ResMiCo: increasing the quality of metagenome-assembled genomes with deep learning

Olga Mineeva[1,2,3,†], Daniel Danciu[1,†], Bernhard Schölkopf[1,2,5], Ruth E. Ley[4], Gunnar Rätsch[1,3,5,6,7,*], Nicholas D. Youngblut[4,*]

1 Department of Computer Science, ETH Zürich, Zürich, Switzerland
2 Department of Empirical Inference, Max Planck Institute for Intelligent Systems, Tübingen, Germany
3 Swiss Institute for Bioinformatics, Lausanne, Switzerland
4 Department of Microbiome Science, Max Planck Institute for Biology, Tübingen, Germany
5 ETH AI center, ETH Zürich, Zürich, Switzerland
6 Department of Biology, ETH Zürich, Zürich, Switzerland
7 Medical Informatics Unit, Zürich University Hospital, Zürich, Switzerland.

[†]The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.
* nyoungblut@tuebingen.mpg.de, gunnar.raetsch@inf.ethz.ch

Abstract

The number of published metagenome assemblies is rapidly growing due to advances in sequencing technologies. However, sequencing errors, variable coverage, repetitive genomic regions, and other factors can produce misassemblies, which are challenging to detect for taxonomically novel genomic data. Assembly errors can affect all downstream analyses of the assemblies. Accuracy for the state of the art in reference-free misassembly prediction does not exceed an AUPRC of 0.57, and it is not clear how well these models generalize to real-world data. Here, we present the Residual neural network for Misassembled Contig identification (ResMiCo), a deep learning approach for reference-free identification of misassembled contigs. To develop ResMiCo, we first generated a training dataset of unprecedented size and complexity that can be used for further benchmarking and developments in the field. Through rigorous validation, we show that ResMiCo is substantially more accurate than the state of the art, and the model is robust to novel taxonomic diversity and varying assembly methods. ResMiCo estimated 4.7% misassembled contigs per metagenome across multiple real-world datasets. We demonstrate how ResMiCo can be used to optimize metagenome assembly hyperparameters to improve accuracy, instead of optimizing solely for contiguity. The accuracy, robustness, and ease-of-use of ResMiCo make the tool suitable for general quality control of metagenome assemblies and assembly methodology optimization.

Author summary

Metagenome assembly quality is fundamental to all downstream analyses of such data. The number of metagenome assemblies, especially metagenome-assembled genomes (MAGs), is rapidly increasing, but tools to assess the quality of these assemblies lack the accuracy needed for robust quality control. Moreover, existing models have been trained

on datasets lacking complexity and realism, which may limit their generalization to novel data. Due to the limitations of existing models, most studies forgo such approaches and instead rely on CheckM to assess assembly quality, an approach that only utilizes a small portion of all genomic information and does not identify specific misassemblies. We harnessed existing large genomic datasets and high-performance computing to produce a training dataset of unprecedented size and complexity and thereby trained a deep learning model for predicting misassemblies that can robustly generalize to novel taxonomy and varying assembly methodologies.

Introduction

Metagenome sequencing is rapidly increasing in popularity due to the lowering costs of sequencing and simplified library construction methods [1, 2]. At the same time, improvements in metagenome assembly tools [3, 4] and high-performance computing resources have increased the feasibility of large-scale metagenome assemblies on 1000s of samples [5–7]. The contiguous sequences (contigs) generated via metagenome assembly can be analyzed directly for such tasks as creating gene catalogs [8, 9], or binning approaches can be used to cluster the contigs into metagenome-assembled genomes (MAGs) that can be used for various comparative genomics applications [10].

These advances have given rise to vast genome assembly databases, such as the Unified Human Gastrointestinal Genome (UHGG) [11], in which MAGs account for 70% of the species. As another example, the Genome Taxonomy Database (GTDB) expanded from 32,000 species to nearly 50,000 in less than one year [12, 13], largely due to the proliferation of MAGs. Given the low-throughput nature of isolating Bacteria and Archaea for independent genome sequencing [11], metagenome assembly approaches will likely continue to dominate.

