WEVOTE: Weighted Voting Taxonomic Identification Method of Microbial Sequences

Ahmed Metwally[1,2,*], Yang Dai[1], Patricia Finn[2], David Perkins[2,3]

1 Bioinformatics Program, Department of Bioengineering, University of Illinois at Chicago, Chicago, IL, USA.
2 Department of Medicine, University of Illinois at Chicago, Chicago, IL, USA.
3 Department of Surgery, University of Illinois at Chicago, Chicago, IL, USA.

*** E-mail: ametwa2@uic.edu**

Abstract

Background: Metagenome shotgun sequencing presents opportunities to identify organisms that may prevent or promote disease. Analysis of sample diversity is achieved by taxonomic identification of metagenomic reads followed by generating an abundance profile. Numerous tools have been developed for taxonomic identification based on different design principles. Tools that have been designed to achieve high precision and practical performance still lack sensitivity. Moreover, tools with the highest sensitivity suffer from low precision, low specificity along with long computation time.

Methods: In this paper, we present WEVOTE (WEighted VOting Taxonomic idEntification), a method that classifies metagenome shotgun sequencing DNA reads based on an ensemble of existing methods using k-mer based, marker-based, and naive-similarity based approaches. Our evaluation, based on fourteen benchmarking datasets, shows that WEVOTE reduces occurrence of the false positives to half of that produced by other high sensitive tools while also maintaining the same level of sensitivity.

Conclusions: WEVOTE is an efficient, automated tool that combines multiple individual taxonomic identification methods. It is expandable and has the potential to reduce false positives and produce a more accurate taxonomic identification for microbiome data. WEVOTE was implemented using C++ and shell script and is available at https://bitbucket.org/ametwally/wevote.

Keywords: Metagenomics; Microbiome; Sequence Classification; Next-Generation Sequencing; Whole Genome Sequencing.

Introduction

The microbiome plays a vital role in a broad range of host-related processes and has a significant effect on host health. Over the past decade, the culture-independent MetaGenome Shotgun (MGS) sequencing has become an emerging tool for studying the diversity and the ecology of microbial communities. One of the key steps in data analysis is the taxonomic classification of sequences assembled from a metagenomic dataset.

The existing taxonomic identification methods of MGS data can be primarily classified into four categories: naive-similarity based methods, methods based on analyzing sequence alignment results, methods that are based on sequence composition, such as k-mers, and marker-based methods. The naive-similarity-based methods rely on mapping each read to a reference database, such as the NCBI nucleotide database, and the taxonomic annotation of the best hit is assigned to the read if it passes a pre-set threshold. Bowtie [1], BLASTN [2], and its faster version MegaBlast [3] are the most commonly used algorithms in this category. Since the number of sequences in the database is enormous, these methods have a high probability of finding a match, but they demand extensive computational time. Therefore, these types of methods usually achieve a higher level of sensitivity compared to other methods [4, 5]. However, one significant drawback is the increased rate of false positive annotations.

The category of analyzing the sequence alignment results includes MEGAN [6], and PhymmBL [4]. This class of methods consists of a preprocessing step and a post-analysis step. In MEGAN, an algorithm involving the lowest common ancestor (LCA) assigns each read an NCBI taxonomic identification number (si. taxon / pl. taxa) that reflects the level of conservation within the sequence. On the other hand, PhymmBL constructs a large number of Interpolated Markov Models (IMMs) using a BLASTN query against a reference database. It would subsequently compute scores which correspond to the probability of the generated IMMs matching a given sequence. Then it classifies the read using the clade labels belonging to the organism whose IMM generated the best score for that read. The methods in this class usually require more computational time than those in the naive-similarity methods.

In contrast, the marker-based methods utilize a curated collection of marker genes where each marker gene set is used to identify a unique group of clades. The fundamental difference between these methods and the naive-similarity methods is in the reference databases. Based on how the database of the marker genes is formed, this type of methods is classified into two main subcategories: (i) methods that depend on a database of a universal single copy of marker genes such as TIPP [7], MetaPhyler [8], and mOTU [9], and (ii) methods that depend on a clade of specific marker genes such as MetaPhlAn [10, 11]. These marker-gene based methods can achieve high accuracy when the reads come from genomes represented by the marker gene database. Otherwise, they only achieve low-level of sensitivity. The running time varies depending on the statistical methods used in each method.

The k-mer based methods use DNA composition as a characteristic to achieve taxonomic annotation. The key idea in these methods is to map the k-mers of each read to a database of k-mers, and then, based on different decision criteria, each read is assigned a taxonomic annotation [5, 12–15]. For example, Kraken [5] uses an exact match to align the overlapped k-mers of the queries with a k-mer reference database, instead of an inexact match of the complete sequence used in the naive-similarity based methods. Because of the exact matching on short k-mers, many efficient data structures can be implemented for searching the k-mer database which allows the k-mers based methods to be extremely fast and require little computation time. It was recently shown that these methods can achieve genus level sensitivity comparable to the naive-similarity methods but with higher precision [16]. At the same time, however, these methods are not robust to sequences that have a high sequencing error rate because they are based on exact matching to the reference database.

In addition to the benchmarking presented in this paper, the study [16] has also revealed that different methods could generate variation in taxonomic output profiles for the same input dataset. Sample type, sequencing error, and read length are the main factors that cause variation. This inconsistency in the predicted taxonomic annotations presents a challenge to investigators in the selection of the identification methods and interpretation of annotations. Although it has been proven that the taxonomic profile obtained from the naive-similarity methods produces a large number of false positives, a vast array of researchers are still dependent on them because they do not want to sacrifice the high level of sensitivity in order to obtain fewer false positives.

In this work we present a novel framework, WEVOTE (WEighted VOting Taxonomic idEntification), which takes advantage of three categories of the taxonomic identification methods; naive-similarity methods, k-mer methods, and marker-based methods. WEVOTE combines the high sensitivity of the naive similarity methods, the high precision of the k-mer methods, and the robustness of the marker-based methods to identify novel members of a marker family from novel genomes [7].

Materials and Methods

The WEVOTE framework and core algorithm

The core of WEVOTE is a weighting scheme organized as a taxonomic tree tallying the annotations from N different taxonomic identification methods. As shown in Fig 1, the input to the WEVOTE is the raw reads of a microbiome sample. First, each of the N identification methods independently assigns a taxon for each read. If any method fails to classify the read based on the given threshold, the WEVOTE preprocessing phase assigns 0 as a taxon, indicating that the read is unclassified by the corresponding method. Then, WEVOTE identifies the taxonomic relationship of

the N taxa per read based on the pre-configured taxonomy tree structure and casts a vote to the final taxon, which may be a common ancestor of the N taxa. Although the current version of our method only includes five methods, the voting scheme is flexible and allows for the inclusion or removal of different methods.

