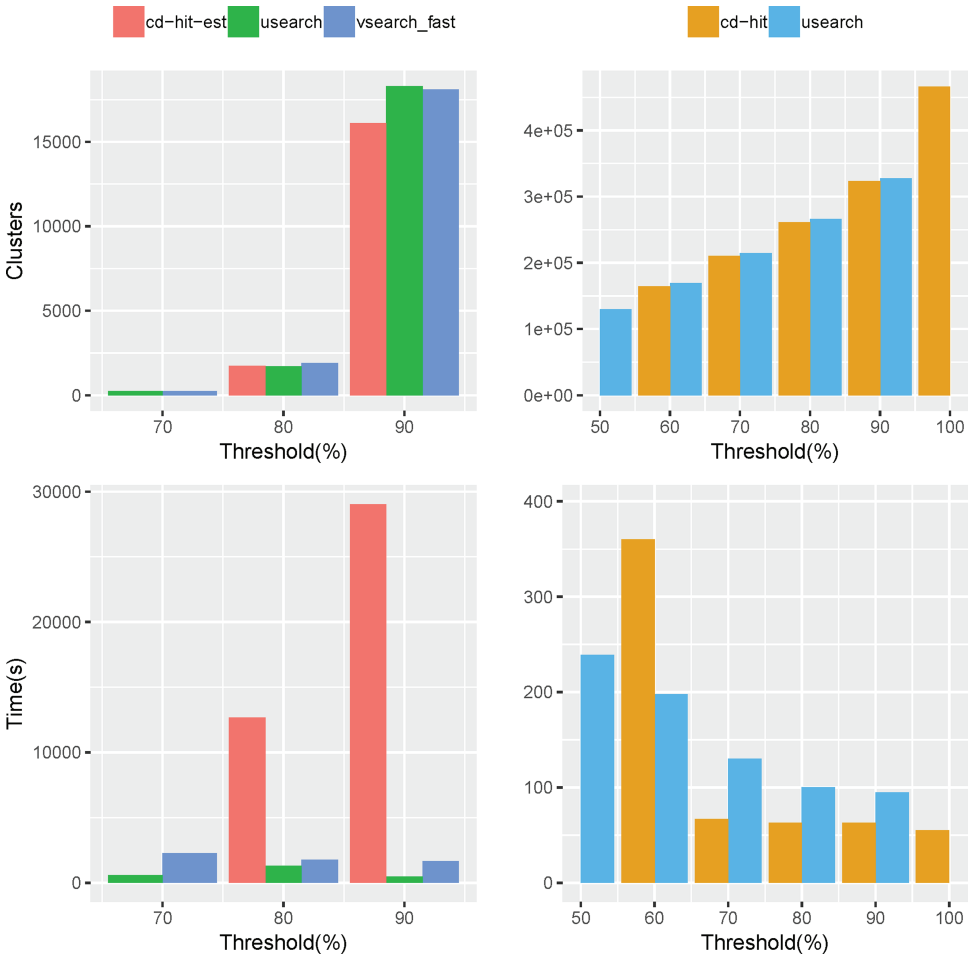

Figure 2. Number of clusters output and time consumed by CD-HIT, USEARCH and VSEARCH on Greengenes (left panels) and Swiss-Prot (right panels) datasets.

The USEARCH 32-bit version halted when processing the Greengenes database at the 100% identity threshold (Table 2). The maximum memory space taken by USEARCH and CD-HIT was similar at the same identity levels. However, the efficiency of USEARCH was at least 10 times better than CD-HIT-EST. Therefore, USEARCH was faster than CD-HIT in processing nucleotide sequences. Interestingly, when USEARCH finished clustering at the 70% identity level, only 231 sequences were obtained, which, compared with the total amount of input, is too small. Thus, the CD-HIT-EST limitation of identity threshold $\geq$80% could be considered meaningful. VSEARCH processed the sequences slower than USEARCH, but generally performed better than CD-HIT-EST. It cost about an average of half an hour for identity thresholds above 50%. Although VSEARCH does not have a 64-bit memory limitation, it uses more than double the memory needed by USEARCH for the same computation task. The number of sequences output by VSEARCH was close to the number output by USEARCH.

Next, we clustered the Swiss-Prot database, which is a protein sequence and knowledge database that is valued for its high-quality annotation, the use of standardized nomenclature, direct links to specialized databases and minimal redundancy. VSEARCH does not support amino acid sequences processing yet, so could not be used to cluster the Swiss-Prot dataset. We found that CD-HIT was clearly more efficient than USEARCH in clustering the protein sequences from 70 to 100% identity levels, and the memory usage of CD-HIT was less than that of USEARCH, as shown in Table 2. The cost of time by CD-HIT showed exponential growth at the 50% identity level.