The correct assembly of metagenomes is challenging due to several factors, including sequencing errors, high taxonomic diversity often comprising 1000s of species, uneven coverage, and repetitive genomic regions [14]. All of these factors contribute to misassemblies, with the most common being structural variations, relocations, translocations, and inversions [15]. Long read sequence data can mitigate some of these issues [14, 16], but the expense relative to short read sequencing generally prevents one from obtaining sufficient sequence coverage for complex communities [17]. While assembly contiguity can be assessed easily by calculating such metrics as N50, assessing assembly accuracy is considerably more challenging due to a few major causes. First, due to a lack of very closely related taxa with genome regions (nearly) identical to the query, contigs cannot simply be mapped to references in order to assess accuracy. Second, reference-free tools that predict misassemblies have generally been trained and validated on small, homogeneous datasets in the past, which raises the question of their robustness to novel data (e.g., novel taxa or assembly methods). Indeed, Mineeva and colleagues showed that existing tools generally performed poorly on a large, heterogeneous dataset [18]. The authors' novel deep learning approach, DeepMAsED, amply outperformed the state of the art and was relatively robust to taxonomic novelty, achieving an AUPRC score of 0.57 on a novel genome dataset; still, there remained substantial room for improvement in model accuracy and also a robust validation on complex datasets spanning the heterogeneity of existing metagenomes from complex communities. Indeed, Lei and colleagues developed metaMIC, a reference-free machine learning (ML) model for predicting misassemblies in metagenome assemblies [19], and showed that DeepMAsED's performance was inferior to metaMIC's; however, only few methodological details were provided on the validation approach.

We present Residual Neural Network for Misassembled Contigs Identification (ResMiCo), a novel approach for reference-free identification of misassembled contigs in

metagenome assemblies. ResMiCo is a deep convolutional neural network with skip connections between non-adjacent layers. Similar architectures have proven to be highly successful when trained on large datasets from various fields [20, 21]. We utilize a novel high throughput pipeline to generate complex and realistic training data covering much of the possible parameter space (e.g., varying data richness, sequencing depth, sequencing error rate, community diversities, and metagenome assembly methods). Through extensive evaluation, we show that the model outperforms the existing state of the art and is robust to metagenome data heterogeneity, including taxonomic novelty and metagenome assembly parameters. ResMiCo is also robust to alternative data simulation approaches, as shown when applied to the Critical Assessment Metagenome Interpretation (CAMI) datasets. We show that using ResMiCo to filter putative genomes reduces the number of misassembled contigs by a factor of four. We also apply ResMiCo to a large collection gut metagenomes from published studies and show that 4.7% $\pm$ 4.2 (s.d.) of contigs per metagenome are misassembled. Lastly, we show that ResMiCo can be used to optimize metagenome assembler parameters for accuracy without the need for simulated or mock-community metagenome datasets.

Materials and methods

Simulated data

We used synthetic datasets for initial model training and testing. The data simulation methodology, depicted in Fig 1, builds on, and significantly expands on our previous work [18]. Reference bacterial and archaeal genomes were selected from Release 202 of the Genome Taxonomy Database (GTDB) [22]. Metagenomes were simulated from publicly available reference genomes via MGSIM (`https://github.com/nick-youngblut/MGSIM`). Simulation parameters varied in all combinations of i) community richness, ii) community abundance distribution, iii) reference genomes selected from the total pool, iv) read length, v) insert size distribution, vi) sequencer error profile, vii) sequencing depth, and viii) metagenome assembler (Table 1). The abundance distribution of each community was modeled as a log-normal distribution. We varied parameter σ to produce differing levels of evenness of relative abundances. Community richness was altered by random sub-sampling from the pool of reference genomes available in the training or test split. The ART [23] read simulator was used to generate paired-end Illumina reads of length 100 or 150 using either the default "Illumina HiSeq 2500" error profile or the "HiSeq2500L150R1/2" error profile used in CAMISIM [24]. Two paired-end read insert size distributions were simulated via the ART parameter settings: "–art-mflen 270 –art-sdev 50" and "–art-mflen 350 –art-sdev 75". We included multiple simulation replicates, in which the community and read simulation parameters were held constant, but each replicate differed via randomization of the genome sub-sampling within each simulation. The reads from each community were assembled independently with metaSPAdes [3] and MEGAHIT [4].

