GenomeFace: a deep learning-based metagenome binner trained on 43,000 microbial genomes

Richard Lettich[1], Robert Egan[1,2], Robert Riley[1,2], Zhong Wang[2,3], Andrew Tritt[1], Katherine Yelick[1,4], Aydın Buluç[1,4]

[1]Applied Math and Computational Research Division, Lawrence Berkeley National Laboratory.

[2]Joint Genome Institute, Lawrence Berkeley National Laboratory.

[3]Department of Molecular & Cell Biology, University of California, Merced.

[4]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley.

Abstract

Metagenomic binning, the process of grouping DNA sequences into taxonomic units, is critical for understanding the functions, interactions, and evolutionary dynamics of microbial communities. We propose a deep learning approach to binning using two neural networks, one based on composition and another on environmental abundance, dynamically weighting the contribution of each based on characteristics of the input data. Trained on over 43,000 prokaryotic genomes, our network for composition-based binning is inspired by metric learning techniques used for facial recognition.

Using a task-specific, multi-GPU accelerated algorithm to cluster the embeddings produced by our network, our binner leverages marker genes observed to be universally present in nearly all taxa to grade and select optimal clusters of sequences from a hierarchy of candidates.

We evaluate our approach on four simulated datasets with known ground truth. Our linear time integration of marker genes recovers more near complete genomes than state of the art but computationally infeasible solutions using them, while being over an order of magnitude faster. Finally, we demonstrate the scalability and acuity of our approach by testing it on three of the largest metagenome assemblies ever performed. Compared to other binners, we produced 47%-183% more near complete genomes. From these datasets, we present the genomes of over 3000 new candidate species which have never been previously cataloged, representing a potential 4% expansion of the known bacterial tree of life.

1 Introduction

Metagenomic assembly, the simultaneous assembly of entire microbial communities, introduces additional challenges beyond those already encountered in the assembly of individual isolate genomes. Factors such as varying abundance of species, repetitive regions, low sequencing coverage, sequencing errors, the presence of multiple strains of a single species, horizontal gene transfer, and highly conserved or homologous sequences co-located in multiple species all cause ambiguities during the assembly process.

In practice, such issues inevitably lead to metagenome assemblers producing a mixture of fragmented genomes, each dispersed across an initially unknown number of contigs rather than an ideal single scaffold per genome.

To deconvolve and group these assembled sequences by their taxa of origin, tools known as metagenome binners are used. These typically work by applying clustering algorithms with per-sample sequence abundance and/or features derived from their composition, most often via some form of 'genomic signature' that quantifies the relative frequency of oligonucleotide patterns within the sequences [1]. Popular metagenomic binners, such as MetaBAT 2 [2] and VAMB [3], have utilized sequences' normalized Tetra-Nucleotide Frequencies (TNF) as a way to discriminate between genomes based on the composition of the sequences themselves.

The design and effectiveness of such techniques are rooted in the observation that oligonucleotide signatures, such as Tetra-Nucleotide Frequencies, tend to remain consistent throughout genomes. Notably, larger variations exist between genomes, correlating with phylogenetic distance. While the exact reasons for this are not fully understood, it is believed that the phenomenon is mediated by species-specific replication and repair machineries, evolutionary pressures, and to a lesser extent, environmental factors [4].

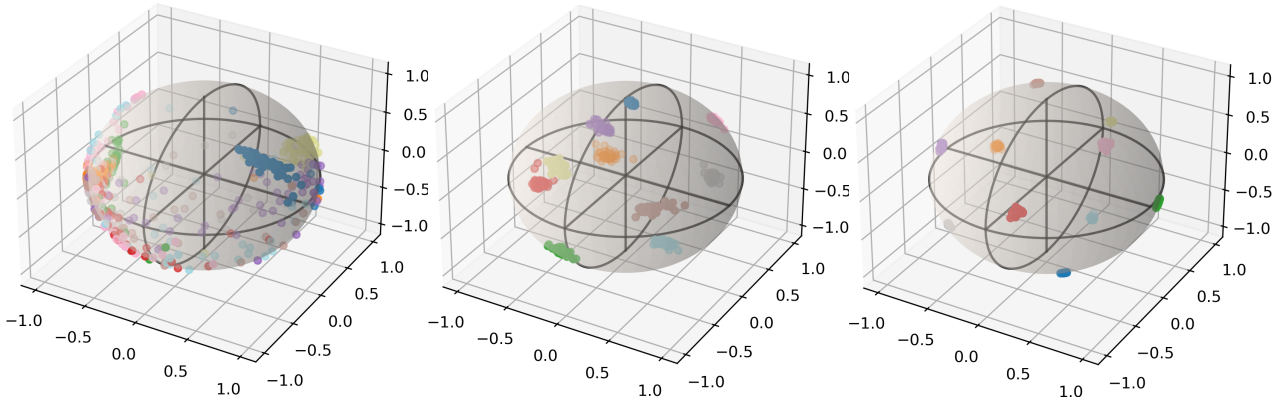
Recent studies have highlighted the advantages of large-scale metagenomic coassembly [5–7]. By aggregating multiple metagenomic samples from the same environment, coassembly provides a more holistic understanding of microbial communities compared to the separate assembly of each sample, commonly known as 'multiassembly'. This approach not only aids in detecting community members with lower sequencing depths but also contributes to the reconstruction of more complete genomes, offering a broader and more detailed view of the environment's taxonomic diversity. While coassembly improves microbial representation, it also significantly increases the computational complexity and precision needed in downstream binning processes, due to the larger number of sequences and the variety of genomes present.

This paper introduces GenomeFace, a binner uniquely tailored for large-scale coassemblies. It was motivated by the challenges of distinguishing between numerous genomes with high precision, integrating marker genes at scale, and efficiently processing assemblies that comprise thousands of genomes and millions of sequences.

Previous research has suggested that generalizing TNF to oligonucleotide signatures with varying lengths, degenerate alphabets, and normalization methods may provide better or complementary signals for metagenome binners, and proposed further investigation into maximizing and unifying their discriminative power [4]. To this end, we draw inspiration from the field of computer vision and facial recognition to train a deep, densely connected neural network on simulated contigs from over 43,000 prokaryotic reference genomes using oligonucleotide frequencies of various lengths as input features. The network employs a softmax cross-entropy loss, incorporating modifications from CurricularFace [8] designed to enhance the discriminative ability of facial embeddings. Although we train the network as a classification problem, its ultimate goal is to produce an embedding that minimizes the distance between sequences from the same genome and maximizes the distance between sequences originating from different genomes, as to yield a robust metric for measuring the taxonomic similarity of sequences.

This style of training is typical for state-of-the-art networks used for facial recognition, however the methodology is not intrinsically linked to the task. Although initial practitioners of such techniques did not provide theoretical justification, it has been shown that minimizing cross-entropy loss can be seen as an approximate bound-optimization algorithm for minimizing a pairwise distance loss on a hypersphere [9], maximizing inter-class and minimizing intra-class variance. This insight forms the foundation for our neural networks, enabling us to train on tens of thousands of genomes without encountering the issues associated with more conventional metric learning loss functions at such a large scale. These problems fundamentally arise from attempting to learn directly on the distances between embeddings of sample pairs, due to the combinatorial explosion in the number of such pairings.

To group the sequences by taxa, we introduce a novel algorithm that leverages our embeddings alongside observed universal single copy genes, which are present in nearly all microbial life, to aid in the formation and selection of sequence clusters as a form of transductive learning.



(a) Start of Training (b) Later (c) End of Training

Fig. 1: Small scale example showing the training process of our neural network, training to classify classify contigs from 10 genomes. Each point represents a contig of length 5000 from a genome being trained on, and colored based upon genome they were drawn from. The spatial position is the embedding given by our neural network, which was modified to produce embeddings in $\mathbb{R}^3$ rather than $\mathbb{R}^{512}$ for illustrative purposes. At the end of training, the sequence embeddings exhibit high intergenome and low intragenome variance

Other metagenome binners have attempted to integrate universal single copy genes. For example, DASTool, an ensemble binner, takes the output of multiple metagenome binners and uses single copy marker genes to estimate the quality of each bin. At each iteration, it greedily selects an optimal bin to output, updating the non-selected candidate bins by removing sequences that occur in the greedily selected bin and then recalculating their score. MetaBinner and Semibin 2, on the other hand, greedily extracts bins that meet quality requirements, then reclusters the remaining sequences.

In contrast to the straightforward approach of using marker genes as a post-clustering oracle, our binner integrates them in a structured and well-defined way directly into the cluster formation process. Employing the embeddings and associated similarity metric produced by our neural networks, we construct hierarchy of clusters delineating various partitionings of the input sequences, finely gradating from all sequences belonging to a single genome or cluster, to each sequence being its own cluster. From this hierarchy, we precisely extract the non-redundant set of clusters that maximizes a well defined objective function that evaluates the perceived quality of the resulting genome bins. Our clustering algorithm's kernel is based on NVIDIA's implementation of the HDBSCAN, modified to efficiently integrate both of our embeddings and enable multi-GPU acceleration.

We demonstrate the outperformance of state-of-the-art binners on four of the simulated CAMI-II Human Microbiome datasets. Of no less importance, we showcase our binner's ability to scale by performing three case studies on large-scale co-assemblies of complex communities, each comprising thousands of species and assembled on the Summit supercomputer from trillions of base pairs of reads. This demonstrates its efficacy in capturing the substantial taxonomic diversity that necessitates such large-scale analyses, a feat recent state-of-the-art but more computationally expensive binners are not able to reach.

The significance of the ability to proce ss these types of data sets is immense. In our flagship case study analyzing the brackish wetlands of Twitchell Island, California, GenomeFace was able to produce 1,923 medium quality Metagenome Assembled Genomes spanning 69 phyla and candidate-phyla from a single assembly, each representing a unique species, with 90 percent of these species not previously cataloged. For perspective, we compare our analysis of only Twitchell Island's wetlands to two of the largest consorted efforts to study aquatic communities ever performed, *Tara Oceans* and *A genome catalogue of lake bacterial diversity and its drivers at continental scale* [10], both of which encompass microbial communities sampled from a diverse range of geographic locations. These studies documented 1,888 and 1,008 prokaryotic species, representing 29 and 22 phyla, respectively.

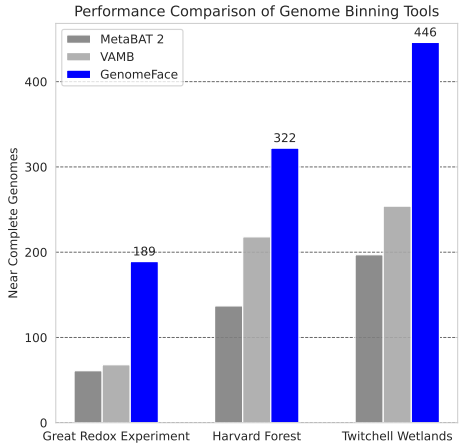
Furthermore, from our three case studies, we present over 3,000 new candidate species, marking a 4% expansion of the known bacterial tree of life.

The successful application of GenomeFace to these extensive co-assemblies stands as a proof of principle, signifying an advance in our ability to study the full taxonomic diversity of microbial ecosystems. This includes shedding light on microbial 'dark matter'–low-abundance organisms that have remained largely elusive and poorly understood–thereby enabling a path forward to further understand ecological interactions, nutrient cycles, the roles of microorganisms, and microbial ecology as a whole.

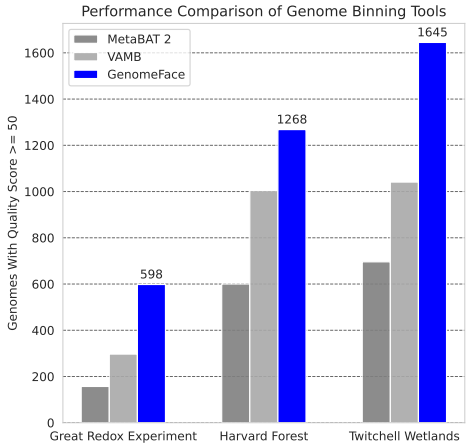
2 Results

We present our results in three parts.