It has been claimed in the USEARCH website that UCLUST is effective at the level of 50% identity and above for proteins and 75% and above for nucleotides (https://drive5.com/usearch/manual/uclust_algo.html). The alignment quality will be poor at low-identity thresholds, so the UCLUST algorithm cannot determine homology because of untrustworthy alignments. The lowest identity thresholds that can be used by CD-HIT and CD-HIT-EST are 40 and 80%, respectively. In addition, Li *et al*. [9] developed another tool, PSI-CD-HIT, that has been included in CD-HIT. The PSI-CD-HIT (UCSC, San Diego, USA) tool is a Perl script based on BLAST, which calculates similarities for processing protein sequences at much lower thresholds, such as 25%. These two ultrafast [11] softwares, USEARCH's UCLUST algorithm and CD-HIT, can effectively remove redundancy from nucleotide and protein sequences, but the results are a little different of the amount of the outputs. This is because the two algorithms are different, although both are based on greedy incremental algorithms, and use different ways to calculate similarity [31]. In brief, we recommend the latest version of CD-HIT instead of UCLUST according to memory and time cost. But users should notice that the old version of CD-HIT performed poorly. Latest version and proper subpackages (e.g. PSI-CD-HIT for protein sequences) ought to be selected.

OTU clustering

Microbiomes obtained from organisms or from the environment such as soil, sludge and water can now be studied because of the advancements of NGS technologies. 16S rRNA sequences are used widely to determine phylogenetic relationships between microorganisms. By clustering 16S rRNA sequences [44] into OTUs, microbiomes can be profiled for bacterial diversity, composition, richness and community structure. The hypervariable [32] region in 16S rRNA offers species-specific signatures to identify taxonomy [45]. Conventionally, if the dissimilarity of two 16S sequences was under a 3 or 5% threshold, both sequences were considered to belong to organisms in the same genus. This identity threshold was proposed in 1994 when few 16S rRNA sequences were available and the validity of using such threshold has not been proved. Although it is not an accurate boundary, most OTU-clustering methods use an identity threshold of 97% [46]. In a recent study, Edgar proposed that the optimal identity threshold should be 100% for the v4 hypervariable region and 99% for full-length sequences [38]. He suggested a zero-radius OTU for its ability to better discriminate species compared with traditional OTUs [29, 47]. In order to compare the different OTU-clustering software tools, we selected benchmark data and listed the different performance in this section.

Benchmark datasets

We used the full dataset from Schloss's laboratory (http://www.mothur.org/MiSeqDevelopmentData/StabilityNoMetaG.tar) for OTU picking with several popular tools. This huge dataset contain 362 pairs of fastq files. Using MOTHUR (Patrick Schloss, Michigan) [48] MiSeq (Illumina, San Diego, California) [20] SOP (Illumina, San Diego, California), 21 pairs of fastq files were extracted from this dataset. First, all pair-end files were merged into one fastq file by USEARCH. The mean merged length was about 253 bp and the maximum and average expected errors (EEs) were 5.4 and 0.1, which indicated this dataset of 16S rRNA sequences was of high quality. Second, the merged file was filtered by USEARCH to remove sequences with EE values $>$1. Third, we removed all the replicated sequences. After preprocessing, the remaining (about 137.6 K) sequences were clustered into OTUs.

The preprocessing and OTU-picking procedures of these software tools including USEARCH, CD-HIT, SWARM, MOTHUR and SUMACLUST are shown in Figure 3. Pair-end sequencing data were firstly merged to Fastq file. Then, we employed Usearch to filter the trimming reads, the short reads and the repeated reads. Cleaned fasta file would be created for different OTU-clustering tools, including SUMACLUST, MOTHER, CD-HIT, SWARM and USEARCH. For MOTHER, there would be four steps, including Unique, Align, Matrix and Clust (Patrick Schloss, Michigan). USEARCH also contains three sub-packages: UPARSE (Robert C. Edgar), UCLUST and UNOISE3 (Robert C. Edgar). We will introduce them in detail in the next section.

Software tools

Several popular de novo OTU-picking methods are discussed in [24]. But in this paper, we employed different benchmark data and focused on different views including memory and time costs algorithms mechanism. De novo clustering methods are preferred for their independence of references for clustering OTUs. Before OTU clustering, the original data from NGS platforms such as Illumina (Illumina, San Diego, California) and Roche 454 (454 Life Sciences, Branford, Connecticut) [31] need to be preprocessed. Preprocessing pipelines include combination (pair-end reads only), filtering [46] and deduplication so that only clean 16S rRNA sequences are clustered into OTUs. The popular de novo OTU-picking methods are based on three main algorithm types, namely, distance matrix-based algorithms, greedy incremental algorithms and single-linkage (SL) algorithms.