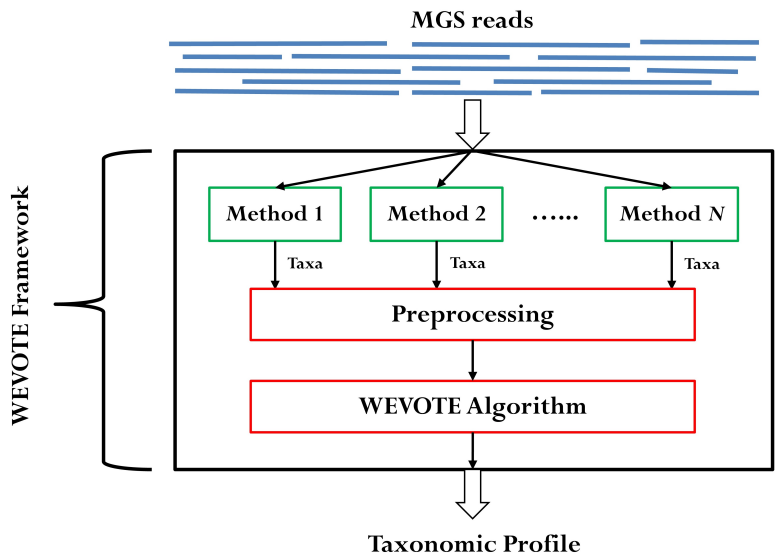

Fig 1. Schematic diagram of the WEVOTE framework. The input to the WEVOTE is the raw reads of the sample. First, each of the identification methods independently assigns a taxon to each read. Then, WEVOTE identifies the taxonomic relationship of the N taxa based on the pre-configured taxonomy tree structure and determines the final taxon assigned to each read.

The WEVOTE utilizes a resolved version of the NCBI taxonomy tree as a backbone for its decision algorithm. This resolved phylogeny tree only contains the nodes that have a taxon corresponding to one of the standard taxonomic ranks (Super-kingdom, Phylum, Class, Order, Family, Genus, and Species). This backbone structure facilitates and accelerates the choice of a consensus taxon based on the taxonomic annotations received from each identification tool. The decision scheme in WEVOTE is shown in Algorithm 1. Here, N denotes the number of tools used in the WEVOTE pipeline; C the number of tools that are able to classify the read at any taxonomic rank, i.e., taxon $\neq 0$; and A the number of tools that agreed upon the WEVOTE decision. The relationship $N \geqslant C \geqslant A$ always holds.

In the case that no single tool can classify the read, WEVOTE will accordingly fail to classify the read and give it a taxon 0 and score of 0. Otherwise, WEVOTE starts by building a weighted tree for each read from the taxa reported by individual tools. The weighted tree is a tree that comprises the nodes of the identified taxa along with their ancestors' taxa including the root. For each identified taxon, the weight of each node on the weight tree is incremented by one. The weight of any node on the final weighted tree represents the number of tools that agreed on this particular

node. Afterwards, WEVOTE annotates the read with the taxon of the node that has the highest weight from the root to that node taxon (RootToTaxon), with the added condition that the node itself has more weight than the WEVOTE threshold. This threshold can be set as half of the number of tools able to classify this read (C). In the case where more than one node satisfies the WEVOTE condition, then the Least Common Ancestor (LCA) of these nodes will be assigned as WEVOTEs decision.

The scoring scheme works as follows. If the number of tools that classified the read (C) equals the number of tools that agreed on the WEVOTE decision (A), then the WEVOTE score will be calculated based on Eq.(1)

$$score = \frac{A}{N} \tag{1}$$

Otherwise, we have $A < C$; the score will be calculated using Eq.(2), where k is an arbitrary number > 1.

$$Score = \frac{A}{N} - \frac{1}{k * N} \tag{2}$$

The choice of the constant k depends on how stringent one wishes to penalize the disagreement among individual tools that are able to classify the read but do not agree with the WEVOTE decision. A small value of k leads to a small WEVOTE score, implying more penalty is placed on the WEVOTE decision score, and vice versa. This scoring scheme makes the score satisfy the condition of $(A - 1)/N < score < A/N$.

Algorithm 1 The WEVOTE Decision Scheme

```
procedure WEVOTE (N taxa for each read)
    for each (read ∈ sequence file) do
        if (C == 0) then
            read.Taxon = 0
            read.DecisionScore = 0
            read.NumAgreedTools = 0
        else if (C ⩾ 1) then
            build a weighted tree of the reported taxa
            Threshold = ceiling(C/2)
            MaxWeight = 0
            MaxNode = 0
            for each (node ∈ Weighted Tree and Weight(node) ⩾ Threshold) do
                if (RootToTaxon(node) > MaxWeight) then
                    Max=node
                    MaxTaxon=node
                else if (RootToTaxon(node) == MaxWeight) then
                    MaxTaxon=LCA(node, MaxTaxon)
            read.Taxon = MaxTaxon
            read.NumAgreedTools = weight(read.Taxon)
            if (A == C) then
                read.DecisionScore ← A/N
            else
                read.DecisionScore ← (A/N) − (1/(k ∗ N))
```

In order to demonstrate the decision scheme described in the WEVOTE algorithm, the case scenarios of WEVOTE for $N = 3$ are shown in Fig 2.