MetaQUAST [15] was used to identify truly misassembled contigs based on mapping all contigs to the reference genomes used for the simulations. The MetaQUAST-identified contig misassembly labels were used as ground truth. We also extracted MetaQUAST-identified breakpoint positions in the misassembled contigs.

For initial model training and testing, we utilized a pool of 18,000 reference genomes selected from Release 202 of the Genome Taxonomy Database (GTDB). The pool was split at the family taxonomic level so that all genomes in the test dataset belonged to families not present in the training dataset. The resulting split was even, with 9000 genomes used for both training and testing. To reduce bias toward particular species, at

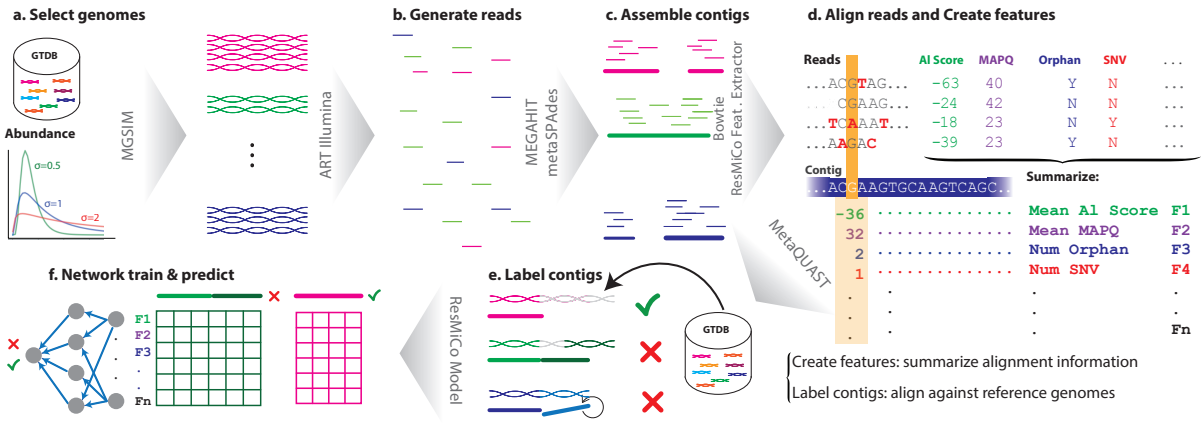

Fig 1. The ResMiCo simulation and training pipeline. a. Select reference genomes from the Genome Taxonomy Database (GTDB) at various abundances; **b.** Simulate reads for the selected genomes using ART-Illumina; **c.** Assemble reads into contigs using MEGAHIT and metaSPAdes; **d.** Align reads back to the assembled contigs using Bowtie2, then extract features such as coverage, number of single-nucleotide variants (SNVs), mean alignment score, etc., for each contig using the given alignments; **e.** Compute labels for each contig by aligning against the reference genomes using MetaQUAST; **f.** For each contig, select a random section (that contains a breakpoint if the contig is misassembled), pad to the network's input length if necessary, and feed the data into the ResMiCo model. Steps **d.** and **e.** are independent and can be parallelized.