USEARCH and CD-HIT developed two similar tools based on greedy incremental algorithms to cluster 16S rRNA sequences. In the de novo OTU-picking method, USEARCH offers two main

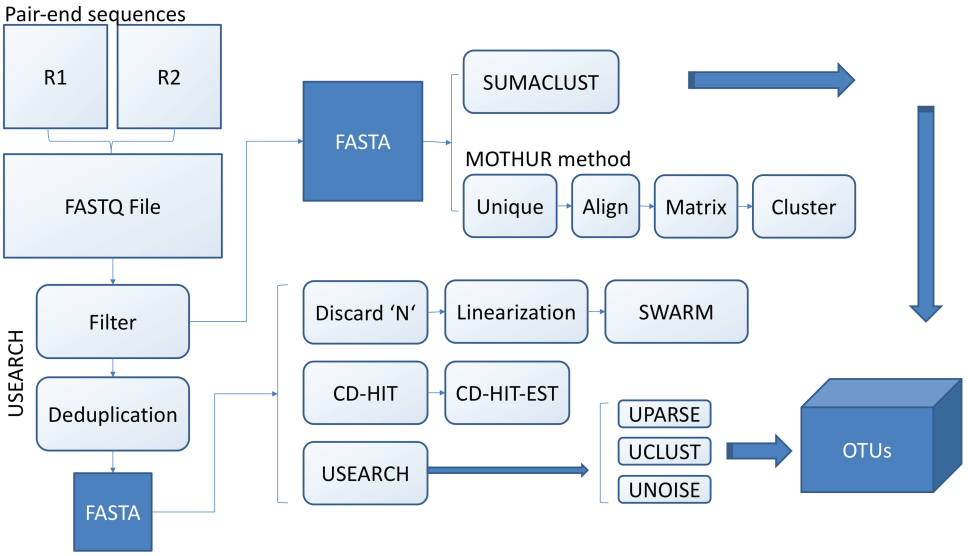

Figure 3. Preprocessing and OTU clustering procedures used to test the performance of the described software.

clustering methods: UCLUST and UPARSE. The order of input sequences is important when using UCLUST because it chooses the first input sequence as a centroid against which all other input sequences are compared. If the sequence similarity is over a 97% identity threshold, the sequence is classified into the centroid; otherwise UCLUST adds this sequence as a new centroid. Each centroid is considered as an OTU. Although before clustering, UCLUST and CD-HIT can sort by the length of sequences to optimize the quality of centroids, this is not recommended. UCLUST and CD-HIT are distance-based greedy clustering methods, so sequences close to one centroid are usually considered to belong to the same species. UPARSE [49] is an abundance-based algorithm that was developed by Edger especially for OTU picking. The differences between UCLUST and UPARSE are shown in Figure 4. The greedy target is different. UCLUST aimed to categorize into the longest sequence clustering while UPARSE tried to rush into the most abundant clustering. After deduplication by USEARCH, sequences are annotated with a size number representing their abundance in the community. Then, UPARSE clusters OTUs according to the size annotation at a 97% identity threshold. Edgar considers it rational to choose the most abundant sequences as centroids and chimeras are filtered by this method. Single sequences and chimeras may cause errors in OTU clustering. SUMACLUST is another greedy OTU clustering method similar to CD-HIT and USEARCH. It claims to be able to detect wrong sequences caused by PCR amplicons from fasta files. SUMACLUST is an open-source software tool that implements multi-thread processing with OpenMP API to guarantee computational efficiency. By default, it uses an abundance-based greedy clustering method that chooses abundant sequences as preferred OTU centers.

MOTHUR is an open source and comprehensive software package that can analyze community sequence data [50]. MOTHUR is based on previous tools but is more powerful than previous tools. The MOTHUR clustering method is totally different from the methods described above, although the preprocessing procedures are similar. After trimming, filtering and deduplication, all the sequences are aligned to the same length to calculate a distance matrix. Unlike the other method where calculating the similarity between sequences after alignment and clustering is integrated in one method, MOTHUR calculates alignment and cluster similarities separately. To obtain a distance matrix, all sequence distances are calculated and saved as a matrix. Then, the cluster command assigns sequences to OTUs by reading the matrix with several options, namely, nearest neighbor, furthest neighbor and average neighbor, which reflect the degree of clustering. Nearest neighbor, also called SL, means each sequence in one OTU is, at most, 97% distant (default OTU threshold or bigger threshold) from the most similar sequence in the OTU. SL is a loose clustering strategy. Furthest neighbor, also called complete linkage (CL), is the most stringent choice that means all sequences in one OTU are at most 97% distant from all other sequences in the OTU. Figure 5 shows the differences between SL and CL. Average neighbor, also called average linkage, is the middle ground between SL and CL. The default clustering method used by MOTHUR is OptiClust (Patrick Schloss, Michigan), which uses metrics to assemble OTUs. The metrics include Matthew's correlation coefficient, true positives and true negatives. It also determines the quality of clustering.