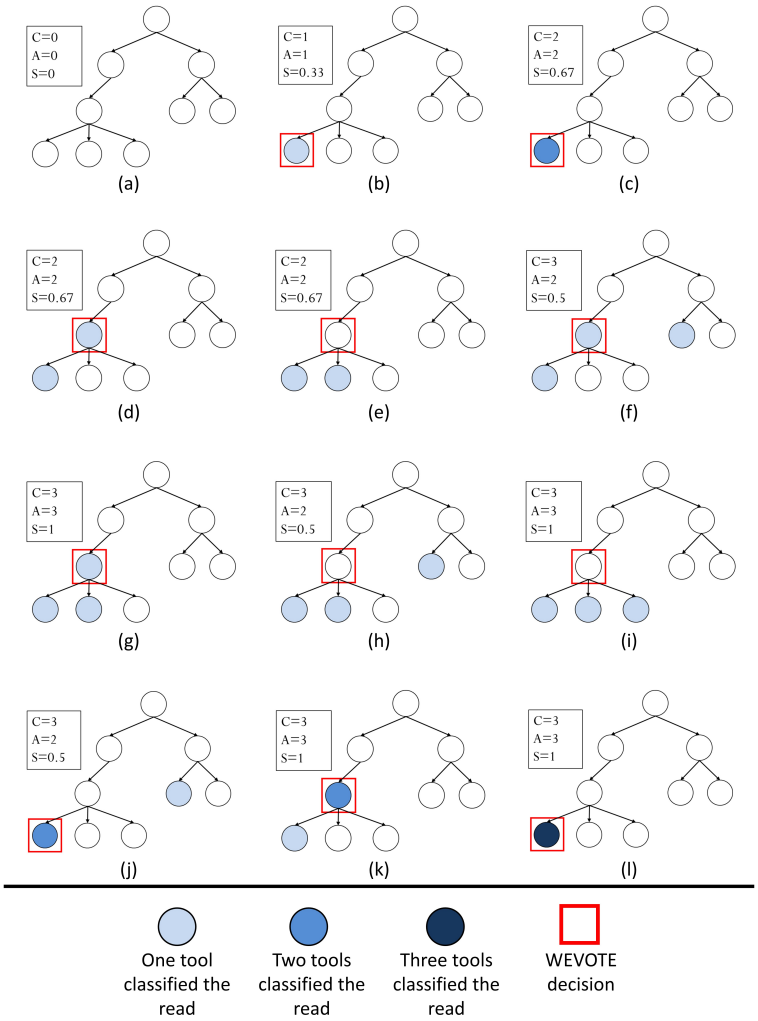

Fig 2. WEVOTE case scenarios using three tools. *C* denotes the # tools able to classify the read, *A* stands for the # of tools that agreed with the WEVOTE Decision, and *S* stand for WEVOTE score. Scenarios are shown for: (a) None of the three tools classified the read; (b) Only one tool classified the read; (c) Two tools classified the read with the same taxon; (d, e) Two tools classified the read with two different taxa; (f-i) Three tools classified the read with three different taxa; (j, k) Three tools classified the read, two taxa are identical, and the other is different; and (l) Three tools identified the read with the same taxon.

The tools used in the current implementation

In our current implementation of WEVOTE, we used BLASTN to represent the naive-similarity method, Kraken [5] and CLARK [12] as the identification tools representing the k-mer methods, and TIPP [7] and MetaPhlAn [10, 11] representing the marker-based methods. The five tools were chosen since they are widely used and represent the three major categories of taxonomic identification methods.

We favored BLASTN over MegaBlast because of its greater sensitivity. The primary reason for the increased sensitivity in BLASTN is the use of a shorter word size as a search seed. Thus, BLASTN is better than MegaBlast in finding alignments for sequences that have a sequencing error that occurs after a short length of matched bases (i.e., the initial exact match is shorter).

Kraken assigns taxonomic annotations to the reads by splitting each sequence into overlapping k-mers [5]. Each k-mer is mapped to a pre-computed database where each node in the database is the lowest common ancestor (LCA) taxon of all genomes that contain that k-mer. For each read, a classification tree is computed by obtaining all the taxa associated with the k-mers in that read. The number of k-mers mapped to each node in the classification tree is assigned as a weight for this node. The node that has the highest sum of weights from the root to leaf is used to classify the read. Kraken is an ultra-fast and highly precise for data involving small sequencing error. CLARK is a recently released tool that is very similar to Kraken and also based on k-mers. It is reported to be faster and more accurate than Kraken at the genus/species level [12]. The fundamental difference between Kraken and CLARK is their k-mers database backbone. Kraken has only one database that can serve for the classification of metagenomic reads at any taxonomic rank. If more than one genome shares the same k-mer, Kraken assign this k-mer to their lowest common ancestor (LCA) taxon. CLARK, on the other hand, builds an index for each taxonomic rank at which the user wishes to classify. Each level's index has only the discriminative k-mers that distinguish its taxa from each other.

TIPP (Taxonomic Identification and Phylogenetic Profiling) is considered a state-of-the-art tool based on a set of marker genes. It uses a customized database of 30 marker genes [17] which mostly are universal single-copy genes. First, it performs multiple sequence alignment of each marker gene set, then builds a phylogeny tree for each marker gene. Also, it builds a resolved taxonomy tree of these marker genes. Then, it uses SATe [18] to decompose the tree of each marker gene to many sub-trees. Afterward, TIPP uses HMMER software [19] to build a Hidden Markov Model (HMM) for each of the sub-trees. For each query read, TIPP uses HMMER again to align the query to the HMMs. Then, TIPP uses the alignments to the HMM that have an alignment score and statistical support greater than a group of pre-set values, and places them on the precomputed taxonomic tree using pplacer [20] in order to assign taxonomy to the query. Although the sensitivity of TIPP is low, it has been shown that it can precisely identify the reads containing high sequencing error or novel members of a marker family from novel genomes [7]. The other tool chosen for this category in our implementation is MethPhlAn. MethPhlAn has a set of clade-specific marker genes. The marker set was built from the genomes available from the Integrated Microbial Genomes (IMG). For a given read, MetaPhlAn compares the read against the precomputed marker set using nucleotide BLAST searches in order to provide clade abundances for one or more sequenced metagenomes.

Results and Discussion

A dozen of simulated datasets have been used in the evaluation of various taxonomic identification tools. In our assessment, we selected fourteen simulated datasets as shown in Table 1. Our choice was based on the ability of these datasets to provide the true identity assignment for each read rather than true relative abundance at each taxonomic level. This information allows for the evaluation of WEVOTE based on various metrics in addition to the assessment of relative abundance.

The first three datasets were used in the evaluation of Kraken [5]. The HiSeq and MiSeq datasets are simulated from sequences obtained from non-simulated microbial projects but were sequenced using two different platforms, i.e., Illumina HiSeq and Illumina MiSeq. The simBA5 is a simulated dataset with a higher percentage of error to mimic increased sequencing errors. Hence, it can be used to measure the ability of each tool to handle non-simulated sequencing data. The simHC20 dataset was used to benchmark CLARK [12] and it contains 20 subsets of long Sanger reads from various known microbial genomes.

The other ten datasets were used in MetaPhlAn [10] evaluations. The HC1 and HC2 consist of reads from high-complexity, evenly distributed metagenomes that contain 100 genomes, and LC1–LC8 consist of low-complexity, log-normally distributed metagenomes that contain 25 genomes. The reads from all ten MetaPhlAn were sampled from KEGG v54 [21] with a length of 100 bp and an error model similar to real Illumina reads.