Parameter	Values
Community richness	50, 1000, 2000, 3000 genomes
Genome abundance	Lognormal with $\mu = 10$ and $\sigma \in \{0.5, 1, 2\}$
Replicates	3 random selections of reference genomes
Read length	100, 150 bps
Insert size	mean=270 & sd=50, mean=350 & sd=75
Error profile	HiSeq 2500, HiSeq 2500 L150R1/2 [24]
Sequencing depth	0.5m, 2m, 8m, 12m, 20m
Assembler	MEGAHIT, MetaSPAdes

Table 1. Parameter values used in the simulation pipeline. The training dataset *n9k-train* was generated using all 1440 parameter combinations for the first insert size: mean=270, sd=50. For the second insert size (mean=350, sd=75), 3 replicates for the "HiSeq 2500 L150R1/2" error profile and one replicate for the "HiSeq 2500 error" profile were generated (960 parameter combinations). The test dataset *n9k-novel* contains one simulation replicate with the "HiSeq 2500" error profile, and using the first (mean=270, sd=50) insert size (240 parameter combinations).

most 50 genomes per species were included in the reference genome pool, with genomes selected at random. The pool was also filtered by CheckM-estimated completeness ($\geq 90\%$) and contamination ($\leq 5\%$). Other filtering criteria included: i) only "high" MIMAG quality, ii) no single-cell genome assemblies, iii) ≤ 500 contigs, iv) a genome size of ≤ 15 mbp, and v) a mean contig length of ≥ 10 kbp. We randomly subsampled reads mapped to each contig to a maximum mean contig coverage of 20. At this coverage level, assemblies of reasonable quality can be produced, and subsampling helps prevent out-of-distribution issues when applying ResMiCo to datasets with substantially higher sequencing depth than in our training dataset.

To create features for model training and testing, we mapped reads to the contigs from the corresponding synthetic metagenome via Bowtie2 [25], and the resulting alignment data was used to generate per contig position features, as listed in Table S1.

We refer to the training dataset as *n9k-train* and to the test dataset, which consists

of genomes novel at a family taxonomic level, as *n9k-novel*. All 2,400 combinations of simulation parameters were used to generate the training set. The test set was generated using the subset of parameters and one replicate to save computational time (Table 1).

	n9k-train	n9k-novel	CAMI Gut	CAMI Skin	CAMI Oral
Contigs	41.6M	6.8M	0.44M	0.32M	.41M
Bases	123B	19.7B	1.5B	890M	1.5B
Average Coverage	12.8	10.2	16.9	15.7	16.3
Misassemblies	3.04%	4.0%	1.2%	1.72%	1.74%
Median Contig length	1513	1510	1473	1394	1455

Table 2. A summary of the five synthetic datasets used for training and evaluating ResMiCo. The *n9k-train* dataset was used for training and validation, while all other datasets were used for testing.

CAMI simulated metagenomes

To benchmark ResMiCo's performance in a new setting, we downloaded the paired-end reads of the Critical Assessment of Metagenome Interpretation (CAMI) human skin, oral, and gut assembly challenges [26]. As with the *n9k-train* dataset, we assembled the reads via metaSPAdes and MEGAHIT, and identified true misassemblies via MetaQUAST based on the reference genomes in each of the three datasets, then compared the results predicted by ResMiCo with the MetaQUAST output. As shown in Table 2, the number of misassembled contigs in the CAMI datasets is ∼50% lower, while coverage is ∼50% higher relative to the *n9k-train* dataset. The breakpoint locations for misassembled contigs follow a nearly identical distribution for all datasets, with more breakpoints clustered towards the ends (Fig S1).

Published, real-world metagenomes

We evaluated ResMiCo on three published metagenome datasets: *UHGG* [11], *TwinsUK* [27], and *Animal-gut* [28]. *UHGG* consisted of randomly selected metagenomes associated with the UHGG MAG collection. The following filtering criteria were applied prior to metagenome selection: i) $\geq$ 5e6 and $\leq$ 80e6 reads, ii) $\geq$ 10 and $\leq$ 300 MAGs associated with the metagenome in the UHGG, iii) maximum read lengths of $\leq$ 150bp, and iv) $\geq$ 10 samples in the study. *TwinsUK* consisted of human gut metagenomes from adults in the TwinsUK cohort [27], while the *Animal-gut* comprised gut metagenomes from a broad taxonomic diversity of vertebrates [28].