SWARM [51] used a novel SL clustering algorithm without an identity threshold. It also does not rely on the order of input sequences based either on length or abundance. The SL clustering method looks like a tree-growing procedure. During the growth phase, SWARM calculates differences in pairwise aligned sequences to profile OTUs. It uses k-mer comparisons and a novel exact global pairwise alignment algorithm. All the input files of amplicon sequences are put into a pool. Then, an amplicon is chosen as an OTU seed and the D-values, which represent the number of nucleotide mismatches, are calculated with all the remaining sequences. By comparing the differences, SWARM chooses a smaller d-value amplicon as a subseed. Although the SWARM clustering method does not rely on the input order of sequences and uses a local d-neighbor (d nucleotide differences), it still uses amplicon abundance information and the internal structure of OTUs to refine the clustering results. This SL clustering process generates stable OTUs, regardless of the first seed choice.

The URLs for these software programs and their features are shown in Table 4.

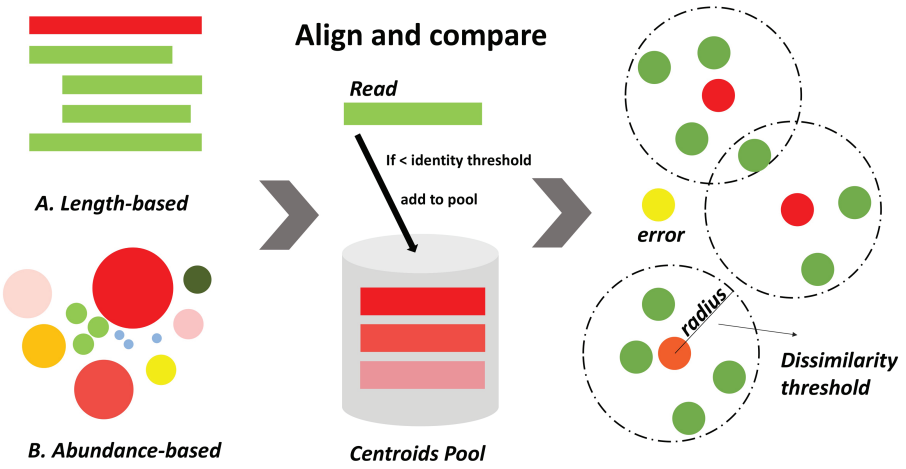

Figure 4. Distance-based greedy clustering **(A)** and abundance-based greedy clustering **(B)** methods in USEARCH and UPARSE, respectively.

Table 3. Performance of popular OTU-picking methods with dataset. After preprocessing, the dataset contained 133 068 unique sequences. CD-HIT (Version 4.6), USEARCH (Version 10.0.240), MOTHUR (Version 1.39.3)[a], SUMACLUST (Version 1.0.20) and SWARM (Version 2.2.2)[b] were used for OTU picking and set with 10 threads on a server with 56 cores CPU, 504 GB memory and Ubuntu 16.04

Chimera detecting	Method	OTUs	Time(s) Deduplication	Clustering	Max memory usage Deduplication (by USEARCH)	Clustering
No	USEARCH	8366	8 s	7 s	1.3 GB	161 MB
	USEARCH (sort)	8534	8 s	7 s	1.3 GB	162 MB
	CD-HIT-est	6057	8 s	5 s	1.3 GB	232 MB
Yes	unoise	507(1153 chimeras)	8 s	5 s	1.3 GB	74 MB
Yes	cluster_otu	303(1753 chimeras)	8 s	5 s	1.3 GB	49 MB
No	sumaclust-L	5113	8 s	6 s	1.3 GB	302 MB
	sumaclust	6103	8 s	7 s	1.3 GB	310 MB

[a] MOTHUR took too much time to generate the matrix and choose OTUs.
[b] SWARM failed because of an abundance annotations bug.