Table 1. The Benchmarking Datasets

Source	Dataset	# reads	length (bp)	# genomes
Kraken	**HiSeq**	10,000	92	10
	MiSeq	10,000	156	10
	simBA5	10,000	100	1,967
CLARK	**simHC20**	10,000	951	20
MetaPhlAn	**HC1**	999,998	88	100
	HC2	999,991	88	100
	LC1	249,995	88	25
	LC2	250,000	88	25
	LC3	250,000	88	25
	LC4	249,999	88	25
	LC5	249,999	88	25
	LC6	250,002	88	25
	LC7	250,000	88	25
	LC8	250,000	88	25

The WEVOTE Benchmarking

Our benchmarking was carried out with two models of WEVOTE: (i) WEVOTE ($N = 3$) including BLASTN, TIPP and Kraken; and (ii) WEVOTE ($N = 5$) including BLASTN, TIPP, MetaPhlAn, Kraken, and CLARK. As described previously, BLASTN represents the naive-similarity method; TIPP and MethPhlAn belong to the category of the marker-based methods; and Kraken and CLARK belong to the category of the k-mer based methods. The default parameter values were set for the individual tools and the score penalty in WEVOTE was set at $k = 2$ (see Appendix A for the full details about the commands used in the command-line). Regarding WEVOTE, we reported all results with a minimum number of tools which agree on the WEVOTE decision equal to 1.

We first looked at how accurately individual reads have been annotated at each taxonomic rank using sensitivity and precision, which are defined in Eq.(3) and Eq.(4), respectively. For each rank l in a simulated dataset:

$$\text{Sensitivity}_{(l)} = \frac{TP_l}{P_l} \tag{3}$$

$$\text{Precision}_{(l)} = \frac{TP_l}{TP_l + FP_l} \tag{4}$$

where P_l denotes the number of reads annotated with some taxon at rank l in the original dataset; TP_l the number of reads correctly annotated at rank l; and FP_l the number of reads incorrectly annotated at rank l. It is observed from Fig 3 that WEVOTE achieves the highest level of precision and a level of sensitivity that is second only to BLASTN at the species rank. It is also interesting to observe that the precision level achieved by WEVOTE ($N = 5$) is lower than that of WEVOTE ($N = 3$), indicating that including multiple tools in each category may play against WEVOTE for a correct annotation. At all other taxonomic ranks, WEVOTE outperforms all the other individual tools in terms of sensitivity and precision in most cases (Table S2).

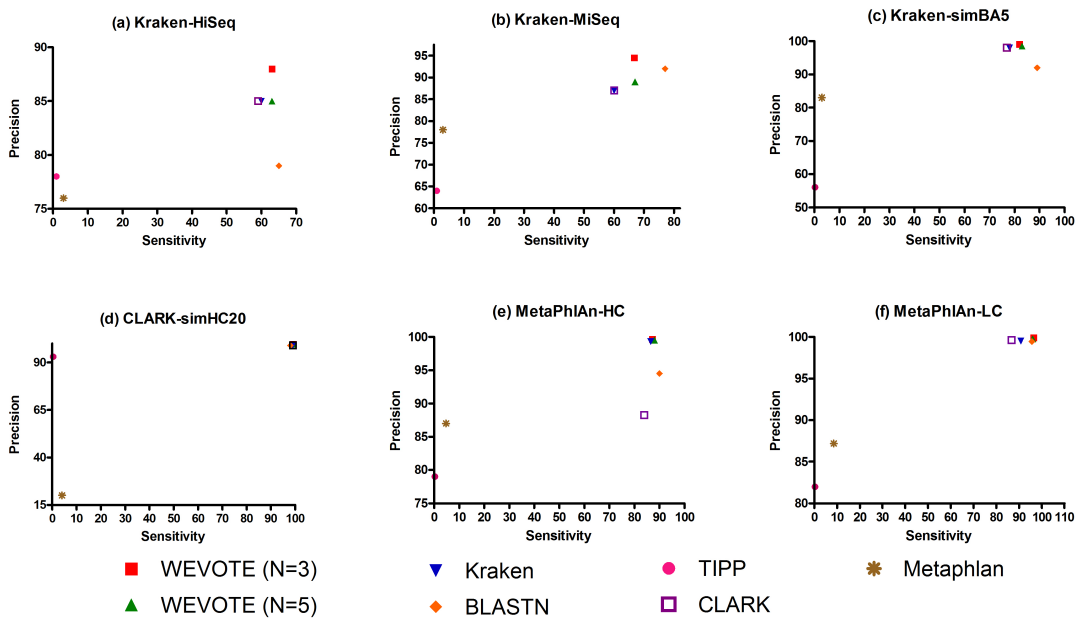

Fig 3. The sensitivity and precision at the species level for all tools. The sensitivity and precision reported for each tool at the species level on each simulated dataset, with the exception of the datasets in MetaPhlAn-HC and MetaPhlAn-LC being the average over two HC and eight LC datasets, respectively.

Since our motivation for the development of WEVOTE is the reduction of the false positive rate FPR while maintaining a high level of sensitivity, we calculated the FPR at each taxon level l as defined in Eq.(5).

$$\mathrm{FPR}_l = \sum_{x \in \{C_l \mid P_x > T_x\}} (P_x - T_x) \tag{5}$$

Here, C_l is the union of all taxa that are in the true and predicted profiles at each taxonomic rank l. For each taxon x at rank l, P_x is the predicted relative abundance and T_x is the true relative abundance at taxonomic rank l. FPR measures the deviation of the relative abundance of reads annotated at a taxonomic rank from the true relative abundance in addition to the relative abundance of reads annotated at a taxonomic rank that is not originally present in the sample, i.e., $T_x = 0$ in this case. Another important metric, classification rate (CR), is the percentage of the classified reads at each taxonomic rank l. This quantity, defined in Eq. (6) was calculated at each taxonomic rank l.

$$CR_{(l)} = \frac{TP_l + FP_l}{P_l} \tag{6}$$

This metric is crucial for WEVOTE because most of the tools that are based on an LCA algorithm, such as MEGAN [6], need to move the annotation to an a higher taxonomic rank, thus resulting in a lose of resolution at the lower taxonomic ranks; genus and species.

It can be seen from Fig 4, Tables S5 and S6 that the FPRs obtained from WEVOTE, specifically with $N = 3$ are always lower than BLASTN, Kraken, and CLARK at all taxonomic ranks. The effect of WEVOTE on the FPR reduction appears clearly at the species level. TIPP has nearly zero FPR at all taxonomic ranks, but it has a much lower classification rate. Moreover, WEVOTE achieved the highest classification rate among all tools at the ranks of Domain, Phylum, Class, Order, and Family. At the Genus level, WEVOTE scored the second after BLASTN by a marginal difference. At the species level, although WEVOTE is still second behind the BLASTN, the difference is more detailed. This is because WEVOTE moves the annotation from the Species level to the rank above if there is not enough support for classification at the Species level. This slight loss of resolution is due to the correction for the false positives as described in Fig 4 (the left panel). One reason for the low classification rate in TIPP and MetaPhlAn may be because the current marker genes database used in TIPP or MetaphlAn do not contain sufficient markers for the genomes represented in the simulated datasets. Taken altogether, our analysis demonstrates that WEVOTE can achieve a substantial improvement in reducing the FPR compared to using any other individual tool that has a high level of sensitivity.