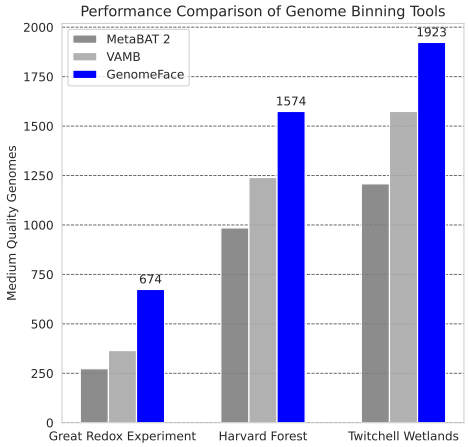
2.1 Real Datasets



((a)) Near Complete Quality Genomes (est. $\geq$ 90% Completeness, $\leq$ 5% Contamination by CheckM2) recovered By binner



((b)) Number of Genomes that Pass Quality Score Filter for Genome Taxonomy Database Inclusion (QS $\geq$ 50). QS is defined as est. Completeness $-$ 5 $\times$ Contamination



((c)) Medium Quality Genomes (est. $\geq$ 50% Completeness, $\leq$ 10% Contamination by CheckM2) recovered By binner

Evaluation

In each of our three case study datasets, we compared GenomeFace with VAMB and MetaBAT 2. The contamination and completeness of each genome bin produced by these binners were evaluated using CheckM2 [11]. Taxonomic assignment was conducted using GTDBTKv2 [12], with the database version being 214.0. All binners were run with a minimum output bin size of 200 kilobases. Semibin was excluded from this comparison because it failed to run.

In Figure 2(c), we compare the number of medium quality genomes produced by each binner, as defined by the Minimum Information about a Metagenome-Assembled Genome (MIMAG) standards, which specify $\geq$50% estimated completeness and $\leq$10% estimated contamination. Figure 2(a) presents a comparison of the number of near-complete genomes generated by each binner, defined as having $\geq$90% estimated completeness and $\leq$5% estimated contamination. This criterion corresponds to the MIMAG standard for high-quality genomes but omits checks for RNA gene presence [13].

Case Study 1: Twitchell Island, California Wetlands

In Case Study 1, we present our findings from the 2.6 terabase coassembly of the brackish wetlands of Twitchell Island, located in the Sacramento Delta, California [14].

Utilizing MetaHipMer2 (MHM2), a distributed metagenome assembler [5], we co-assembled 2.6 terabytes of FASTQ reads from 21 samples collected across seven different locations with varying salinity and conditions [15]. This process resulted in 71.3 million scaffolds, each at least 500 bp, totaling 71.7 Gbp.

For binning with MetaBAT2, the 2.9 million scaffolds that were at least 2.5 kbp in length were used (subsequently, scaffolds of $\geq$ 1000 bp were attempted to be integrated post-binning), producing 3,901 bins larger than 200 kilobases, of which 1,208 were of medium quality or better. Employing VAMB with its manuscript default of sequences $\geq$ 2000 bp in length resulted in 1,240 medium-quality metagenome-assembled genomes (MAGs). GenomeFace, tested with its default setting for scaffolds greater than or equal to 1500 bp in length, generated 1,923 medium-quality genomes, distributed across 69 phyla and candidate phyla.

Case Study 2: Soil at Harvard Forest (Barre Woods Site)

To demonstrate our method's efficacy in characterizing microbes within a complex soil environment, we coassembled 28 metagenome samples, totaling 1.3 terabase pairs (Tbp), sequenced at the DOE Joint Genome Institute. This sequencing was part of a proposal titled "Molecular mechanisms underlying changes in the temperature-sensitive respiration response of forest soils to long-term experimental warming" [16]. The study aimed to elucidate the impact of climate change on the metabolic activities of microbial communities involved in the decomposition of soil organic matter (SOM) at the Barre Woods Harvard Forest in Petersham, MA, USA, a site of Long-Term Ecological Research (LTER) [17].

Coassembly using MetaHipMer2 (MHM2) yielded 70.8 million scaffolds, each greater than 500 bp, cumulatively amounting to 73.03 Gbp of assembled sequence.

For binning, MetaBAT 2 was employed with its default contig length settings, consistent with the approach used in Case Study 1. Similarly, VAMB was run with its manuscript default settings, as in Case Study 1. In contrast, GenomeFace was run with a minimum contig length of 2000 bp due to GPU memory requirements. Metagenome binning with MetaBAT2 resulted in (Richard) x metagenome-assembled genomes (MAGs) spanning (Richard) x distinct phyla. In comparison, GenomeFace generated (Richard) x MAGs from (Richard) x distinct phyla, which included (Richard) mention candidate phyla not found in MetaBAT2 bins, any other interesting observations.

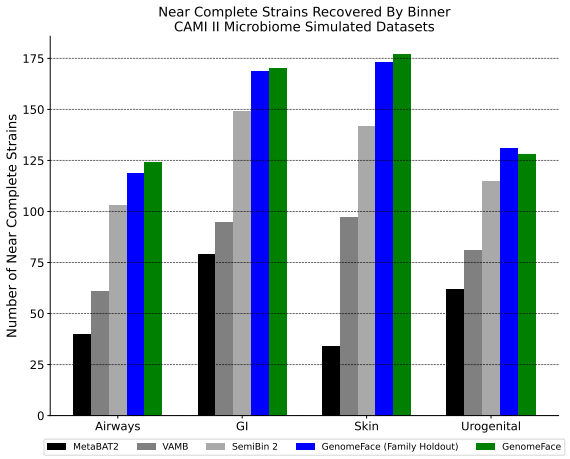
Case Study 3: Tropical Forest Soil

In this case study, we focus on the analysis of a substantial dataset derived from the Luquillo Experimental Forest (LEF) in Puerto Rico, a renowned site within the Long Term Ecological Research Network, as first detailed in [7]. This dataset comprises an impressive 3.4 terabase pairs (Tbp) of metagenomic sequence data obtained from 95 metagenome samples.

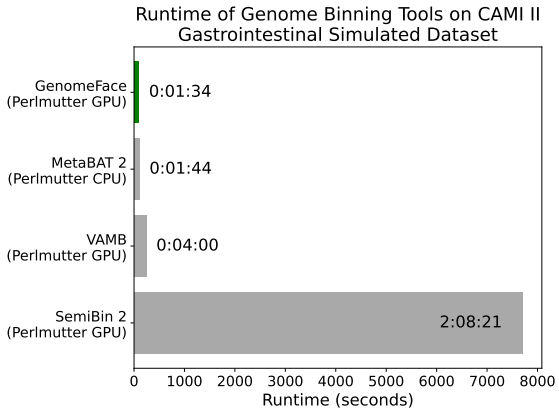
Using 512 nodes on the Oak Ridge National Laboratory (ORNL) Summit supercomputer and taking approximately 1 hour and 24 minutes, the data were assembled into 55,342,847 scaffolds, each at least 500 bp in length.

Upon initially inspecting the results from this study, we conducted an analysis to understand the unexpectedly low number of genomes relative to the number of single-copy genes observed in the assembly. We noted that the read mappings exhibited significantly higher inter-sequence variance than expected, coupled with substantial read duplication, likely a result of PCR amplification. To mitigate this variance, we employed samtools to remove duplicate reads.

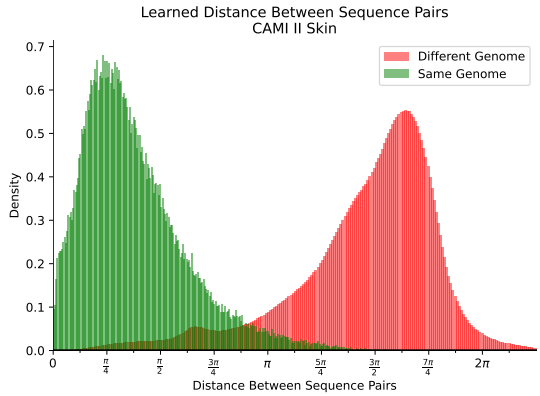
2.2 Simulated Datasets



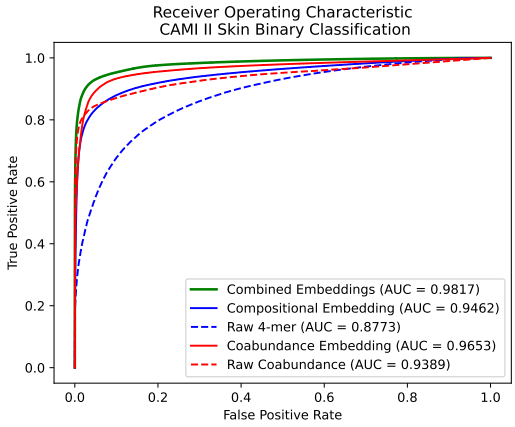
(d) Number Of Near Complete Strains Recovered By Binner



(e) Binner Runtime Comparison



(f) Graph Displaying Emperical Probability distributions of composite distance between sequence pairs, conditioned on belonging to the same genome (green) and different genomes (red).



(g) Receiver Operating Characteristic Curve Comparing Raw Input Modalities to their embeddings, using distances as thresholds for Binary Classification.

Fig. 2: Analysis of Simulated Datasets

In order to objectively measure the capabilities of GenomeFace, we utilized a set of simulated datasets from the Critical Assessment of Metagenomic Interpretation (CAMI) challenge [18]. CAMI's synthetic metagenomic datasets are previously unused in benchmarking VAMB [3] and SemiBin2 [19].

We compared GenomeFace's performance against VAMB, MetaBAT 2, and the relatively new SemiBin2. We quantified binner performance by the number of Near-Complete (NC) genomes reconstructed with a benchmark of $\geq$90% recall and $\geq$95% precision. Measurements were performed using VAMB's command line benchmark tool, which follows the same methodology described in AMBER [20], used in the CAMI challenge [18].

A distinctive feature of GenomeFace is its use of supervised training, integrating prior knowledge into the binning process. This introduces questions about its ability to generalize across unseen taxonomic groups. To address this, we adopted a "Family Holdout" cross-validation approach.

For each of the four simulated datasets, we identified all included genomes. Subsequently, we removed any genome from the training set that was taxonomically related to any genome within the simulated datasets, based on classifications from the Genome Taxonomy Database. For example, if the airways dataset included a bacterium from the Pseudomonadaceae family, all genomes from that family would be

excluded from the training set, decreasing the dataset size from around 43,000 to approximately 30,000. This exclusion strategy was executed with two distinct objectives: firstly, to verify that GenomeFace's learning mechanism is truly understanding and applying a generalized metric for distinguishing taxa, rather than simply memorizing the training set; and secondly, to ensure that GenomeFace's proficiency extends to categorizing taxa without any prior training on that family, showcasing its capacity to identify and classify entirely unfamiliar genomic structures.

Binning Performance

Across all four datasets, GenomeFace consistently demonstrated superior performance, achieving the highest number of Near Complete genomes (Figure 2(d)), performing almost identically under the Family Holdout constraint. Such results are indicative not only of GenomeFace's precision but also of its resilience and adaptability when presented with unfamiliar taxa.

Efficiency

In all four smaller 'toy' datasets, GenomeFace demonstrated competitive runtimes in comparison with VAMB and MetaBAT 2. Figure 2(e) presents a comparison of the runtime of these four binners on the CAMI Gastrointestinal metagenome. The binners were benchmarked on the Perlmutter supercomputer. VAMB, Semibin 2, and GenomeFace were run on Perlmutter GPU Nodes, equipped with an AMD EPYC 7763 CPU, four NVIDIA A100 GPUs, and 256 GB of DDR4 memory. MetaBat 2, which does not utilize GPU acceleration, was allocated Perlmutter CPU nodes that feature two AMD EPYC 7763 CPUs and 512 GB of DDR4 memory, but lack GPUs.

Binner	Methodology	Requires Training	Uses Marker Genes	Approx No. Params (Assume 10 Samples)
MetaBAT 2	Analytically Modeled Similarity Measure	No	No	18
VAMB	Unsupervised Embedding (Variational Autoencoder)	Yes	No	330,000
SemiBin 2	Self-Supervised Metric Learning	Yes*	Iterative Brute-Force Reclustering	370,000
GenomeFace	Large Scale Supervised Metric Learning	No	Dynamic Programming-based Selection from Hierarchy	100,000,000

Table 1: Methodology Comparison of Binners

3 Discussion

Key Findings

Rare Biosphere

Microbial communities, though often dominated by a few prevalent species, are also typically characterized by a 'long-tail' of low-abundance organisms. Known as the rare biosphere, these organisms compose the vast majority of microbial diversity [21–23].