Table 4. Algorithms, order dependence and speed of OTU clustering methods

Software	Algorithm	Order dependence	Speed	URL
USEARCH	greedy heuristic	abundance, length	fast	https://www.drive5.com/usearch/
CD-HIT	greedy heuristic	length	fast	http://weizhongli-lab.org/CD-HIT/
MOTHUR	SL, AL, CL	none	slow	https://www.mothur.org/
VSEARCH	greedy heuristic	abundance, length	fast	https://github.com/torognes/VSEARCH
SUMACLUST	greedy heuristic	abundance, length	fast	https://omictools.com/sumaclust-tool
SWARM	SL	abundance	medium	https://github.com/torognes/swarm

Comparison

We tested with several popular OTU-picking methods, as shown in Table 3. At the same identity thresholds, the greedy incremental-based algorithms CD-HIT-EST, UCLUST and SUMACLUST performed similarly [52], and the sum of total OTUs produced by them were close at the same magnitude (from 6 K to 8 K OTUs). These and other similar greedy incremental-based clustering methods have an obvious shortcoming: the order of input sequences influences the final OTUs that are produced. UCLUST offers two clustering options: to use the original order or to sort by the length. In CD-HIT-EST, sorting by length before clustering is a stable procedure. No matter the order of input sequences, OTUs lose a lot of information of the 'true' OTUs because sequences in the middle or ends of datasets are compared only to the existing centroids in the OTU database arbitrarily without chimera detecting. Therefore, generating OTUs by UCLUST and CD-HIT is not recommended and the produced OTUs are not convincing. We suppose to use UPARSE and UNOISE3 algorithms to generate high-quality OTUs. The dedicated methods for OTU picking offered by USEARCH, UPARSE and UNOISE3 produce smaller outputs. UPARSE and UNOISE3 are abundance-based greedy clustering methods with chimera detecting that chose only hundreds of OTUs from the

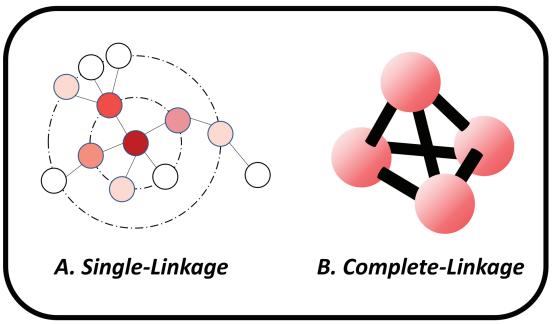

Figure 5. SL and complete-linkage options used in the MOTHUR clustering method.

total of 133 K duplicated sequences. These two methods use 97 and 100% as immutable OTU-picking identity thresholds. All these greedy incremental-based clustering methods have very short times and use small memory space.

MOTHUR [23] is a very bulky method to cluster OTUs, and has a bird's-eye view to catch OTUs. Distance matrices are important for MOTHUR to cluster OTUs. Matrices can reflect the similarity or distance of each sequence to all other sequences in a data file. Calculating the matrix and cluster from the matrix has a high degree of time complexity. MOTHUR is not suitable to process huge datasets even if the dataset has been duplicated. Other procedures in MOTHUR are also very complicated and redundant.

SWARM uses a novel SL algorithm and alignment method, produces many more OTUs than other methods and costs little time. By default, the d-value taken by SWARM is 1. However, the input file needs strict preprocessing. SWARM considers the abundance information of a sequence but is error sensitive.

Conclusion

The developing sequencing technologies have advanced research in metagenomics by making sequencing less expensive and more available. These advances have also led to the development of bioinformatics, and a wide variety of metagenomics-related software has been developed. Different OTU-picking methods are now available to analyze 16S rRNA sequences. The most popular methods take advantage of the abundance of information, although the implementation of the software varies. The need to preprocess different datasets differently to meet the different input standards of these software programs still presents some problems. Further, their outputs are also organized differently and are not always user friendly. Unsupervised machine learning methods based on the Naïve Bayes algorithm [53] have been proposed. SL and distance matrices have also been used widely. Some metrics used to measure the quality of constructed OTUs, such as Matthew's correlation coefficient, true positives and true negatives, show unstable effects on different datasets. Parallel OTU-clustering software tools were also developed for big data, including ESPRIT-Forest [54]. ESPRIT-Forest employed MPI and OpenMP as parallel mechanism. Recently, Hadoop and Spark were developed due to the friendly coding style, and they were applied in several bioinformatics problems [55], such as multiple sequence alignment [56, 57] and evolutionary tree reconstruction [58–60]. We can expect that Hadoop and Spark will be employed in the sequence clustering in the future. No matter which method performs well and picks good OTUs, it is better to select a method with complete tools that can analyze [61]. Software that can identify the comprehensive information and remove errors in 16S rRNA datasets may be the most useful for deeper research. Also, the development of long-read [62] sequencing technologies may help to improve the accuracy of community information of microbial species.

Key Points

- The paper analyzed and compared the sequence-clustering tools, and provided helpful suggestions.
- Algorithm mechanisms were explained clearly, which could help the readers to select proper tools for 16 s rRNA sequences.
- Besides meta-genomics analysis, sequence redundancy removal in the machine learning preprocess is another highlight of our paper. We also elaborated on this problem and provided useful instructions and guidelines on how to better address it.