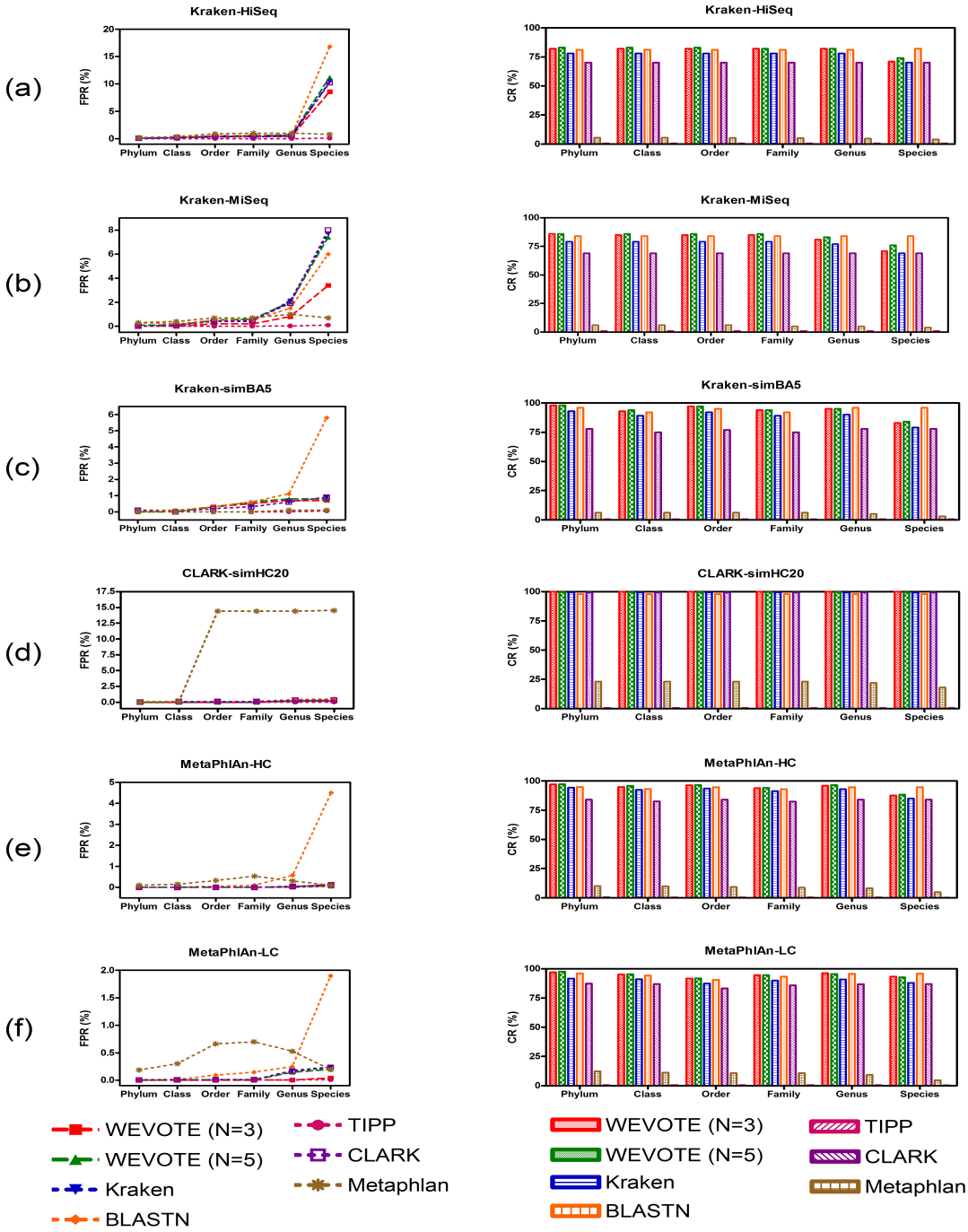

Fig 4. The False Positive Rate (FPR) and Classification Rate (CR). For simulated datasets, the FPRs (left) and CR (right) are calculated for all the tools. Results shown are for: (a) Kraken-HiSeq dataset; (b) Kraken-MiSeq dataset; (c) Kraken-simBA5 dataset; (d) CLARK-simHC20; (e) MetaPhlAn-HC and (f) MetaPhlAn-LC. The smaller the FPR is, the more accurate a metagenomic abundance profile is. A high Classification Rate at any taxonomic rank indicates that the corresponding identification tool is highly sensitive at this rank.

In addition, we calculated the Hellinger distance [22] (H_l) between a sample's metagenomic abundance profile generated by WEVOTE and its true abundance profile at each taxonomic rank l. The Hellinger distance measures the deviation of the predicted profile from the true profile. It is calculated as shown in Eq. (7). The $\sqrt{2}$ is added to the denominator to keep $0 \leq H \leq 1$. The definition of C_l, P_x, and T_x are the same as the ones used in calculating the FPR in Eq. (5).

$$H_l = \frac{\sqrt{\sum_{x \in C_l} (\sqrt{P_x} - \sqrt{T_x})^2}}{\sqrt{2}} \tag{7}$$

As the Hellinger distance represents an error distance, a small value is always preferable. Particularly, $H = 0$ means that the predicted profile is exactly the same as the true profile; while $H = 1$ means that the predicted profile is completely different from the true profile.

Fig 5 and Table S7 show the Hellinger distance between the true relative abundance profile and the profiles generated by all tools at different taxonomic levels. For all used simulated datasets, WEVOTE, particularly when $N = 3$, almost has the smallest Hellinger distance among all other individual identification tools across all taxonomic ranks. Although the Hellinger distance difference between WEVOTE and BLASTN is sometimes small, the interpretation is quite different. The error that originates from BLASTN is due to the false positive annotations while the error that originates from WEVOTE is due to the lack of support in annotating the read at the corresponding level. The large Hellinger distance in case of TIPP or MetaPhlAn is mainly because we included all the taxa in our calculation. Since TIPP has a small Classification Rate with P_x being near zero for many taxa that present in the dataset, this has led to the accumulation in the error distance.