Despite their limited numbers, these species commonly exert disproportionate influence on their ecosystems. Often they perform specialized roles, such as autotrophic nitrifiers, or those with niche substrate preferences, such as methanotrophs and methylotrophs. Low abundance species can also play key roles, as exemplified by giant sulfur bacteria; the Beggiatoa [24], Thioploca [25], and Thiomargarita [26] genera, while small in number, hold substantial biomass and contribute significantly to biomass and sulfur cycling in sediment due to their large-diameter cells [27]. Other rare biosphere keystone species' presence in sediment has been shown to influence plant biomass and defensive mechanisms, having downstream effect on herbivore interactions and broader ecological dynamics [28].

Large Scale Coassembly: An Inevitable Evolution

Gaining insights into the rare biosphere has proven elusive, causing these scarcely understood taxa to sometimes been referred to 'microbial dark matter' [21]. Genome assembly in metagenomic studies has been achieved predominantly through multiassembly, a technique wherein each sample is assembled individually. While multiassembly is less computationally demanding because it processes only a subset of the data at once, this also inherently limits the effective sequencing coverage of genomes, leading to insufficient sequencing coverage for low abundance species. Multiple metagenomic studies have observed multiassembly resulting in more fragmented and incomplete Metagenome-Assembled Genomes (MAGs) displaying less diversity compared to coassembling the same data [10**?**].

This limitation is not merely anecdotal; The Critical Assessment of Metagenome Interpretation (CAMI) II challenge found in a controlled benchmark, every assembler tested required a minimum of 10× sequencing coverage is necessary to achieve 90% recall of assembled genomes [18]. This implies that even with multiple rounds of multiassembly, the genomes of only the most abundant community members are likely to be constructed (and reconstructed), thereby disregarding the rare biosphere.

The Challenge of Binning After Coassembly

While large scale coassembly offers a more comprehensive view of microbial communities, this inherently intensifies the binning process. The produced assemblies are not only larger and more challenging computationally, but are also more entropically complex.

Metagenome binning, especially on datasets derived from complex environmental microbial communities, is still an unreliable process. By developing deep neural networks trained on a large corpus of high-quality reference genomes to capture sequence composition information, exploiting sample statistics to avoid sample inter-dependency, employing universal prokaryotic marker genes to guide contig clustering, and leveraging multi-GPU acceleration to enable efficient binning on extremely large datasets, we showed that GenomeFace significantly improves genome binning over existing software on several synthetic and real-world datasets, and in some cases, by a large margin.

[Tie in and elaborate in impact, as described in introduction]

4 Online Methods

4.1 Overview

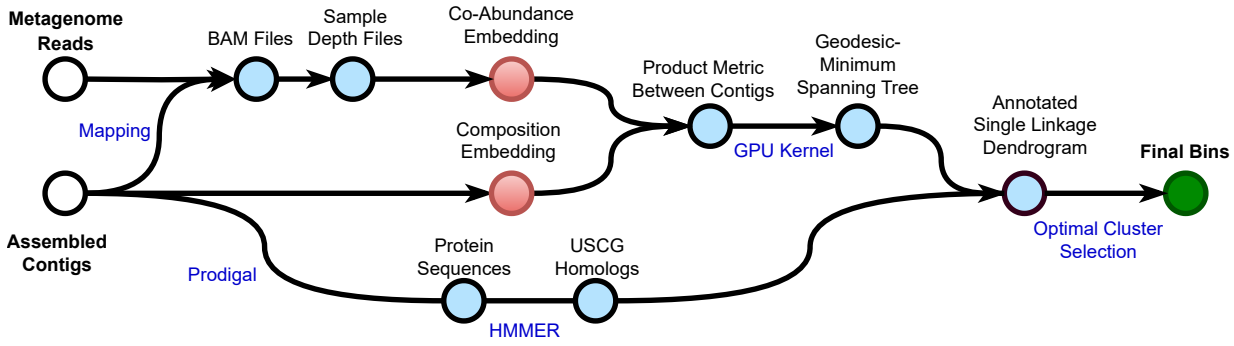


Fig. 3: Overview of GenomeFace

To cluster assembled sequences GenomeFace uses, the composition of the sequences themselves along with the [read mapped abundance]. Each sequence is passed through a modality specific neural network that outputs an embedding optimized to maximize discriminative ability. Let x_n represent the output of

the composition based embedding for the nth sequence, and y_n for co-abundance based embedding. The distance between two sequences $\mathbf{C}_i, \mathbf{C}_j$ is then defined based on their embeddings as

$$d(\mathbf{C}_i, \mathbf{C}_j) = \sqrt{\alpha \cdot \arccos^2(x_i^T x_j) + (1-\alpha)\arccos^2(y_i^T y_j)}$$

where $\alpha \in [0, 1]$ is a mixing coeffient to weight the contribution of each modality, that (by default) is determined by the number of environmental samples.

We consider each sequence a vertex in a graph, and find the minimum spanning tree of the complete graph where the edge weight between c_i and c_j is defined by the distance between them using our GPU accelerated kernel. We term this graph the 'geodesic minimum spanning tree' or G-MST.

From the G-MST of N sequences, a single linkage dendrogram based on the distances is built to form a hierarchy of $2N-1$ *candidate* clusters. Each leaf cluster $\mathbf{C}_n$ xx'is annotated using the Universal Single Copy Genes the sequence it represents codes. These genes are propagated upward in the dendrogram to obtain marker gene counts for every candidate cluster based on the sequences they contain, and bosed on their annotated genes. An abstract reward value is then calculated for each candidate cluster based on their gene counts, and an arrangement of candidate clusters, **Clusters**, is outputted to maximize $\sum_{\mathbf{C}_i \in \mathbf{Clusters}} R(\mathbf{C}_i)$ based on a cluster reward function R.

4.2 Compositional Neural Network

4.2.1 Pipelined Contig Sampling and Feature Generation

To train our compositional neural network genomes from the Genome Taxonomy Database were filtered to those with an estimated CheckM contamination of less 5%, and a minimum completion of 80%. The minimum completion requirement was implemented out of concern the network would incorrectly bias homologous regions of a genomes to a certain taxonomy. A limit of 25 genomes per genera was implemented, as to satisfy memory requirements and prevent severe taxonomic imbalances in our training set. This results in 39,006 genomes[fixme] for our primary training.

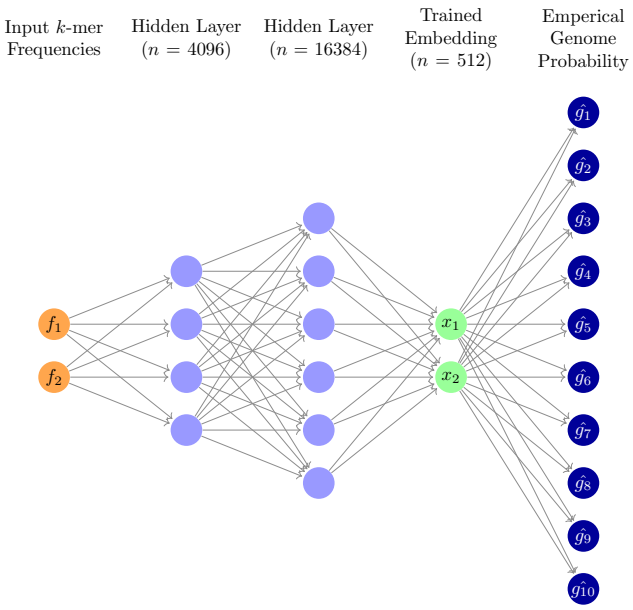
A native python module was developed as to be called by a Tensorflow Dataset Iterator. The python loads a collection of genomes into memory, and calls a sample method. When the sample method is called, a batch of 2^{18} genomes are chosen, with replacement, as to generate a training contig from. For each sampled reference genome, a scaffold is chose from those with probability proportional to the length of the scaffold. On this scaffold, a window of length 2000 is sampled uniformly to represent a contig. For these 2^{18} contigs, their 1-5-mers frequencies are counted, with reverse complements considered equivalent, such that for k-merse of even k, a feature vector of length $2^{k-1} + 2^{2k-1}$ is formed, and of length 2^{2k-1} for odd k-mers. For tractability, the degenerate R-Y alphabet where A,G$\mapsto$ R, and C,T $\mapsto$ Y, k-mers of length 6-10 are used with reverse compliments being considered equivalent, such that for even and odd length k, feature vectors of size $2^{k/2-1} + 2^{k-1}$ and 2^{k-1} were respectively produced. After counting, each respective kmer frequency vector was l_1 normalized.

We parallelize k-mer counting of each batch with multithreading and during the processing the Python Global Interpret Lock (GIL) is released. This is safe, as no Python managed memory allocations are accessed or made. After frequency counting is complete, the GIL is reacquired, the kmer frequencies are coerced into a tensorflow friendly input by using the NumPy C-API, and returned to Python with labels.

By instructing Tensorflow is instructed to prefetch batches, we are able to asynchronously sample and count kmers for sequences in for one batch utilizing the CPU, while simultaneously training the network using the GPUs in our system.

4.2.2 Network Architecture

The input to the composition neural network is the 1-5mer and 6-10RY-mers feature vectors, as described above. These are concatenated to form one logical vector feature vector. Since we sample contigs, it is non trivial to normalize the 'dataset' in its entirety, so a Batch Normalization layer is applied to this



concatenation. A dense layer with output size 4096 with hyperbolic tangent activation is applied, then another Batch Normalization layer. Again, another a dense layer with output size 16384 and a hyperbolic tangent activation follows, and another Batch Normalization Layer follows that.

Similarly to described in [insert paper here], we form our embedding by applying a Dense layer with output size 512. No (i.e. a "linear") activation is applied, followed by a batch normalization layer. The output of this batch normalization layer is then l_2 normalized, completing the base embedding network. For regularization, weights of every dense layer has a very mild l_2 penalty applied.

4.2.3 Training the Compositional Network

The popular softmax loss often comes into play in deep neural networks when addressing multi-class classification challenges. It employs the softmax activation function in the network's output layer and then proceeds with categorical cross-entropy training. In the following passage, we detail the inuition for is use in our embedding training.

In a network engineered for d-way categorical classification that employs softmax loss, the second to last layer produces an embedding of size n. The final layer then multiplies this embedding by a matrix W, which exists in $\mathbb{R}^{d \times n}$, consequently yielding a d-dimensional vector. Each vector component aligns with a class. The softmax activation guarantees that the vector components resulting are not only positive but also sum up to one, thus producing a valid predicted probability distribution.

We'll refer to the a-th row vector of the matrix W as W_a. Our network employs the following equation to estimate the probability of a contig pertaining to genome k:

$$\hat{g_k} = \frac{e^{W_k \vec{x}}}{\sum_{i=1}^{d} e^{W_i \vec{x}}},$$

where $\vec{x}$ is the sample's embedding from the second-to-last layer of the network.

The network is trained by minimizing the cross-entropy H between the predicted and ground truth probability distributions. Cross-entropy quantifies the minimum average number of bits required to represent events from distribution p using an optimal coding scheme designed for distribution q. It is defined as:

$$H(p, q) = -\mathbb{E}_p[\log_2 q] = -\sum_{i \in \mathcal{X}} p(i) \log_2 q(i).$$

Assume y represents the ground truth distribution, embodied as a one-hot vector, with k indicating the true class of the sample. We can express the softmax loss as follows:

$$\begin{aligned}\mathcal{L}_1 &= -\sum_{i=1}^{d} y_i \log_2(\hat{g}_i) \\ &= -\log_2 \frac{e^{W_k \vec{x}}}{\sum_{i=1}^{d} e^{W_i \vec{x}}}\end{aligned}$$

Previous studies noted that we can interpret the output from the second to last layer as distributing classes radially in a natural manner. Observe that for $y_i = (Wx)_i$, $y_i = ||\vec{x}||||\vec{W}_i|| \cos \angle_{\vec{x}}^{\vec{W}_i}$. In this context, the magnitude of the output vector may suggest the confidence level of the classification. However, it's the radial angle between the output vector and the weight vectors, in conjunction with their relative magnitudes, that naturally encodes the 'feature description'.

To further enhance the discriminative capacity of the features, we apply l_2 normalization to both the rows of W and each $\vec{x}$. This normalization ensures that their product, and hence the per-class probability, depends exclusively on the angle between them. Following this, we introduce a softmax scale (or inverse-temperature) denoted by s.