Acknowledgement

We thank Margaret Biswas, PhD, from Edanz Group (www.edanzediting.com/ac) for editing the draft of this manuscript.

Funding

The work was supported by the National Key R&D Program of China (SQ2018YFC090002), (No. 61771331, No.61872309) and the funding from Shandong Provincial Key Laboratory of Biophysics.

References

1. Liu B, Wu H, Chou KC. Pse-in-One 2.0: an improved package of web servers for generating various modes of pseudo components of DNA, RNA, and protein sequences. *Nat Sci* 2017;**09**(4):67–91.
2. Li YH, Xu JY, Tao L, Li XF, Li S, Zeng X, Chen SY, Zhang P, Qin C, Zhang C, Chen Z. SVM-Prot 2016: a web-server for machine learning prediction of protein functional families from sequence irrespective of similarity. *PloS One* 2016;**11**(8):e0155290.
3. Liu B, Liu F, Wang X, Chen J, Fang L, Chou KC. Pse-in-One: a web server for generating various modes of pseudo components of DNA, RNA, and protein sequences. *Nucleic Acids Res* 2015;**43**(Web Server issue): W65–71.
4. Chen W, Zhang X, Brooker J, Lin H, Zhang L, Chou KC. PseKNC-General: a cross-platform package for generating various modes of pseudo nucleotide compositions. *Bioinformatics* 2014;**31**(1):119–20.
5. Boutros P, Okey A. Unsupervised pattern recognition: an introduction to the whys and wherefores of clustering microarray data. *Brief Bioinform* 2005;**6**(4):331–43.
6. Lafond-Lapalme J, Duceppe MO, Wang S, Moffett P, Mimee B. A new method for decontamination of de novo transcriptomes using a hierarchical clustering algorithm. *Bioinformatics* 2016;**33**(9):1293–300.