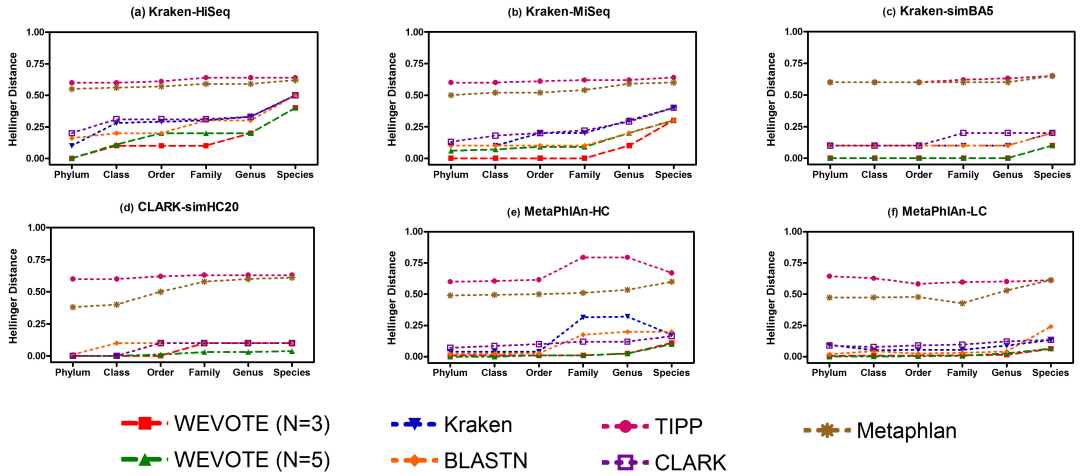

Fig 5. The Hellinger distance. The deviation between the predicted abundance profile and the true abundant profile was measured in terms of the Hellinger distance for each tool at different taxonomic ranks. Results shown are for: (a) Kraken-HiSeq dataset; (b) Kraken-MiSeq dataset; (c) Kraken-simBA5 dataset; (d) CLARK-simHC20; (e) MetaPhlAn-HC and (f) MetaPhlAn-LC. The lower the error, the more precise the corresponding tool is at the corresponding rank. $H = 0$ means that the predicted relative abundance profile is exactly the same as the true profile; while $H = 1$ means that the predicted profile is completely different from the true profile.

Lastly, we examined the details of various case scenarios that were encountered in the evaluation of the two WEVOTE variants, i.e., $N = 3$ and $N = 5$. The scatter plots in Fig. 6 show the percentages of annotations in which the individual tools agreed upon the WEVOTE decision for all the datasets. Table S4 shows the actual number of tools that agreed on the WEVOTE decision per datasets. It can be observed that the majority of WEVOTE annotations is determined based on more than $N/2$ agreements; 2 in the case of $N = 3$ and 3 in the case of $N = 5$. For only a small portion of each dataset, all the tools agreed on the WEVOTE decision. The interesting observation is that a very small portion out of all the annotated reads by WEVOTE, has agreed with one tool annotation in the case of N=3, or either 1 or 2 in the case of N=5. Therefore, if we set a threshold on WEVOTE to report the annotation that more than half the tools have to agree on, then then WEVOTE will increases the precision and its sensitivity will only be marginally affected as demonstrated in Fig. 7. We have chosen Kraken-HiSeq and Kraken-Miseq datasets only for this analysis because they had low precision among all the used taxonomic identification tools (Fig. 3), hence providing room for improvement.

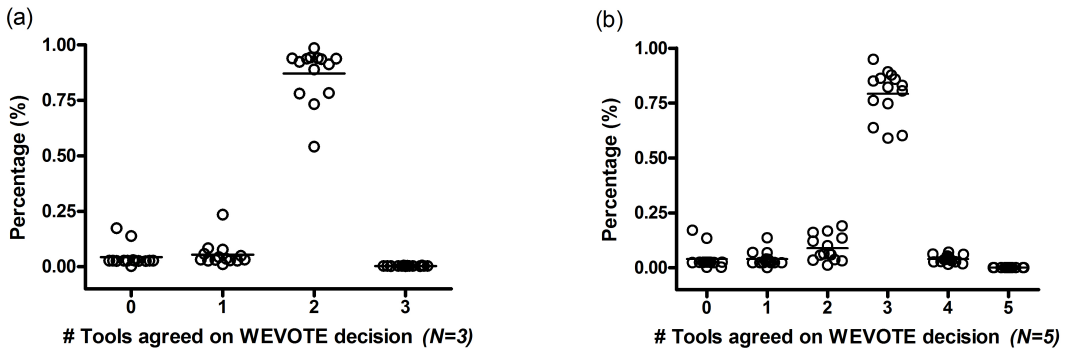

Fig 6. The percentage distribution of the number of individual tools that agreed on the WEVOTE decision for the 14 datasets. Here, 0 means that the read was not classified by any tools, 1 means that one tool has agreed on the WEVOTE assigned taxon for the read, and so on. A=3 in the case of (a) means that all the 3 tools agreed with WEVOTE on its assigned taxon for the corresponding read, A=5 in case of (b) means that all the used 5 tools agreed with WEVOTE on its assigned taxon for the corresponding read.

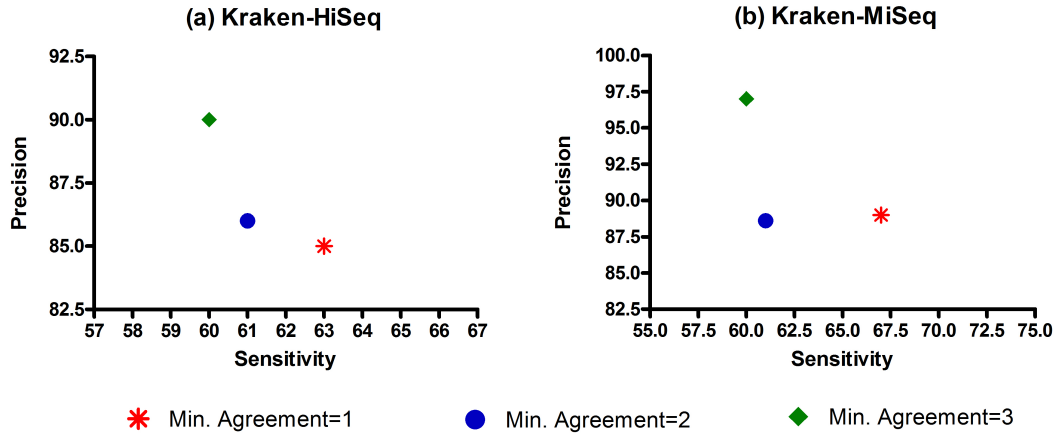

Fig 7. The sensitivity and precision at the species level for the WEVOTE (N=5) using different thresholds for the minimum number of tools that agreed with WEVOTE decision. (a) Kraken-HiSeq dataset; and (b) Kraken-MiSeq dataset.

Computational resources and running performance

All the experiments were performed on the supercomputer (EXTREME) at the University of Illinois at Chicago. To benchmark different WEVOTE, we used only one node with 16 cores (Intel Xeon E5-2670 @ 2.60 GHz, cache size of 20 MB, and 128 GB RAM). Since the WEVOTE core algorithm and all the used individual tools are parallelizable, we utilized 16 threads for all experiments conducted in this work. Due to

the high requirement on the memory for constructing Kraken and CLARK databases, we used the Highmem machine on EXTREME which has 1TB RAM specification.