Under this construction, each component $(Wx)_i$ of the matrix-vector product $W\vec{x}$ can be interpreted as the cosine of the angle between the ith row of W and $\vec{x}$. From a heuristic viewpoint, we can consider the ith row of W as the "center" embedding that represents its corresponding class. With these modifications, the loss function can be reformulated as:

$$\mathcal{L} = -\log_2 \frac{e^{s \cdot \cos \theta_k}}{\sum_{i=1}^{d} e^{s \cos \theta_i}}$$

Minimizing the cross-entropy loss implies that, for each sample, the embedding is close to its class center (θ_k is small) and far away from the centers of other classes ($\theta_i, \quad i \neq k$ is large). To further enhance the model, a softmax scale (or inverse-temperature) s is applied. This increases the range of logits entering the softmax function, specifically the cosines. For large d, without the scale factor s, the max achievable probability would be limited to at most e/d in the ideal case.

While this captures the essence of our network, we integrate the modificaitons to softmax-crossentropy presented by CurricularFace[29] for curricular learning.

4.3 Co-Abundance Network

Like other metagenome binners, we integrate per sample read coverage into our binner. Disregarding sequencing bias, reads in single environmental sample from a genome can be seen as being drawn at uniform rate throughout the genome – thus the mean read coverage for a contig can be seen as a proxy measure for abundance of the organism from which it originates, with each sample providing additional information for distinguishing between genomes in the binning process. Viewed as a vector, the mean per sample read coverage could of course be clustered on directly. However, this is less than ideal. The uniform coverage rate assumption implies the distribution of coverage for per base pair in a genome is theoretically Poisson, Thus as read coverage increases for a genome, we would expect mean variance does also, indicating a constant differential in read coverage between two contigs should be viewed less significantly at a higher coverage than a lower one. In observation, the variance of the coverage is significantly greater what is expected of the Poisson distribution, due to factors such as sequencing biases and read mapping ambiguities.

To account for this, we train a second network to embed our coverages into a higher dimensional space. However, as the number of environmental samples used for a ecological study can vary from as few as 1 to more than 1000, the appropriate construction of a general network to do so is not immediately obvious. Additionally, since each sample depth coverage portrays the same type of information, there is some symmetry inherent to the problem.

Making clear this symmetry, we design our Co-abundance neural network to enforce the principle of permutation invariance in our output binning: the order of input of the environmental sample coverage does not affect the produced clustering. Before we describe the implementation, we will justify its properties.

The motivation for permutation invariance arises from the view of the samples as exchangeable from a Bayesian perspective. In statistics, a sequence of random variables (in our case, the samples) is considered exchangeable if its probability distribution remains unchanged when the variables are permuted. Although each sample may come from a unique source or biome, from the perspective of the binner, the order does not convey any semantical information that should influence the output. Therefore, for robustness, these samples should be treated as exchangeable, operating under the premise that any given order of samples could have occurred under the underlying probability distribution.

We denote our abundance embedding network by $\phi : \mathbb{R}^{|samples|} \to \mathbb{R}^m$. As clustering algorithms operate on distances, the idea of permutation invariance in the produced binning translates to an intriguing property for our neural network: for any two abundance vectors a_1, a_2, the distances in the embedding space should remain unchanged under any permutation of the samples. Formally, for every permutation σ_i, we have

$$d(\phi(a_1), \phi(a_2)) = d(\phi(\sigma_i a_1), \phi(\sigma_i a_2)).$$

This observation leads us to a key conclusion: for every permutation σ_i, we can find a corresponding transformation $U_i : \mathbb{R}^m \to \mathbb{R}^m$ such that $U_i \circ \phi(a) = \phi(\sigma_i a)$. The transformations U_i form a group that is isomorphic to the group of permutations. Specifically, the group structure is induced by each U_i being an isometry, i.e., it preserves distances in $\mathbb{R}^m$. This characterizes our neural network ϕ as an equivariant map, and that it should "pass" the permutation group through it, and enforcing a factorial way symmetry (i.e. |sample|!) on it's input. To enforce this on the symmetry, we use a permutationally equivariant neural network.

One might naturally question the benefit of a permutationally equivariant neural network, as opposed to just mapping each sample to a higher-dimensional space independently and then concatenating the results. Crucially, our approach recognizes that the sequencing depths across different samples are not independent - they exhibit systematic biases due to sequencing techniques and factors like sequence composition and position within the genome. Moreover, our model allows for a representation that avoids the naive averaging, instead allowing for a geometry which could more accurately describes the joint likelihood of the differential abundances across multiple environmental samples.

4.3.1 Co-Abundance Network Architecture

4.4 Balancing Embeddings

$$d(c_i, c_j) = c_1 \arccos^2(\langle x_i, x_j \rangle) + c_2 \arccos^2(\langle y_i, y_j \rangle)$$

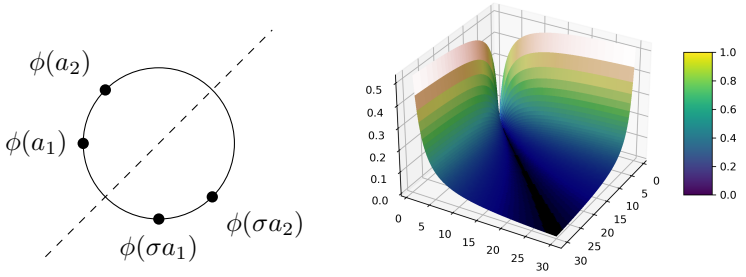
where c_1, c_2 are mixing coeffient determined by the number of samples. See supplement for full derivation.

We model the weight placed on each modality should be inversely proportionally to the inter-class variance of the modality. We assuming the variance of the compositional embedding, and therefore c_1, should be constant with regards to number of environmental samples n.

For the Coabundance embedding, we modeled the a variance component (v_1) term which should be inversely proportional to the number of samples, as though they were independent, however, considering systemic biases (such as read mapping ambiguities caused by repeated regions) which would be constant (v_2). Therefore, we arrived that c_2, reciprocal to the variance, would be

$$c_2(n) = \frac{1}{v_2 + \frac{v_1}{n}}.$$

To determine the constants v_1, v_2 we took the follow steps:



(a) Illustration of Permutation Isometry

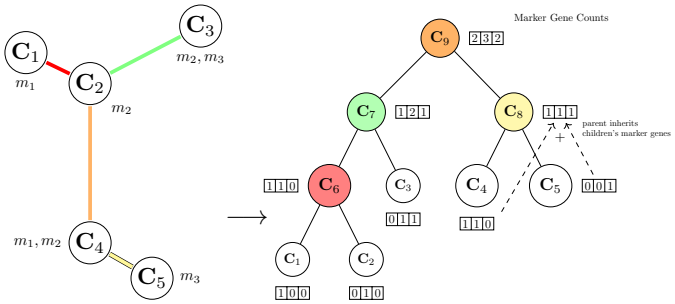
(b)

Fig. 4: (a) Illustration of Permutation Isometry: Points $\phi(a_1)$ and $\phi(a_2)$ on a unit circle, and their symmetric counterparts $\phi(\sigma a_1)$ and $\phi(\sigma a_2)$ obtained through reflection across the dashed line. The isometric property of permutations is demonstrated, where distances between corresponding points remain unchanged under any permutation. (b) Surface plot illustrating the relationship between input values and the distances between embeddings in the actual trained neural network. Each point on the plot represents a pair of contigs from a theoretical single sample assembly. The parabolic shape captures the impact of read coverage values for each of the contigs (x and y axes) on the resulting distances (z axis) in the embedding space. As coverage increases, the curve gradually becomes less steep, reflecting the decreasing impact of incremental changes in coverage at higher levels, consistent with the characteristics of the Poisson distribution. This visualization demonstrates how the network approximates an implicit statistical distance, or pseudo-likelihood that contigs originate from the same genome while accounting for the underlying variability.

1. We simplified the dissimilarity formula by setting c_1 to a constant value of 1. This adjustment is based on the understanding that scaling both c_1 and c_2 by the same factor only scales the output scores by a constant, leaving the AUROC unchanged.
2. We considered the effect of using various number of environmental samples n on the output embedding with using the CAMI II Airways dataset by only considering the first n samples, therafter removing sequences with no coverage after modification. By inspection, it was determined the AUROC of the composite distance as a function of c_2 was a unimodal function. Thus, for each number of samples n, we used golden section search to find the value of c_2 that maximized the AUROC for the restricted dataset. This resulted in a series of pairs $(n, c_2(n))$ correlating each sample size with its corresponding optimal c_2 value.
3. Finally, we used the pairs of $(n, c_2(n))$ as data points for curve fitting. Applying Scikit-Learn's curve_fit tool, we fit the function $c_2(n)$ defined above, allowing us to fit v_1, v_2 to the observed pairs, thereby allowing us to predict the optimal $c_2(n)$ for any number of samples n.

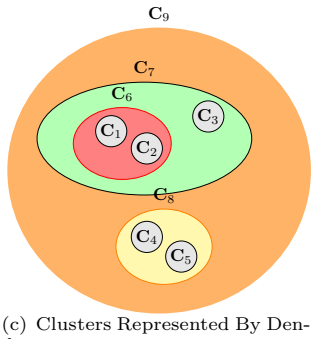
After, to standardize the range of our dissimilarity metric to $[0, 2\pi^2]$ we scale c_1 and c_2 such that $c_1 + c_2 = 2$.

4.5 Clustering



(a) Geodesic Minimum Spanning Tree

(b) Single Linkage Dendrogram



(c) Clusters Represented By Dendrogram

Cluster $\mathbf{C}_i$	Reward $R(\mathbf{C}_i)$
$\mathbf{C}_1$	0.06
$\mathbf{C}_2$	0.06
$\mathbf{C}_3$	0.41
$\mathbf{C}_4$	0.41
$\mathbf{C}_5$	0.06
$\mathbf{C}_6$	0.41
$\mathbf{C}_7$	0.00
$\mathbf{C}_8$	1.00
$\mathbf{C}_9$	0.00

(d) Candidate Cluster Reward Values

Our clustering algorithm leverages domain-specific knowledge to achieve more accurate and meaningful cluster assignments. Universal Single Copy Marker Genes (USCG) can be viewed as weak pairwise labels. Two occurrences of the same USCG imply that the contigs coding for them should be in different clusters, while we know that each cluster should ideally include one of each gene. By incorporating information from universal single-copy marker genes, our method can make more informed decisions when merging or separating clusters compared to generic clustering approaches that rely purely on distance or density estimates alone. Our tailored approach to clustering provides a distinct advantage over traditional, non-task-specific clustering algorithms and ensemble binners, allowing for improved overall quality of the resulting genomic bins.

The algorithm operates by first building a Geodesic Minimum Spanning Tree (G-MST) from the contigs' embeddings. It then constructs a Single Linkage Cluster Dendrogram from the G-MST, which serves as a hierarchical representation of candidate clusters. The cluster represented by each parent node is the merger of the clusters represented by its child nodes, where the leaf nodes represent clusters defined by a single contig. This hierarchical representation allows us to efficiently incorporate marker gene information without the need to reprocess contigs or marker genes for every candidate cluster considered. By simply adding the marker gene counts of child nodes in the dendrogram to obtain the marker gene counts of each parent node, we can quickly estimate the purity and completeness of each candidate.

Additionally, this hierarchical structure presents a topological ordering on the candidate clusters. This topological ordering enables us to find the optimal set of clusters from the candidate clusters in near linear time by using dynamic programming. In contrast, ensemble binners face a far more challenging task—an NP-complete problem—as finding the optimal set of clusters for an arbitrary set of candidate clusters essentially involves solving an optimal weighted set packing problem, where the weight for each set is the cluster's reward value.

Our algorithm can be seen as examining balls in the embedding space surrounding each contig's embeddings, which form a cover for the contigs' embeddings. Contigs clusters are naturally defined by the connected components of this cover. Nodes in the dendrogram represent thresholds such that the connected components have changed. As we navigate the dendrogram, the balls either expand or contract based on the edge weights of the dendrogram, as induced by the Euclidean Minimum Spanning Tree.

By integrating marker gene information directly into the clustering process, our algorithm learns in a transductive or semi-supervised manner from the data. The marker genes serve as weak pairwise labels, supplying information about the relationships between contigs' classes. This information guides the decision-making process for growing or shrinking the balls in the embedding space. If the presence of marker genes signals potential contamination in a cluster, the algorithm may opt to shrink the balls

around the contigs that cluster as to fracture it and lower contamination; on the other hand, if the marker gene information suggests that merging clusters would improve completeness, the algorithm may choose to grow the balls surrounding the pertinent contigs. We actively learn a non parametric decision boundary for inclusion for each cluster – not only utilizing those contigs with marker genes, but also those without. For further intuition, refer to HDBSCAN's [30] paper, from which our algorithm was based.