7. Yu Z, Chen H, You J, Wong HS, Liu J, Li L, Han G. Double selection based semi-supervised clustering ensemble for tumor clustering from gene expression profiles. *IEEE/ACM Trans Comput Bioinform* 2014;**11**(4):727–40.
8. Zhang Y, Xie J, Yang J, Fennell A, Zhang C, Ma Q. QUBIC: a bioconductor package for qualitative biclustering analysis of gene co-expression data. *Bioinformatics* 2017;**33**(3):450–2.
9. Aibar S, González-Blas CB, Moerman T, Imrichova H, Hulselmans G, Rambow F, Marine JC, Geurts P, Aerts J, van den Oord J, Atak ZK. SCENIC: single-cell regulatory network inference and clustering. *Nat Methods* 2017;**14**(11):1083–6.
10. Li X, Chen W, Chen Y, Zhang X, Gu J, Zhang MQ. Network embedding-based representation learning for single cell RNA-seq data. *Nucleic Acids Res* 2017;**45**(19):e166.
11. Li W, Fu L, Niu B, Wu S, Wooley J. Ultrafast clustering algorithms for metagenomic sequence analysis. *Brief Bioinform* 2012;**13**(6):656–68.
12. Li W, Godzik A. Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences. *Bioinformatics* 2006;**22**(13):1658.
13. Qiu WR, Sun BQ, Tang H, Huang J, Lin H. Identify and analysis crotonylation sites in histone by using support vector machines. *Artif Intell Med* 2017;**83**:75–81.
14. Zhao YW, Su ZD, Yang W, Lin H, Chen W, Tang H. IonchanPred 2.0: a tool to predict ion channels and their types. *Int J Mol Sci* 2017;**18**(9):1838.
15. Feng P, Zhang J, Tang H, Chen W, Lin H. Predicting the organelle location of noncoding RNAs using pseudo nucleotide compositions. *Interdiscip Sci* 2016; **9**(4):1–5.
16. Su ZD, Huang Y, Zhang ZY, Zhao YW, Wang D, Chen W, Chou KC, Lin H, Hancock J. iLoc-lncRNA: predict the subcellular location of lncRNAs by incorporating octamer composition into general PseKNC. *Bioinformatics* 2018.
17. Dao FY, Yang H, Su ZD, Yang W, Wu Y, Hui D, Chen W, Tang H, Lim H. Recent advances in conotoxin classification by using machine learning methods. *Molecules* 2017; **22**(7):1057.
18. Zou Q, Chen W, Huang Y, Liu X, Jiang Y. Identifying multifunctional enzyme with hierarchical multi-label classifier. *J Comput Theor Nanosci* 2013;**10**(4):1038–43.
19. Wu Y, Tang H, Chen W, Lin H. Predicting human enzyme family classes by using pseudo amino acid composition. *Curr Proteomics* 2016;**13**(2):99–104.
20. Franzén O, Hu J, Bao X, Itzkowitz SH, Peter I, Bashir A. Improved OTU-picking using long-read 16S rRNA gene amplicon sequencing and generic hierarchical clustering. *Microbiome* 2015;**3**:43.
21. Gilbert JA, Jansson JK, Knight R. The Earth Microbiome project: successes and aspirations. *BMC Biol* 2014;**12**:69.
22. Luo C, Tsementzi D, Kyrpides N, Read T, Konstantinidis KT. Direct comparisons of Illumina vs. Roche 454 sequencing technologies on the same microbial community DNA sample. *PLoS One* 2012;**7**(2):e30087.
23. Schloss PD, Westcott SL, Ryabin T, Hall JR, Hartmann M, Hollister EB, Lesniewski RA, Oakley BB, Parks DH, Robinson CJ, Sahl JW. Introducing mothur: open-source, platform-independent, community-supported software for describing and comparing microbial communities. *Appl Environ Microbiol* 2009;**75**(23):7537–41.
24. Westcott SL, Schloss PD. De novo clustering methods outperform reference-based methods for assigning 16S rRNA gene sequences to operational taxonomic units. *PeerJ* 2015; **3**:e1487.
25. Rognes T, Flouri T, Nichols B, Quince C, Mahé F. VSEARCH: a versatile open source tool for metagenomics. *PeerJ*. 2016 Oct 18;**4**:e2584.
26. Li D, Ju Y, Zou Q. Protein folds prediction with hierarchical structured SVM. *Curr Proteomics* 2016;**13**(2):79–85.
27. Zhao X, Zou Q, Liu B, Liu X. Exploratory predicting protein folding model with random forest and hybrid features. *Curr Proteomics* 2014;**11**(4):289–99.
28. Lin C, Chen W, Qui C, Wu Y, Krishnan S, Zou Q. LibD3C: ensemble classifiers with a clustering and dynamic selection strategy. *Neurocomputing* 2014;**123**:424–35.
29. Zepeda Mendoza ML, Sicheritz-Pontén T, Gilbert MTP. Environmental genes and genomes: understanding the differences and challenges in the approaches and software for their analyses. *Brief Bioinform* 2015;**16**(5):745–58.
30. Sun Y, Cai Y, Huse SM, Knight R, Farmerie WG, Wang X, Mai V. A large-scale benchmark study of existing algorithms for taxonomy-independent microbial community analysis. *Brief Bioinform* 2012;**13**(1):107.
31. Dröge J, Mchardy AC. Taxonomic binning of metagenome samples generated by next-generation sequencing technologies. *Brief Bioinform* 2012;**13**(6):646.
32. Wang X, Yao J, Sun Y, Mai V. M-pick, a modularity-based method for OTU picking of 16S rRNA sequences. *BMC Bioinformatics* 2013;**14**(1):43.
33. Fu L, Niu B, Zhu Z, Wu S, Li W. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics* 2012;**28**(23):3150.
34. Chen W, Yang H, Feng P, Ding H, Lin H. iDNA4mC: identifying DNA N4-methylcytosine sites based on nucleotide chemical properties. *Bioinformatics* 2017;**33**(22):3518–23.
35. Feng PM, Chen W, Lin H, Chou KC. iHSP-PseRAAAC: identifying the heat shock protein families using pseudo reduced amino acid alphabet composition. *Anal Biochem* 2013;**442**(1):118–25.
36. Song L, Li D, Zeng X, Wu Y, Guo L, Zuo Q. nDNA-Prot: identification of DNA-binding proteins based on unbalanced classification. *BMC Bioinformatics* 2014;**15**:298.
37. Wan S, Duan Y, Zou Q. HPSLPred: an ensemble multi-label classifier for human protein subcellular location prediction with imbalanced source. *Proteomics* 2017;**17**:1700262.
38. Wang C, Hu L, Guo M, Liu X, Zuo Q. imDC: an ensemble learning method for imbalanced classification with miRNA data. *Genet Mol Res* 2015;**14**(1):123–33.
39. Teeling H, Glöckner FO. Current opportunities and challenges in microbial metagenome analysis—a bioinformatic perspective. *Brief Bioinform* 2012;**13**(6):728–42.
40. Edgar RC. Search and clustering orders of magnitude faster than BLAST. *Bioinformatics* 2010;**26**(19):2460–1.
41. Boeckmann B, Bairoch A, Apweiler R, Blatter MC, Estreicher A, Gasteiger E, Martin MJ, Michuod K, O'donovan C, Phan I, Pilbout S. The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. *Nucleic Acids Res* 2003; **31**(1):365.
42. DeSantis TZ, Hugenholtz P, Larsen N, Rojas M, Brodie EL, Keller K, Huber T, Dalevi D, Hu P, Andersen GL. Greengenes, a chimera-checked 16S rRNA gene database and workbench compatible with ARB. *Appl Environ Microbiol* 2006;**72**(7): 5069–72.
43. Schloss PD. Secondary structure improves OTU assignments of 16S rRNA gene sequences. *ISME J* 2013;**7**(3):457–60.
44. Edgar RC. SEARCH_16S: a new algorithm for identifying 16S ribosomal RNA genes in contigs and chromosomes. bioRxiv 2017 Jan 1:124131.