In order to achieve the maximum performance from Kraken and CLARK, we used the default versions of the two tools, which require at least 80 GB of RAM. Therefore, if there is only a limited amount of memory available, users can run these tools using their mini versions, i.e., MiniKraken and CLARK-l, which only require 4 GB of RAM. In this case, the output could be 11%-25% less sensitive, but it will still preserve a high level of precision. The WEVOTE algorithm is particularly useful in this case because it can exploit the high precision level of Kraken and CLARK without using high memory machines and compensate the sensitivity by using BLASTN.

Table 2 shows the running time for each tool per dataset. For MetaPhlAn, which include two HC datasets and eight LC datasets, the running time is presented as the average over the datasets. The standard deviation of each category is also provided, however, the details for all individual datasets can be found in Table S8. For all used simulated datasets, Kraken and CLARK finished in less than 3 minutes. For BLASTN, the most time-consuming tool that is currently implemented in WEVOTE pipeline, its running time is proportional to the number of reads and the read length in a dataset. The WEVOTE algorithm, whether with $N = 3$ or $N = 5$, finished in less than 33 seconds for all the datasets. The WEVOTE core algorithm is mainly affected by the number of the used tools, and, more specifically, the number of tools that identified taxa for the reads. The total time of the entire WEVOTE pipeline is the summation of the running times of the individual tools and the WEVOTE algorithm. It can be reduced if the tools are run in parallel, but it will be primarily dominated by the time required by BLASTN.

Table 2. Running time of the used tools.

Simulated Dataset	Kraken (min)	BLASTN (min)	TIPP (min)	CLARK (min)	MetaPhlAn (min)	WEVOTE algorithm [N=3] (sec)	WEVOTE algorithm [N=5] (sec)	WEVOTE Pipeline [N=5](min)
HiSeq	<1	2	4	1	1	0.1	1	10
MiSeq	<1	8	4	1	1	0.2	1	16
simBA5	<1	7	3	1	1	0.1	1	14
simHC20	<1	9	5	1	1	0.3	1.5	19
HC (std)	2 (0.0)	30 (1.4)	14 (0.0)	3 (0.0)	2 (0.0)	5 (0.3)	32 (1.4)	53 (1.4)
LC (std)	1 (0.0)	9 (2.9)	8 (0.5)	2 (0.0)	1 (0.5)	1 (0.1)	7 (0.9)	23 (3.5)

Conclusion and future work

We have developed the WEVOTE framework for the consolidation of taxonomic identifications obtained from different classification tools. The performance evaluation based on the fourteen simulated microbiome datasets consistently demonstrates that WEVOTE achieves a high level of sensitivity with the lowest false positive rate

compared to the individual methods across different taxonomic levels. The major advantage of the WEVOTE pipeline is that a user can make the choice of which tools to use in order to explore the trade-off between sensitivity, precision, time, and memory. The WEVOTE architecture is flexible such that additional taxonomic tools can be easily added, or the current tools can be replaced by improved ones. Moreover, the score assigned to the taxon for a read indicates the confidence level of the assignment. This information is especially useful for the assessment of false positive annotations at a particular taxonomic level. The classification score given by WEVOTE can be used for any downstream analysis that requires the high confidence of the annotated sequences. In our current implementation, we have used a uniform weight for each method to vote. However, we will explore the potential of incorporating different weighted votes for individual methods. Future work also includes the investigation of clinical microbiome samples and experimental validation for species of interest.

Abbreviations

WEVOTE: WEighted VOting Taxonomic idEntification method; MGS: MetaGenome Shotgun; LCA: Least Common Ancestor; TIPP: Taxonomic Identification and Phylogenetic Profiling; BLAST: Basic Local Alignment Search Tool; NCBI: National Center for Biotechnology Information.

Acknowledgments

We would like to thank Nam-Phuong Nguyen and Tandy Warnow from the Department of Computer Science, The University of Illinois at Urbana-Champaign for their fruitful discussions regarding MGS taxonomic identification methods. We also want to thank Brian Nguyen and Zahraa Hajjiri for their meaningful review of the paper. In addition, we want to thank Rachel Poretsky from UIC for allowing us to use her lab's Highmem node to construct Kraken and CLARK database.

Funding

This work is supported in part by UIC Chancellor's Research Award given to (AM), and by National Institutes of Health grants RO1 HL081663 and RO1 AI053878 to (PWF and DLP).

References

1. Langmead B, Trapnell C, Pop M, Salzberg SL. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome biology. 2009 1;10(3):R25. Available from: `http://genomebiology.com/2009/10/3/R25`.

2. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. Journal of molecular biology. 1990 10;215(3):403–10. Available from: http://www.sciencedirect.com/science/article/pii/S0022283605803602.

3. Zhang Z, Schwartz S, Wagner L, Miller W. A greedy algorithm for aligning DNA sequences. Journal of computational biology : a journal of computational molecular cell biology. 2004 1;7(1-2):203–14. Available from: http://online.liebertpub.com/doi/abs/10.1089/10665270050081478.

4. Brady A, Salzberg SL. Phymm and PhymmBL: metagenomic phylogenetic classification with interpolated Markov models. Nature methods. 2009 9;6(9):673–6. Available from: http://dx.doi.org/10.1038/nmeth.1358.

5. Wood DE, Salzberg SL. Kraken: ultrafast metagenomic sequence classification using exact alignments. Genome biology. 2014 1;15(3):R46. Available from: http://genomebiology.com/2014/15/3/R46.

6. Huson DH, Auch AF, Qi J, Schuster SC. MEGAN analysis of metagenomic data. Genome research. 2007 3;17(3):377–86. Available from: http://genome.cshlp.org/content/17/3/377.

7. Nguyen NP, Mirarab S, Liu B, Pop M, Warnow T. TIPP: taxonomic identification and phylogenetic profiling. Bioinformatics (Oxford, England). 2014 12;30(24):3548–55. Available from: http://bioinformatics.oxfordjournals.org/content/30/24/3548.long.

8. Liu B, Gibbons T, Ghodsi M, Treangen T, Pop M. Accurate and fast estimation of taxonomic profiles from metagenomic shotgun sequences. BMC genomics. 2011 1;12 Suppl 2(2):S4. Available from: http://bmcgenomics.biomedcentral.com/articles/10.1186/1471-2164-12-S2-S4.

9. Sunagawa S, Mende DR, Zeller G, Izquierdo-Carrasco F, Berger SA, Kultima JR, et al. Metagenomic species profiling using universal phylogenetic marker genes. Nature methods. 2013 12;10(12):1196–9. Available from: http://dx.doi.org/10.1038/nmeth.2693.

10. Segata N, Waldron L, Ballarini A, Narasimhan V, Jousson O, Huttenhower C. Metagenomic microbial community profiling using unique clade-specific marker genes. Nature methods. 2012 8;9(8):811–4. Available from: http://dx.doi.org/10.1038/nmeth.2066.