Formally, given the clusters represented by taking the nodes of the Single Linkage Dendrogram, and R defined to be a function that outputs a real valued rewards a cluster based on it's marker gene contents, we optimize to find a set of clusters from the hierarchy **Clusters**

$$\sum_{\mathbf{C}_i \in \mathbf{Clusters}} R(\mathbf{C}_i)$$

subject to the constraint that **Clusters** is a partition of the contigs – every contig exists in some cluster, and no contig is in two clusters.

We give an overview of the algorithm, detail our chosen cluster reward function, and then give the algorithm in detail.

4.5.1 Overview of the Clustering Algorithm

The main steps of the clustering algorithm are as follows:

1. Build the Geodesic Minimum Spanning Tree (G-MST) from the contigs' embeddings.
2. Construct a Single Linkage Cluster Dendrogram from the G-MST and initialize a marker gene count for each node in the dendrogram.
3. Enumerate and process marker genes for each contig.
4. Traverse the dendrogram in reverse topological order, propagating marker genes counts upwards, and computing the reward for each node.
5. Traverse the dendrogram in topological order to extract the final set of clusters, choosing the optimal reward clusters.

4.5.2 Cluster Reward Function

Following the premise that universal single-copy genes (USCGs) should occur in every genome and be single-copy, we use the frequency of their occurrence (or over-occurrence) to measure the completion and contamination of a cluster.

In our tests, we used the set of 40 USCGs from Creevey et al [31]. that were derived from Bacteria, Archaea, and Eukaryotes for their generality and were evaluated to be of higher quality in terms of measured universality and uniqueness compared to other gene sets[32]. Similarly to CheckM1's methodology of grading the first marker gene as correct and viewing any later occurrences as contamination[33], we define the following heuristics to estimate Purity and F_1 score. Let M be our set of universal single-copy marker genes, and for $m \in M$, define $\text{Count}_m(\mathbf{C}_i)$ to be the number of times the marker gene m occurs in some cluster $\mathbf{C}_i$. Then

$$TP(\mathbf{C}_i) = \sum_{m \in M} \min\{1,\ \text{Count}_m(\mathbf{C}_i)\} \qquad \text{True Positive}$$

$$FP(\mathbf{C}_i) = \sum_{m \in M} \max\{0,\ \text{Count}_m(\mathbf{C}_i) - 1\} \qquad \text{False Positive}$$

$$FN(\mathbf{C}_i) = |M| - TP(\mathbf{C}_i) \qquad \text{False Negative}$$

$$P(\mathbf{C}_i) = \frac{TP(\mathbf{C}_i)}{TP(\mathbf{C}_i) + FP(\mathbf{C}_i)} \qquad \text{Purity}$$

$$F_1(\mathbf{C}_i) = \frac{2TP(\mathbf{C}_i)}{2TP(\mathbf{C}_i) + FP(\mathbf{C}_i) + FN(\mathbf{C}_i)} \qquad F_1 \text{ Score}$$

While any cluster reward function may be chosen, we choose the reward function

$$R(\mathbf{C}_i) = \mathbb{1}[P(\mathbf{C}_i) \geq 0.85] \times F_1(\mathbf{C}_i)^4,$$

as a good trade-off between valuing purity and completeness. We use the F_1 score rather than the F_β with a lower β to be more tolerant of genomes with mutations where a small number of genes in our set become non-single copy, but compensate with a default minimum 85% estimated purity for any reward (modifiable as a tuning parameter). Since the raw F_1 score is subadditive with respect to the merger of two bins, even if there was no apparent contamination, this would result in multiple small incomplete bins being preferred to a larger completely pure one. To negate this, we map the F_1 score through a sufficiently convex function, and arbitrarily choose to raise it to the fourth power.

Geodesic Minimum Spanning Tree Construction

We modify NVIDIA's [34] HDBSCAN implementation for euclidean minimum spanning tree construction which finds the k-nearest nearbors using FAISS's k-select [35] fused with a matrix multiply kernel to calculate the euclidean k-nearest neighbors, thereafter finding the Euclidean-Minimum Spanning Tree [36].

NVIDIA's implementation to find the euclidean distance by combining with magnitude of input vectors with their respective inner products. It can be conceptualized as loading the vectors, denoted v_n as the rows matrix V, and finding VV^T, then calculating $d(v_i, v_j) = \sqrt{||v_i||^2 + ||v_j||^2 - 2(VV^T)_{i,j}}$. (This is done interspersed with reductions with the k-select, the actual calculation of (VV^T) in it's entirety would be prohibitively large.)

We instead use two matrices, denote X and Y, one for each modality, such that the nth row of each represents the embedding of the nth sequence for that modality. The two matrix multiplies are performed sequentially the first pipelined into the second, then the k-select is applied in a similar manner as previously, using the two matrix multiplies to calculate the distance as $d(C_i, C_j) = \alpha \arccos^2((XX^T)_{i,j}) + (1-\alpha)\arccos^2((YY^T)_{i,j})$ for some mixing cosntant α.

The kernel was also modified to utilize all available GPUs when finding the k-nearest neighbors.

Efficient Single Linkage Dendrogram Construction

To ensure clarity, we detail the process of constructing the Single Linkage Dendrogram using the weight-sorted edges of the Geodesic Minimum Spanning Tree (G-MST). This construction can be efficiently achieved in $N\alpha(N)$ time using a modified union-find data structure, where N represents the number of sequences to be clustered, and α is the inverse Ackermann function, which is effectively constant for all practical purposes.

We introduce a modified version of the union-find data structure. This enhanced structure allows each set to reference a vertex in our dendrogram. For a given set s, this reference field is denoted as s.ref.

The initial traversal of the dendrogram following its construction can be omitted. Since the dendrogram is built in reverse topological order relative to the root, the steps usually performed during the first traversal can be integrated into the construction phase itself. However, for simplicity, we describe these as separate steps in our presentation.

Structure DendrogramNode		
extract:	Boolean	▷ Indicates this Cluster is part of Optimal Solution
BestSubReward:	Float	▷ Max Reward Clustering of node's subtree
markerCounts:	Array[Integer, $\|M\|$]	
leftChild:	DendrogramNode | Nil	
rightChild:	DendrogramNode | Nil	
sequenceID:	Integer | Nil	▷ Which Sequence Represented, if Node is Leaf

---

Algorithm 1 Efficient Single Linkage Dendrogram Construction.

Each leaf node in the dendrogram is initialized with a 'markerCounts' vector, where each index corresponds to a specific gene, and the value at that index indicates the count of occurrences of the gene in the associated sequence.

Require: A set of sequences and their Geodesic-MST with weight-sorted edges
Ensure: Single Linkage Dendrogram of the given sequences

▷ Initialize data structures

Initialize the empty dendrogram
Initialize modified union-find data structure UF

▷ Create initial dendrogram leaves

for each sequence seq **do**
    Initialize a representative leaf/node in the dendrogram v_{seq}
    Initialize a singleton set s_{seq} in UF to represent seq. Set s_{seq}.ref $= v_{seq}$
    Set v_{seq}.markerCounts to the count of marker genes in sequence seq
end for

▷ Construct the dendrogram

$v \leftarrow$ Nil
for each edge (src, dst, weight) in WeightSorted(G-MST) **do**
    $s_{\text{rep}} \leftarrow UF$.find(src)
    $d_{\text{rep}} \leftarrow UF$.find(dst)
    $v \leftarrow$ newDendrogramNode()
    v.leftChild $\leftarrow s_{\text{rep}}$.ref
    v.rightChild $\leftarrow d_{\text{rep}}$.ref
    UF.union($s_{\text{rep}}, d_{\text{rep}}$)
    rep $\leftarrow UF$.find(s_{rep})
    rep.ref $\leftarrow v$
end for
return v ▷ Return Root of the Dendrogram

Marker Gene Propagation and Cluster Optimization

In this step, we use dynamic programming to identify the optimal set of clusters, **BinOutput**, that given cluster reward function R maximizes the sum of rewards, $\sum_{\mathbf{C}_i \in \mathbf{BinOutput}} R(\mathbf{C}_i)$, under the constraint that **BinOutput** forms a partition of the input sequences, i.e. each sequence belongs in exactly one output cluster. Here, $\mathbf{C}_i$ represents nodes in the dendrogram, with clusters defined by the points representing their leaves.

This is achieved by solving the subproblem defined for a node $\mathbf{C}_i$, the maximum reward achievable considering only the clusters within its subtree for inclusion in **BinOutput**. The base case for leaf nodes is direct, where the reward is $R(\mathbf{C}_i)$. For non-leaf nodes, we compute the maximum reward by comparing the reward for including the node itself in **BinOutput**, $R(\mathbf{C}_i)$, against the sum of the optimal rewards from its child nodes' subtrees. This is because by our partition constraint, if some node $\mathbf{C}_i \in$ **BinOutput**, none of it's descendants in the dendrogram can be, as they represent members of refinements of $\mathbf{C}_i$. Thus, traversal is performed in reverse topological order, allowing us to calculate each node's $\mathbf{C}_i$.BestSubReward up to the root, which yields the maximum reward for the entire sequence set.

The optimal set **BinOutput** is then extracted in the next step by a forward traversal, selecting clusters marked for extraction using the extract field on the deprogram nodes, and collecting the sequences their leaves represent.

Note, when we set $\mathbf{C}_i$.extract as True, we do not remove the flag from their descendants. Doing so would increase the complexity. This is of no concern, as we may simply stop traversal of $\mathbf{C}_i$'s descendants during our next step.

Algorithm 2 Marker Gene Propagation and Reward Calculation

Require: Root of Dendrogram, *DendrogramRoot*

for each N in Reverse(BFS(*DendrogramRoot*)) **do**
  if N is a leaf **then**
    N.BestSubReward $\leftarrow$ R(N.markerCounts)
    N.extract $\leftarrow$ True
  else
    for i in range(0, NumMarkerGenes) **do**
      N.markerCounts[i] $\leftarrow$ N.leftChild.markerCounts[i] + N.rightChild.markerCounts[i]
    end for
    NReward $\leftarrow$ R(N.markerGeneCounts)
    Evaluate if the current cluster's reward is greater than the combined optimal rewards of its left and right subtrees.
    if NReward $\geq$ N.leftChild.BestSubReward + N.rightChild.BestSubReward **then**
      If the current cluster's reward is higher, update this node's best sub-reward and mark it for extraction.
      N.BestSubReward $\leftarrow$ NReward
      N.extract $\leftarrow$ True
    else
      If the combined rewards of the subtrees are greater, set this node's reward as their sum; The optimal clustering subproblem at this node is formed by merging the optimal clusters of sub-trees.
      N.BestSubReward $\leftarrow$ N.leftChild.BestSubReward + N.rightChild.BestSubReward
      N.extract $\leftarrow$ False
    end if
  end if
end for

Cluster Extraction

For completeness, we detail how to extract the clusters from the dendrogram after Cluster optimization is performed.

Algorithm 3 Cluster Extraction

Require: Root of Dendrogram, *DendrogramRoot*
Ensure: Optimal set of clusters, **BinOutput**

```
Initialize an empty list for BinOutput
function ExtractClusters(N)
    if N.extract is True then
        Initialize an empty list bin
        for L in Leaves(N) do
            Add L.sequenceID to bin
        end for
        Add bin to BinOutput
    else
        ExtractClusters(N.leftChild)
        ExtractClusters(N.rightChild)
    end if
end function
ExtractClusters(DendrogramRoot)
return BinOutput
```

4.6 Evaluation

4.7 Synthetic Dataset Evaluation

For the evaluation of the CAMI Human Microbiome datasets, we recreated similar benchmarks to those in VAMB and Semibin 2's papers. The reference, or 'gold-standard', assemblies and simulated reads for each sample were acquired from the CAMI challenge resources [**?**]. We combined the gold-standard assemblies from each collection site into a single fasta 'multi-assembly' file. Subsequently, the simulated read sets corresponding to these assemblies were aligned using BWA-MEM [37]. This alignment process yielded BAM files. Coverage profiles in a format compatible with MetaBAT 2 were generated using CoverM, and these profiles were used as input for MetaBAT 2, VAMB, and GenomeFace. Semibin 2 was provided with the raw BAM files directly, as it is incompatible with the MetaBAT 2 coverage profile format. We used VAMB's benchmarking tool in our evaluation, which measures the purity and completeness of each genome bin in relation to the number of base pairs in the gold-standard assembly. This tool adopts the same evaluation methodology as the AMBER benchmarker [20], which was used in the CAMI challenge. Since the multiassembly produced from Oral site samples were used used to train the co-abundance network, it was excluded from evaluation.