45. Ngom-Bru C, Barretto C. Gut microbiota: methodological aspects to describe taxonomy and functionality. *Brief Bioinform* 2012;**13**(6):239.
46. Flynn JM, Brown EA, Chain FJ, MacIsaac HJ, Cristescu ME. Toward accurate molecular identification of species in complex environmental samples: testing the performance of sequence filtering and clustering methods. *Ecol Evol* 2015;**5**(11):2252–66.
47. Edgar RC. Updating the 97% identity threshold for 16S ribosomal RNA OTUs. Bioinformatics 2018 Feb 28; 1:5.
48. Bokulich NA, Rideout JR, Mercurio WG, Shiffer A, Wolfe B, Maurice CF, Dutton RJ, Turnbaugh PJ, Knight R, Caporaso JG. mockrobiota: a public resource for microbiome bioinformatics benchmarking. *mSystems* 2016;**1**(5):e00062-16.
49. Edgar RC. UPARSE: highly accurate OTU sequences from microbial amplicon reads. *Nat Methods* 2013;**10**(10):996.
50. Schloss PD, *et al*. Introducing mothur: open-source, platform-independent, community-supported software for describing and comparing microbial communities. *Appl Environ Microbiol* 2009;**75**(23):7537.
51. Mahé F, Rognes T, Quince C, de Vargas C, Dunthorn M. Swarm: robust and fast clustering method for amplicon-based studies. *PeerJ* 2014;**2**:e593.
52. Chen W, Zhang CK, Cheng Y, Zhang S, Zhao H. A comparison of methods for clustering 16S rRNA sequences into OTUs. *Plos One* 2013;**8**(8):e70837.
53. Hao X, Jiang R, Chen T. Clustering 16S rRNA for OTU prediction: a method of unsupervised Bayesian clustering. *Bioinformatics* 2011;**27**(5):611–8.
54. Cai Y, Zheng W, Yao J, Yang Y, Mai V, Mao Q, Sun Y. ESPRIT-Forest: parallel clustering of massive amplicon sequence data in subquadratic time. *PloS Comput Biol* 2017;**13**(4):e1005518.
55. Zou Q, Li XB, Jiang WR, Lin ZY, Li GL, Chen K. Survey of MapReduce frame operation in bioinformatics. *Brief Bioinform* 2014;**15**(4):637–47.
56. Zou Q, Hu Q, Guo M, Wang G. HAlign: fast multiple similar DNA/RNA sequence alignment based on the centre star strategy. *Bioinformatics* 2015;**31**(15):2475–81.
57. Su W, Liao X, Lu Y, Zou Q, Peng S. Multiple sequence alignment based on a suffix tree and center-star strategy: a linear method for multiple nucleotide sequence alignment on spark parallel framework. *J Comput Biol* 2017;**24**(12):1230–42.
58. Wan S, Zou Q. HAlign-II: efficient ultra-large multiple sequence alignment and phylogenetic tree reconstruction with distributed and parallel computing. *Algorithms Mol Biol* 2017;**12**:25.
59. Zou Q, Wan S, Zeng X, Ma ZS. Reconstructing evolutionary trees in parallel for massive sequences. *BMC Syst Biol* 2017;**11**(6):100.
60. Wang J, Guo M. A review of metrics measuring dissimilarity for rooted phylogenetic networks. *Brief Bioinform* 2018. doi:10.1093/bib/bby062.
61. Beaumont M, Goodrich JK, Jackson MA, Yet I, Davenport ER, Vieira-Silva S, Debelius J, Pallister T, Mangino M, Raes J, Knight R. Heritable components of the human fecal microbiome are associated with visceral fat. Genome biology. 2016 Dec;17(1):189.
62. Franzén, O Hu J, Bao X, Itzkowitz SH,Peter I, Bashir A. Erratum to: Improved OTU-picking using long-read 16S rRNA gene amplicon sequencing and generic hierarchical clustering. *Microbiome* 2015;**3**(1):1–1.