11. Truong DT, Franzosa EA, Tickle TL, Scholz M, Weingart G, Pasolli E, et al. MetaPhlAn2 for enhanced metagenomic taxonomic profiling. Nature Methods. 2015 9;12(10):902–903. Available from: http://dx.doi.org/10.1038/nmeth.3589.

12. Ounit R, Wanamaker S, Close TJ, Lonardi S. CLARK: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers. BMC Genomics. 2015 3;16(1):236. Available from: http://bmcgenomics.biomedcentral.com/articles/10.1186/s12864-015-1419-2.

13. Ames SK, Hysom DA, Gardner SN, Lloyd GS, Gokhale MB, Allen JE. Scalable metagenomic taxonomy classification using a reference genome database. Bioinformatics (Oxford, England). 2013 9;29(18):2253–60. Available from: http://bioinformatics.oxfordjournals.org/content/29/18/2253.short.

14. Menzel P, Lee Ng K, Krogh A. Kaiju: Fast and sensitive taxonomic classification formetagenomics; 2015. Available from: http://biorxiv.org/content/early/2015/12/18/031229.abstract.

15. Rosen GL, Reichenberger ER, Rosenfeld AM. NBC: the Naive Bayes Classification tool webserver for taxonomic classification of metagenomic reads. Bioinformatics (Oxford, England). 2011 1;27(1):127–9. Available from: http://bioinformatics.oxfordjournals.org/content/27/1/127.

16. Lindgreen S, Adair KL, Gardner PP. An evaluation of the accuracy and speed of metagenome analysis tools. Scientific Reports. 2016 1;6:19233. Available from: http://www.nature.com/srep/2016/160118/srep19233/full/srep19233.html.

17. Wu M, Eisen JA. A simple, fast, and accurate method of phylogenomic inference. Genome Biology. 2008 1;9(10):R151. Available from: http://genomebiology.com/2008/9/10/R151.

18. Liu K, Warnow TJ, Holder MT, Nelesen SM, Yu J, Stamatakis AP, et al. SATe-II: very fast and accurate simultaneous estimation of multiple sequence alignments and phylogenetic trees. Systematic biology. 2012 1;61(1):90–106. Available from: http://www.ncbi.nlm.nih.gov/pubmed/22139466.

19. Eddy SR. Profile hidden Markov models. Bioinformatics (Oxford, England). 1998 1;14(9):755–63. Available from: http://www.ncbi.nlm.nih.gov/pubmed/9918945.

20. Matsen FA, Kodner RB, Armbrust EV. pplacer: linear time maximum-likelihood and Bayesian phylogenetic placement of sequences onto a fixed reference tree. BMC bioinformatics. 2010 1;11:538. Available from: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3098090{&}tool=pmcentrez{&}rendertype=abstract.

21. Kanehisa M, Goto S, Furumichi M, Tanabe M, Hirakawa M. KEGG for representation and analysis of molecular networks involving diseases and drugs. Nucleic acids research. 2010 1;38(Database issue):355–60. Available from:

http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2808910{&}tool=pmcentrez{&}rendertype=abstract.

22. Deza E, Deza MM. Encyclopedia of Distances. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009. Available from: http://link.springer.com/10.1007/978-3-642-00234-2.

Appendix (A)

We provide information on the command line, the databases and the software version that were used for the execution of the individual tools: Kraken, TIPP, BLASTN, CLARK, and MethPhlAn. The complete path to each executable of database is not shown here for clarity:

Kraken

Step 1: Map the reads file to the Kraken database:

```
   $kraken --db <kraken-db> -fasta-input --threads 16
--output <KrakenOutput> [input-file.fa]
```

Step 2: Generate Kraken report:

```
   $kraken-report --db <kraken-db> <KrakenOutput> >
[KrakenOutput.report]
```

Software version: kraken-0.10.5-beta.

Database: used the kraken-build script to download and configure the standard Kraken database. This downloads NCBI taxonomic information, as well as the complete genomes in RefSeq for the bacterial, archaeal, and viral domains (Downloaded on 11/14/2015).

BLASTN

Map the reads file to the NCBI database and report the top hit:

```
   $blastn -db nt -query <input-file.fa> -out
<NaiveOutput> -outfmt "6 qseqid sseqid sgi staxids
length qstart qend sstart send pident evalue score
bitscore stitle" -num_threads 16 -perc_identity 90
-max_target_seqs 1 -evalue 1e-5 -best_hit_score_edge 0.05 -best_hit_overhang
0.25
```

Software version: ncbi-blast-2.2.29+-x64-linux

Database: NCBI NT (Downloaded on 12/20/2015).

TIPP

Map the reads file to the database of 30 marker genes and reports the taxon of the classified reads:

```
$run_abundance.py -f [input-file.fa] -c
/.sepp/tipp.config -x 16 -d [TIPPOutput]
```

Software version: Downloaded the source code from: https://github.com/smirarab/sepp.git on (Downloaded on 12/1/2015).

Database: Downloaded the references datasets from www.cs.utexas.edu/ phylo/software/sepp/tipp.zip on (Downloaded on 12/1/2015).

CLARK

Step 1: Configure the setting and choose the database:

```
$set_targets.sh <CLARK-DB> bacteria viruses
```

Step 2: Map the reads file to the CLARK database:

```
$classify_metagenome.sh -O <input-file.fa> -R
<output-prefix> -n 16
```

Software version: CLARKSCV1.2.3

Database: used the download_data.sh script to download NCBI taxonomic information, as well as the complete bacterial and virus genomes (Downloaded on 4/20/2016).

MetaPhlAn

Map the reads file to the database of MetaPhlAn marker genes:

```
$python metaphlan.py <input-file.fa> --bowtie2db
bowtie2db/mpa --bt2_ps sensitive-local --bowtie2out
<output-prefix.bt2out> --input_type multifasta --nproc 16 > <output-file>
```

Software version: MetaPhlAn version 1.7.7

Database: The same marker genes database that downloaded with MetaPhlAn version 1.7.7 (Downloaded on 12/13/2015).

WEVOTE

The input to WEVOTE algorithm is a CSV file. Each file has information about one read from the sequence fasta file. This information is in the form of <read header, tool #1 taxon, tool #2 taxon, ..., tool #N taxon >. In the case of inability of any tool to classify a read, taxon should be zero. WEVOTE, then, use this input file along with the NCBI taxonomy database to annotate the sequences:

```
   $wevote -i <input-file.csv> -d <taxonomy-database>
-p <output-prefix> -n 16 -k 2
```

Software version: WEVOTE version 1.0.0

Database: NCBI taxonomy database (Downloaded on 4/17/2016).