Declarations

- Authors' contributions
  - Richard Lettich and Katherine Yelick conceived the study.
  - Richard Lettich designed the Clustering Algorithim, Trained The Neural Network, Programmed the Kernel and GenomeFace program.
  - (not complete)
  - Richard Lettich, Robert Riley, Zhong Wang wrote the paper
  - All authors participated in planning and analysis of real dataset Case studies, reviewed the manuscript, and approved the final version. (not true for work in progress) manuscript

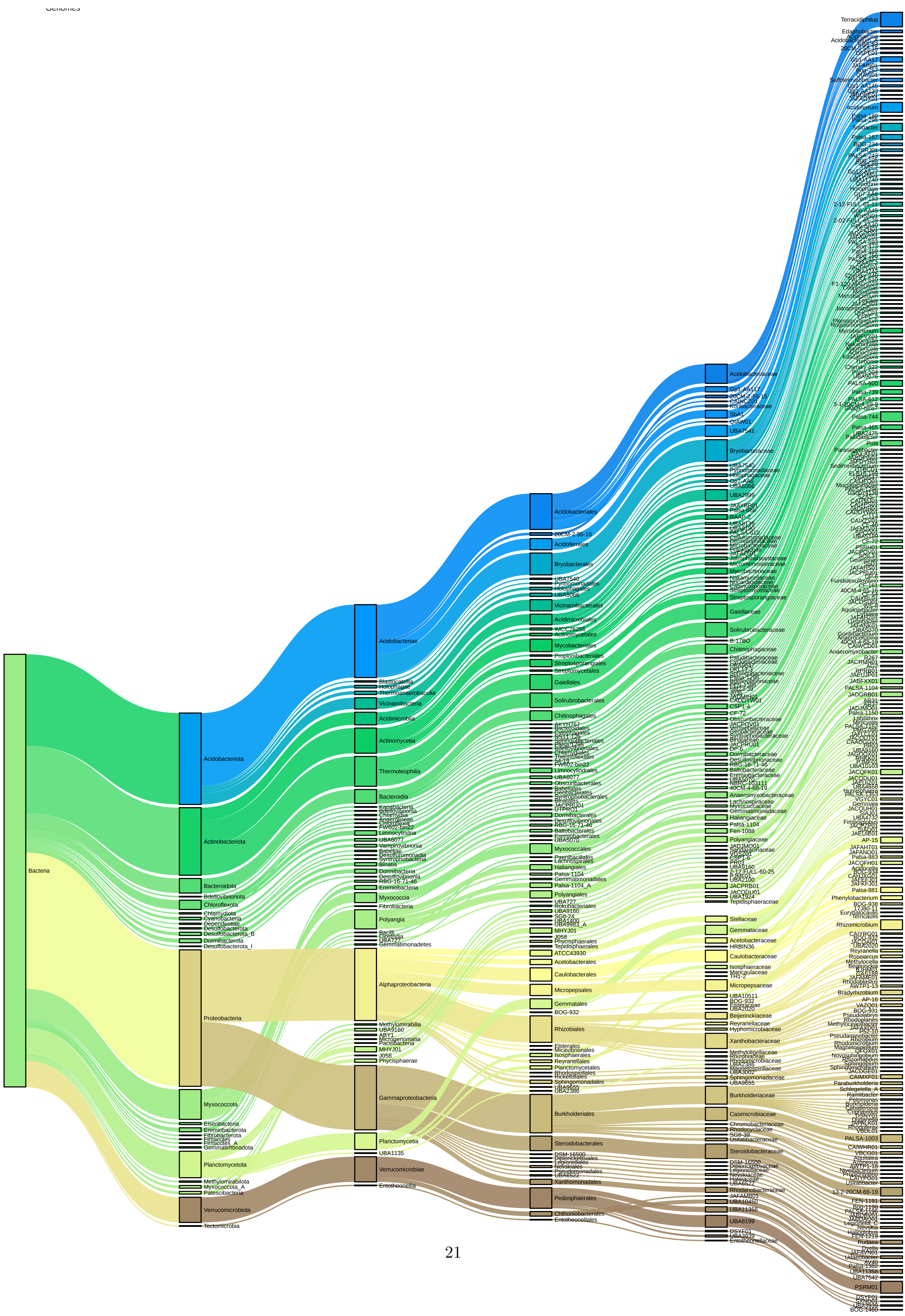


Fig. 1: Taxonomic Decomposition of GRE Bins

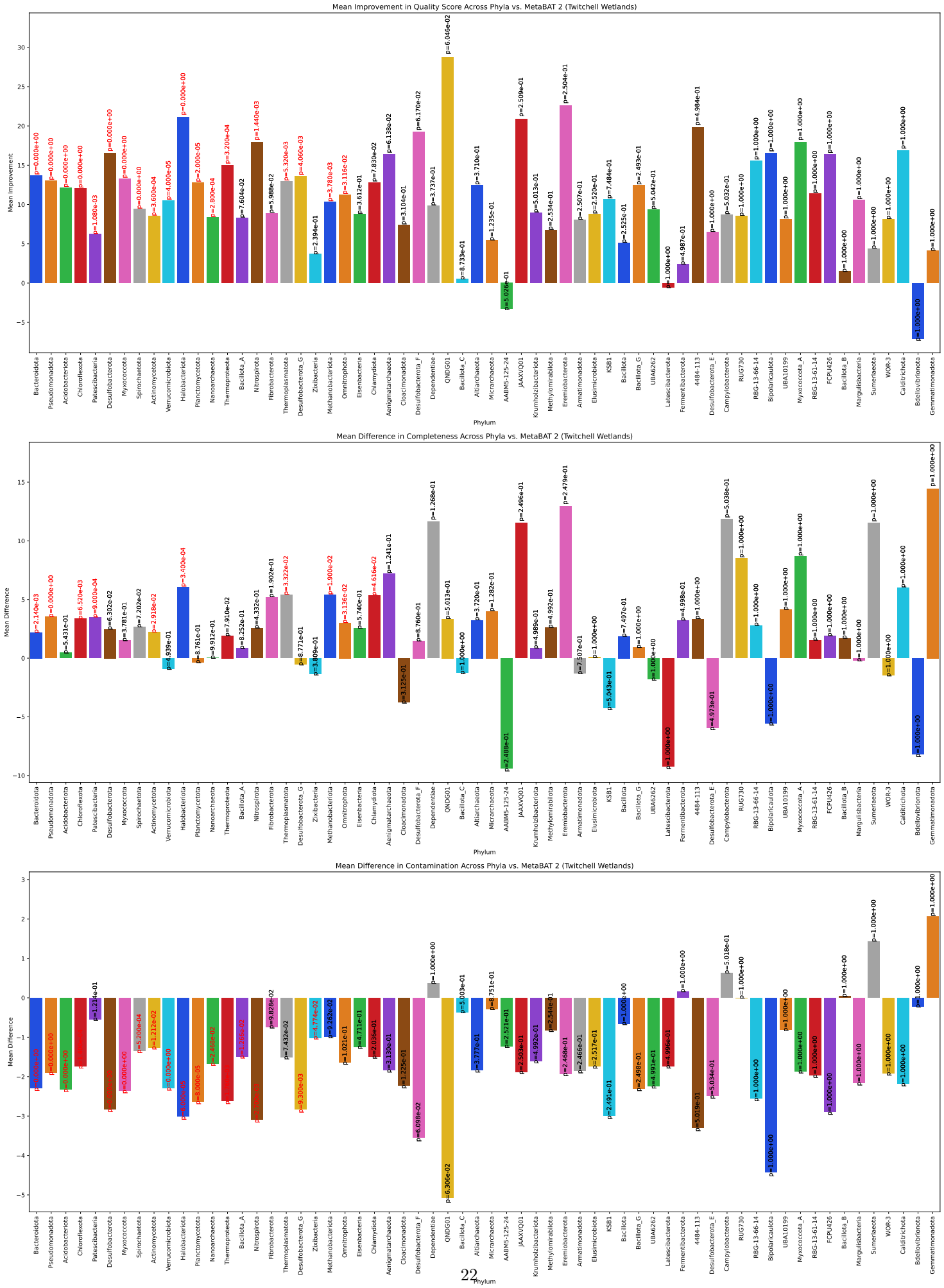


Fig. 2: Caption

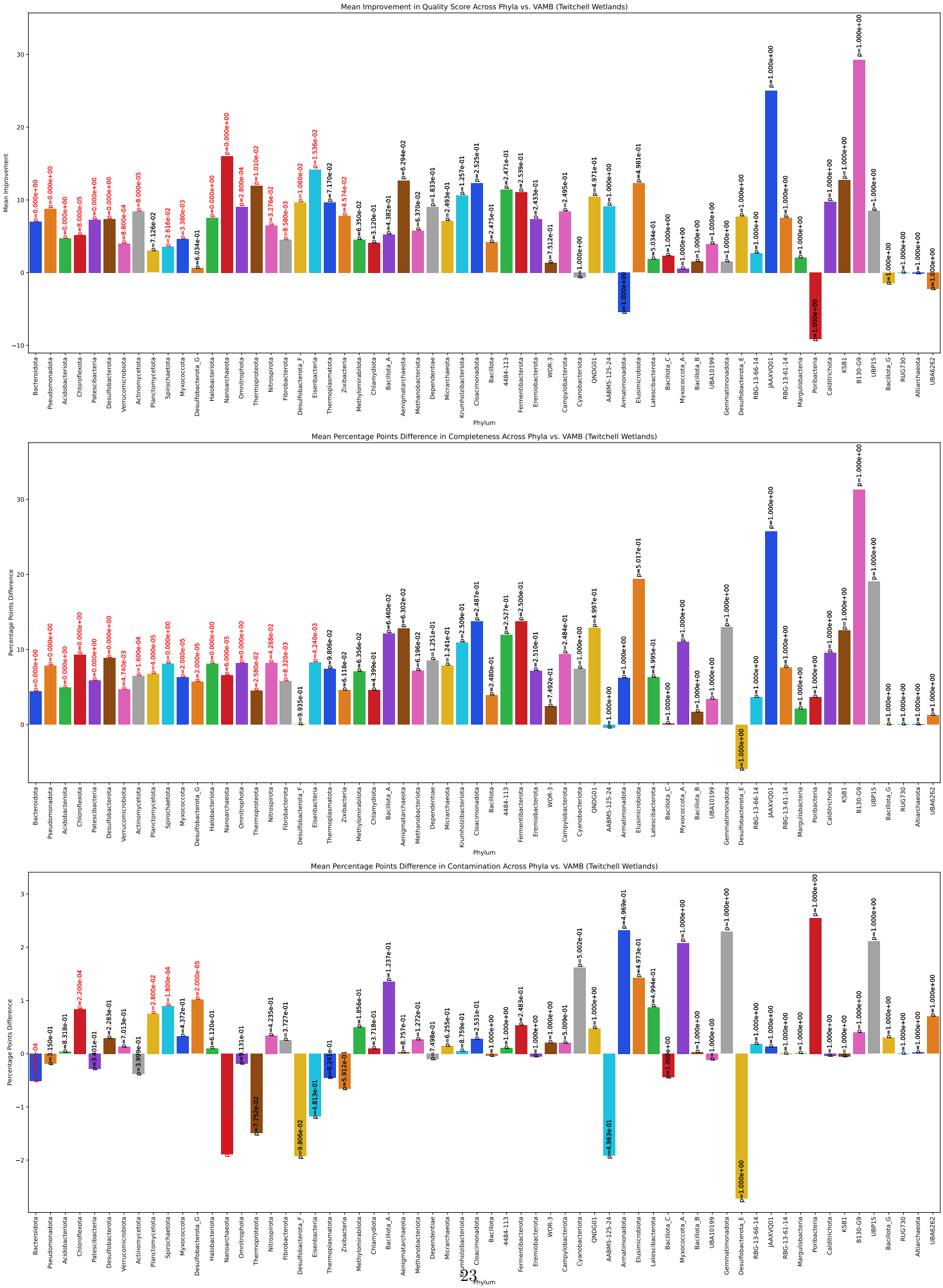
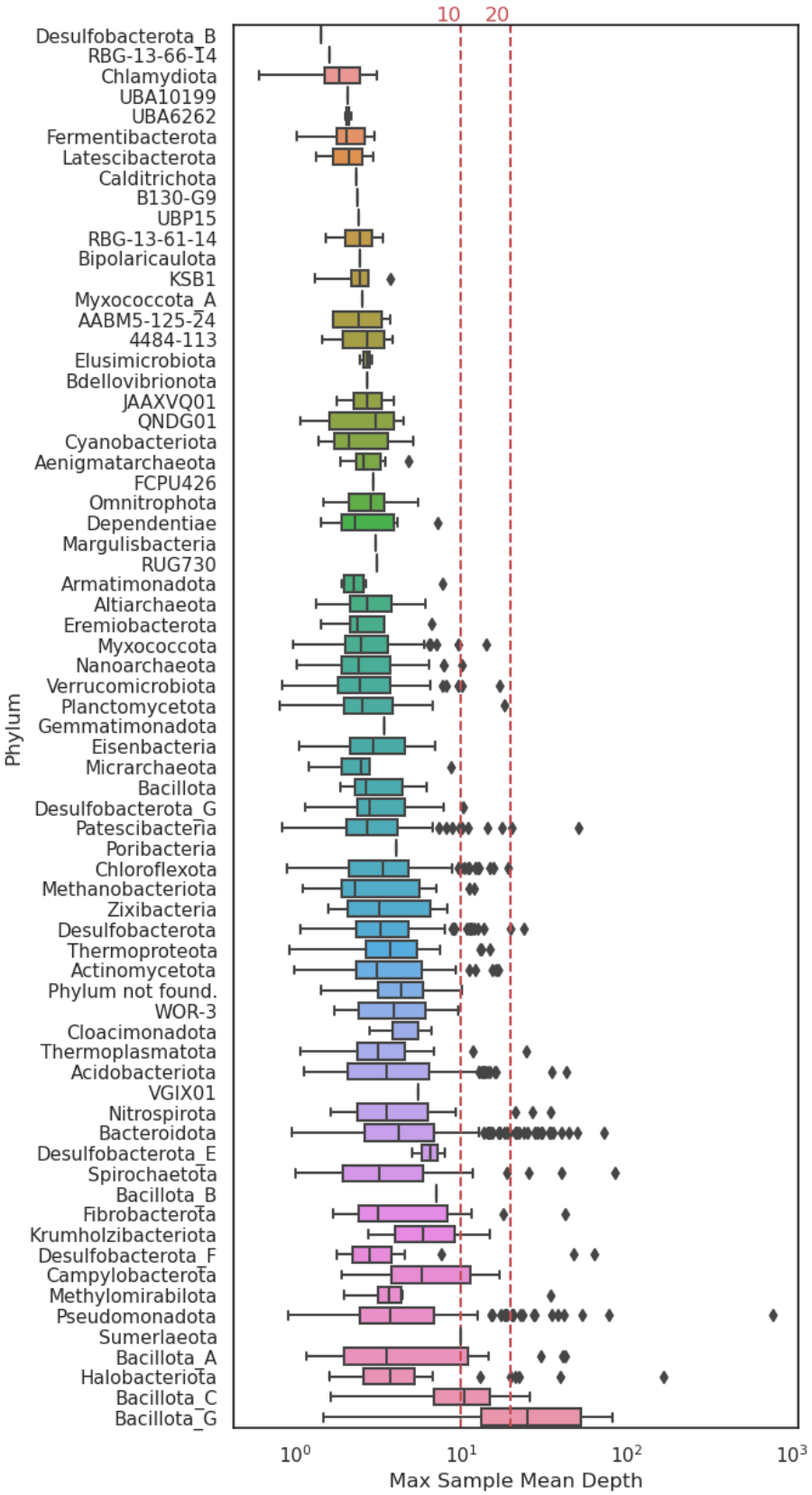


Fig. 3: Caption


10
20
Desulfobacterota_B
RBG-13-66-14
Chlamydiota
UBA10199
UBA6262
Fermentibacterota
Latescibacterota
Calditrichota
B130-G9
UBP15
RBG-13-61-14
Bipolaricaulota
KSB1
Myxococcota_A
AABM5-125-24
4484-113
Elusimicrobiota
Bdellovibrionota
JAAXVQ01
QNDG01
Cyanobacteriota
Aenigmatarchaeota
FCPU426
Omnitrophota
Dependentiae
Margulisbacteria
RUG730
Armatimonadota
Altiarchaeota
Eremiobacterota
Myxococcota
Nanoarchaeota
Verrucomicrobiota
Planctomycetota
Gemmatimonadota
Eisenbacteria
Micrarchaeota
Bacillota
Desulfobacterota_G
Patescibacteria
Poribacteria
Chloroflexota
Methanobacteriota
Zixibacteria
Desulfobacterota
Thermoproteota
Actinomycetota
Phylum not found.
WOR-3
Cloacimonadota
Thermoplasmatota
Acidobacteriota
VGIX01
Nitrospirota
Bacteroidota
Desulfobacterota_E
Spirochaetota
Bacillota_B
Fibrobacterota
Krumholzibacteriota
Desulfobacterota_F
Campylobacterota
Methylomirabilota
Pseudomonadota
Sumerlaeota
Bacillota_A
Halobacteriota
Bacillota_C
Bacillota_G
Phylum
10^0
10^1
10^2
10^3
Max Sample Mean Depth

Phylum	Source	Completeness	Contamination	Quality Score
4484-113	gf	61.53	0.09	61.08
		64.60	0.43	62.45
		95.22	0.96	90.42
	metabat	63.08	1.99	53.13
		69.37	2.19	58.42
		80.69	5.45	53.44
	vamb	50.24	0.02	50.14
		52.94	0.15	52.19
		82.27	0.99	77.32
B130-G9	gf	88.35	1.86	79.05
FCPU426	gf	75.44	1.44	68.24
	metabat	73.53	4.33	51.88
JAAXVQ01	gf	77.67	0.85	73.42
		88.08	3.24	71.88
		89.20	0.17	88.35
	metabat	73.27	0.37	71.42
	vamb	63.49	0.04	63.29
RBG-13-61-14	gf	91.51	0.15	90.76
	vamb	83.96	0.15	83.21
RBG-13-66-14	gf	74.61	1.88	65.21
	vamb	71.00	1.70	62.50
RUG730	gf	100.00	0.00	100.00
	metabat	91.50	0.00	91.50
	vamb	100.00	0.00	100.00
UBA6262	gf	56.83	1.00	51.83
		93.72	1.06	88.42
	metabat	93.60	1.63	85.45
	vamb	92.44	0.36	90.64
UBP15	gf	89.49	2.68	76.09
	vamb	70.45	0.57	67.60
VGIX01	gf	65.01	0.51	62.46

[38] [**?**] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48] [37] [49] [50] [30] [11] [51] [34] [12] [35] [36]

References

[1] Kariin, S., Burge, C.: Dinucleotide relative abundance extremes: a genomic signature. Trends in Genetics **11**(7), 283–290 (1995) https://doi.org/10.1016/s0168-9525(00)89076-9

[2] Kang, D.D., Li, F., Kirton, E., Thomas, A., Egan, R., An, H., Wang, Z.: MetaBAT 2: an adaptive binning algorithm for robust and efficient genome reconstruction from metagenome assemblies. PeerJ **7**, 7359 (2019) https://doi.org/10.7717/peerj.7359

[3] Nissen, J.N., Johansen, J., Allesøe, R.L., Sønderby, C.K., Armenteros, J.J.A., Grønbech, C.H., Jensen, L.J., Nielsen, H.B., Petersen, T.N., Winther, O., Rasmussen, S.: Improved metagenome binning and assembly using deep variational autoencoders. Nature Biotechnology **39**(5), 555–560 (2021) https://doi.org/10.1038/s41587-020-00777-4

[4] Mrazek, J.: Phylogenetic signals in DNA composition: Limitations and prospects. Molecular Biology and Evolution **26**(5), 1163–1169 (2009) https://doi.org/10.1093/molbev/msp032

[5] Hofmeyr, S., Egan, R., Georganas, E., Copeland, A.C., Riley, R., Clum, A., Eloe-Fadrosh, E., Roux, S., Goltsman, E., Buluç, A., Rokhsar, D., Oliker, L., Yelick, K.: Terabase-scale metagenome coassembly with MetaHipMer. Scientific Reports **10**(1) (2020) https://doi.org/10.1038/s41598-020-67416-5

[6] Delgado, L.F., Andersson, A.F.: Evaluating metagenomic assembly approaches for biome-specific gene catalogues. Microbiome **10**(1) (2022) https://doi.org/10.1186/s40168-022-01259-2

[7] Riley, R., Bowers, R.M., Camargo, A.P., Campbell, A., Egan, R., Eloe-Fadrosh, E.A., Foster, B., Hofmeyr, S., Huntemann, M., Kellom, M., Kimbrel, J.A., Oliker, L., Yelick, K., Pett-Ridge, J., Salamov, A., Varghese, N.J., Clum, A.: Terabase-scale coassembly of a tropical soil microbiome. Microbiology Spectrum **11**(4) (2023) https://doi.org/10.1128/spectrum.00200-23

[8] Huang, Y., Wang, Y., Tai, Y., Liu, X., Shen, P., Li, S., Li, J., Huang, F.: CurricularFace: Adaptive curriculum learning loss for deep face recognition. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, ??? (2020). https://doi.org/10.1109/cvpr42600.2020.00594 . https://doi.org/10.1109/cvpr42600.2020.00594

[9] Boudiaf, M., Rony, J., Ziko, I.M., Granger, E., Pedersoli, M., Piantanida, P., Ayed, I.B.: A unifying mutual information view of metric learning: Cross-entropy vs. pairwise losses. In: Computer Vision – ECCV 2020, pp. 548–564. Springer, ??? (2020). https://doi.org/10.1007/978-3-030-58539-6_33 . https://doi.org/10.1007/978-3-030-58539-6_33

[10] Garner, R.E., Kraemer, S.A., Onana, V.E., Fradette, M., Varin, M.-P., Huot, Y., Walsh, D.A.: A genome catalogue of lake bacterial diversity and its drivers at continental scale. Nature Microbiology **8**(10), 1920–1934 (2023) https://doi.org/10.1038/s41564-023-01435-6

[11] Chklovski, A., Parks, D.H., Woodcroft, B.J., Tyson, G.W.: CheckM2: a rapid, scalable and accurate tool for assessing microbial genome quality using machine learning. Nature Methods **20**(8), 1203–1212 (2023) https://doi.org/10.1038/s41592-023-01940-w

[12] Chaumeil, P.-A., Mussig, A.J., Hugenholtz, P., Parks, D.H.: GTDB-Tk v2: memory friendly classification with the genome taxonomy database. Bioinformatics **38**(23), 5315–5316 (2022) https://doi.org/10.1093/bioinformatics/btac672 https://academic.oup.com/bioinformatics/article-pdf/38/23/5315/47465885/btac672.pdf

[13] Bowers, R.M., Kyrpides, N.C., Stepanauskas, R., Harmon-Smith, M., Doud, D., Reddy, T.B.K., Schulz, F., Jarett, J., Rivers, A.R., Eloe-Fadrosh, E.A., Tringe, S.G., Ivanova, N.N., Copeland, A., Clum, A., Becraft, E.D., Malmstrom, R.R., Birren, B., Podar, M., Bork, P., Weinstock, G.M., Garrity, G.M., Dodsworth, J.A., Yooseph, S., Sutton, G., Glöckner, F.O., Gilbert, J.A., Nelson, W.C., Hallam, S.J., Jungbluth, S.P., Ettema, T.J.G., Tighe, S., Konstantinidis, K.T., Liu, W.-T., Baker, B.J., Rattei, T., Eisen, J.A., Hedlund, B., McMahon, K.D., Fierer, N., Knight, R., Finn,

R., Cochrane, G., Karsch-Mizrachi, I., Tyson, G.W., Rinke, C., Lapidus, A., Meyer, F., Yilmaz, P., Parks, D.H., Murat Eren, A., Schriml, L., Banfield, J.F., Hugenholtz, P., Woyke, T.: Minimum information about a single amplified genome (misag) and a metagenome-assembled genome (mimag) of bacteria and archaea. Nature Biotechnology **35**(8), 725–731 (2017) https://doi.org/10.1038/nbt.3893

[14] He, S., Malfatti, S.A., McFarland, J.W., Anderson, F.E., Pati, A., Huntemann, M., Tremblay, J., Rio, T.G., Waldrop, M.P., Windham-Myers, L., Tringe, S.G.: Patterns in wetland microbial community composition and functional gene repertoire associated with methane emissions. mBio **6**(3) (2015) https://doi.org/10.1128/mbio.00066-15

[15] NCBI: National Center for Biotechnology Information. Twitchell wetlands sequencing reads. https://www.ncbi.nlm.nih.gov/sra/. PRJNA362960, PRJNA366360, PRJNA366364, SRP010671, SRP010730, SRP010738, SRP010741, SRP010747, SRP010748 ,SRP010751, SRP010862, SRP010870, SRP010748, SRP011309 (2013-2017)

[16] Alteio, L.V., Schulz, F., Seshadri, R., Varghese, N., Rodriguez-Reillo, W., Ryan, E., Goudeau, D., Eichorst, S.A., Malmstrom, R.R., Bowers, R.M., Katz, L.A., Blanchard, J.L., Woyke, T.: Complementary metagenomic approaches improve reconstruction of microbial diversity in a forest soil. mSystems **5**(2) (2020) https://doi.org/10.1128/msystems.00768-19

[17] Harvard Forest: Long-Term Ecological Research at Harvard Forest. https://harvardforest.fas.harvard.edu/research/LTER. Online; accessed 14 December 2023 (Accessed 2023)

[18] Meyer, F., Fritz, A., Deng, Z.-L., Koslicki, D., Lesker, T.R., Gurevich, A., Robertson, G., Alser, M., Antipov, D., Beghini, F., Bertrand, D., Brito, J.J., Brown, C.T., Buchmann, J., Buluç, A., Chen, B., Chikhi, R., Clausen, P.T.L.C., Cristian, A., Dabrowski, P.W., Darling, A.E., Egan, R., Eskin, E., Georganas, E., Goltsman, E., Gray, M.A., Hansen, L.H., Hofmeyr, S., Huang, P., Irber, L., Jia, H., Jørgensen, T.S., Kieser, S.D., Klemetsen, T., Kola, A., Kolmogorov, M., Korobeynikov, A., Kwan, J., LaPierre, N., Lemaitre, C., Li, C., Limasset, A., Malcher-Miranda, F., Mangul, S., Marcelino, V.R., Marchet, C., Marijon, P., Meleshko, D., Mende, D.R., Milanese, A., Nagarajan, N., Nissen, J., Nurk, S., Oliker, L., Paoli, L., Peterlongo, P., Piro, V.C., Porter, J.S., Rasmussen, S., Rees, E.R., Reinert, K., Renard, B., Robertsen, E.M., Rosen, G.L., Ruscheweyh, H.-J., Sarwal, V., Segata, N., Seiler, E., Shi, L., Sun, F., Sunagawa, S., Sørensen, S.J., Thomas, A., Tong, C., Trajkovski, M., Tremblay, J., Uritskiy, G., Vicedomini, R., Wang, Z., Wang, Z., Wang, Z., Warren, A., Willassen, N.P., Yelick, K., You, R., Zeller, G., Zhao, Z., Zhu, S., Zhu, J., Garrido-Oter, R., Gastmeier, P., Hacquard, S., Häußler, S., Khaledi, A., Maechler, F., Mesny, F., Radutoiu, S., Schulze-Lefert, P., Smit, N., Strowig, T., Bremges, A., Sczyrba, A., McHardy, A.C.: Critical assessment of metagenome interpretation: the second round of challenges. Nature Methods **19**(4), 429–440 (2022) https://doi.org/10.1038/s41592-022-01431-4

[19] Pan, S., Zhao, X.-M., Coelho, L.P.: SemiBin2: self-supervised contrastive learning leads to better MAGs for short- and long-read sequencing. Bioinformatics **39**(Supplement_1), 21–29 (2023) https://doi.org/10.1093/bioinformatics/btad209

[20] Meyer, F., Hofmann, P., Belmann, P., Garrido-Oter, R., Fritz, A., Sczyrba, A., McHardy, A.C.: AMBER: Assessment of Metagenome BinnERs. GigaScience **7**(6), 069 (2018) https://doi.org/10.1093/gigascience/giy069 https://academic.oup.com/gigascience/article-pdf/7/6/giy069/25099078/giy069.pdf

[21] Lynch, M.D.J., Neufeld, J.D.: Ecology and exploration of the rare biosphere. Nature Reviews Microbiology **13**(4), 217–229 (2015) https://doi.org/10.1038/nrmicro3400

[22] Sogin, M.L., Morrison, H.G., Huber, J.A., Welch, D.M., Huse, S.M., Neal, P.R., Arrieta, J.M., Herndl, G.J.: Microbial diversity in the deep sea and the underexplored "rare biosphere". Proceedings of the National Academy of Sciences **103**(32), 12115–12120 (2006) https://doi.org/10.1073/pnas.0605127103

[23] Pascoal, F., Costa, R., Magalhães, C.: The microbial rare biosphere: current concepts, methods and ecological principles. FEMS Microbiology Ecology **97**(1) (2020) https://doi.org/10.1093/femsec/fiaa227

[24] Jørgensen, B.B., Dunker, R., Grünke, S., Røy, H.: Filamentous sulfur bacteria, beggiatoa spp., in arctic marine sediments (svalbard, 79°n). FEMS Microbiology Ecology, (2010) https://doi.org/10.1111/j.1574-6941.2010.00918.x

[25] Woese, C.R., Fox, G.E.: Phylogenetic structure of the prokaryotic domain: The primary kingdoms. Proceedings of the National Academy of Sciences **74**(11), 5088–5090 (1977) https://doi.org/10.1073/pnas.74.11.5088

[26] Schulz, H.N., Brinkhoff, T., Ferdelman, T.G., Marine
, M.H., Teske, A., Jørgensen, B.B.: Dense populations of a giant sulfur bacterium in namibian shelf sediments. Science **284**(5413), 493–495 (1999) https://doi.org/10.1126/science.284.5413.493

[27] Jørgensen, B.B., Dunker, R., Grünke, S., Røy, H.: Filamentous sulfur bacteria, beggiatoa spp., in arctic marine sediments (svalbard, 79°n). FEMS Microbiology Ecology, (2010) https://doi.org/10.1111/j.1574-6941.2010.00918.x

[28] Hol, W.H.G., Boer, W., Termorshuizen, A.J., Meyer, K.M., Schneider, J.H.M., Dam, N.M., Veen, J.A., Putten, W.H.: Reduction of rare soil microbes modifies plant-herbivore interactions. Ecology Letters **13**(3), 292–301 (2010) https://doi.org/10.1111/j.1461-0248.2009.01424.x

[29] Huang, Y., Wang, Y., Tai, Y., Liu, X., Shen, P., Li, S., Li, J., Huang, F.: Curricularface: Adaptive curriculum learning loss for deep face recognition. CoRR **abs/2004.00288** (2020) 2004.00288

[30] Campello, R.J.G.B., Moulavi, D., Zimek, A., Sander, J.: Hierarchical density estimates for data clustering, visualization, and outlier detection. ACM Transactions on Knowledge Discovery from Data **10**(1), 1–51 (2015) https://doi.org/10.1145/2733381

[31] Creevey, C.J., Doerks, T., Fitzpatrick, D.A., Raes, J., Bork, P.: Universally distributed single-copy genes indicate a constant rate of horizontal transfer. PLoS ONE **6**(8), 22099 (2011) https://doi.org/10.1371/journal.pone.0022099

[32] Wang, S., Ventolero, M., Hu, H., Li, X.: A revisit to universal single-copy genes in bacterial genomes. Scientific Reports **12**(1), 14550 (2022) https://doi.org/10.1038/s41598-022-18762-z

[33] Parks, D.H., Imelfort, M., Skennerton, C.T., Hugenholtz, P., Tyson, G.W.: CheckM: assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes. Genome Research **25**(7), 1043–1055 (2015) https://doi.org/10.1101/gr.186072.114

[34] Raschka, S., Patterson, J., Nolet, C.: Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. arXiv preprint arXiv:2002.04803 (2020)

[35] Johnson, J., Douze, M., Jegou, H.: Billion-scale similarity search with GPUs. IEEE Transactions on Big Data **7**(3), 535–547 (2021) https://doi.org/10.1109/tbdata.2019.2921572

[36] Arefin, A.S., Riveros, C., Berretta, R., Moscato, P.: kNN-borůvka-GPU: A fast and scalable MST construction from kNN graphs on GPU. In: Computational Science and Its Applications – ICCSA 2012, pp. 71–86. Springer, ??? (2012). https://doi.org/10.1007/978-3-642-31125-3_6 . https://doi.org/10.1007/978-3-642-31125-3_6

[37] Li, H.: Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. arXiv (2013). https://doi.org/10.48550/ARXIV.1303.3997 . https://arxiv.org/abs/1303.3997

[38] Hyatt, D., Chen, G.-L., LoCascio, P.F., Land, M.L., Larimer, F.W., Hauser, L.J.: Prodigal: prokaryotic gene recognition and translation initiation site identification. BMC Bioinformatics **11**(1), 119 (2010) https://doi.org/10.1186/1471-2105-11-119

[39] Larralde, M., Zeller, G.: PyHMMER: a python library binding to HMMER for efficient sequence analysis. Bioinformatics **39**(5) (2023) https://doi.org/10.1093/bioinformatics/btad214

[40] Larralde, M.: Pyrodigal: Python bindings and interface to prodigal, an efficient method for gene prediction in prokaryotes. Journal of Open Source Software **7**(72), 4296 (2022) https://doi.org/10.21105/joss.04296

[41] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org (2015). https://www.tensorflow.org/

[42] Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv (2014). https://doi.org/10.48550/ARXIV.1412.6980 . https://arxiv.org/abs/1412.6980

[43] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Bach, F., Blei, D. (eds.) Proceedings of the 32nd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 37, pp. 448–456. PMLR, Lille, France (2015). https://proceedings.mlr.press/v37/ioffe15.html

[44] Head, T., MechCoder, Louppe, G., Iaroslav Shcherbatyi, Fcharras, Zé Vinícius, Cmmalone, Schröder, C., Nel215, Campos, N., Young, T., Cereda, S., Fan, T., Rene-Rex, Kejia (KJ) Shi, Schwabedal, J., Carlosdanielcsantos, Hvass-Labs, Pak, M., SoManyUsernamesTaken, Callaway, F., Estève, L., Besson, L., Cherti, M., Karlson Pfannschmidt, Linzberger, F., Cauet, C., Gut, A., Mueller, A., Fabisch, A.: scikit-optimize/scikit-optimize: v0.5.2. Zenodo (2018). https://doi.org/10.5281/ZENODO.1207017 . https://zenodo.org/record/1207017

[45] Deng, J., Guo, J., Xue, N., Zafeiriou, S.: ArcFace: Additive angular margin loss for deep face recognition. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, ??? (2019). https://doi.org/10.1109/cvpr.2019.00482 . https://doi.org/10.1109/cvpr.2019.00482

[46] Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., Teh, Y.W.: Set transformer: A framework for attention-based permutation-invariant neural networks. In: Proceedings of the 36th International Conference on Machine Learning, pp. 3744–3753 (2019)

[47] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I.: Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc., ??? (2017). https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[48] Jeffrey, H.J.: Chaos game representation of gene structure. Nucleic Acids Research **18**(8), 2163–2170 (1990) https://doi.org/10.1093/nar/18.8.2163 https://academic.oup.com/nar/article-pdf/18/8/2163/7059915/18-8-2163.pdf

[49] Bushnell, B.: BBMap: A Fast, Accurate, Splice-Aware Aligner (2022). https://sourceforge.net/projects/bbmap/

[50] Liu, B., Cao, Y., Lin, Y., Li, Q., Zhang, Z., Long, M., Hu, H.: Negative Margin Matters: Understanding Margin in Few-shot Classification. arXiv (2020). https://doi.org/10.48550/ARXIV.2003.12060 .

https://arxiv.org/abs/2003.12060

[51] Olm, M.R., Brown, C.T., Brooks, B., Banfield, J.F.: dRep: a tool for fast and accurate genomic comparisons that enables improved genome recovery from metagenomes through de-replication. The ISME Journal **11**(12), 2864–2868 (2017) https://doi.org/10.1038/ismej.2017.126