Metagenome read data processing was done as described in Youngblut and colleagues [28]. Briefly, reads were validated with fqtools [29]. Adapters were trimmed with Skewer [30]. The "bbduk" command from bbtools (`https://sourceforge.net/projects/bbmap/`) was used to trim and filter reads based on Phred scores. The "bbmap" command from bbtools was used to filter reads mapping to the hg19 human genome assembly. Read quality reports for each step of the pipeline were generated and visualised with FastQC and MultiQC, respectively (`https://www.bioinformatics.babraham.ac.uk/`, [31]). Metagenomes were assembled via metaSPAdes with default parameters, and contigs < 1000bp were removed. Read mapping and ResMiCo feature generation were conducted as done for the simulation datasets.

Data preprocessing

Count features were normalized by coverage (the number of reads mapped to the position) such that they are in the zero to one range. For numerical features, we

pre-computed mean and standard deviations using all contigs in the *n9k-train* dataset and saved these values. For all datasetes, we standardized numerical features to set the mean to zero and the variance to one using values computed on the training set. Missing values were replaced by zero (the new mean). We summarize the preprocessing applied to each feature in Table S1.

Since we observed that ResMiCo does not generalize well to insert size distributions substantially deviating from the training dataset (Table S5), we excluded metagenomes for which the 0.06 and 0.94 quantiles of the mean insert size distribution lay outside the 0.05 and 0.95 quantiles of the mean insert size distribution of the *n9k-train* dataset, which are equal to 178 and 372, respectively.

Model and training

Architecture

The ResMiCo neural network architecture is shown in Fig 2. It belongs to a class of deep convolutional residual neural networks. The main building unit, depicted at the bottom of the figure, is the residual block, consisting of two batch-normalized-convolutions [32] with a ReLu activation. The input of the residual block is connected with the output by simple element-wise addition [20]. Since convolutions are not padded, the last $2 * (K - 1)$ positions in the residual input, where K is the convolution kernel size, are cut off to match the output size. When the residual block is downsampling the data (by using a stride of $S = 2$ in the first convolution), the residual input is downsampled with a $K = 1$ convolution of an identical stride.

Residual blocks with the same number of filters and identical output shapes are grouped into *residual groups*. ResMiCo has 4 residual groups, with the center groups consisting of 5 residual blocks each, while the outside groups contain 2 residual blocks. Each of the last three residual groups starts with a convolution that doubles the number of filters and halves the input size using a stride $S = 2$. All layers use a ReLU activation, except for the last fully connected layer, which uses a sigmoid activation. The output of the convolutional layers is summarized along the spatial axis via global average pooling, resulting in an output shape that depends only on the number of filters in the last convolutional layer (128 in our case) rather than on the contig length, thus allowing ResMiCo to handle contigs of variable length. Contigs in a batch are padded to the longest length, and the effects of padding are neutralized by creating a mask that is fed to the global average pooling layer. The resulting 128 features of the global average pooling are fed into the final two layers: a fully connected layer with 50 neurons and a one-neuron output layer with a sigmoid activation function.

Training

The model was trained on the *n9k-train* training dataset for 60 epochs. One *epoch* is one pass of the entire training dataset through the algorithm. All misassembled contigs were used as positive training examples. In contrast, we randomly selected a 10% subset at every training epoch for the over-represented class of correctly assembled contigs, thus artificially increasing the positive sample rate to 23.6%. This helped to balance the dataset and reduce the computational load during training. For contigs shorter than 20,000 base-pairs, the entire contig is selected and zero-padded to the maximum batch length. For misassembled contigs longer than 20,000 base-pairs, a random 20,000 base-pair interval around each breakpoint (as identified by MetaQUAST) was selected. For long contigs with no misassemblies, a random 20,000 base-pair interval is selected.

During model training, the binary cross-entropy loss between the target and predicted output was minimized by an Adam optimizer [33]. We used a batch size of

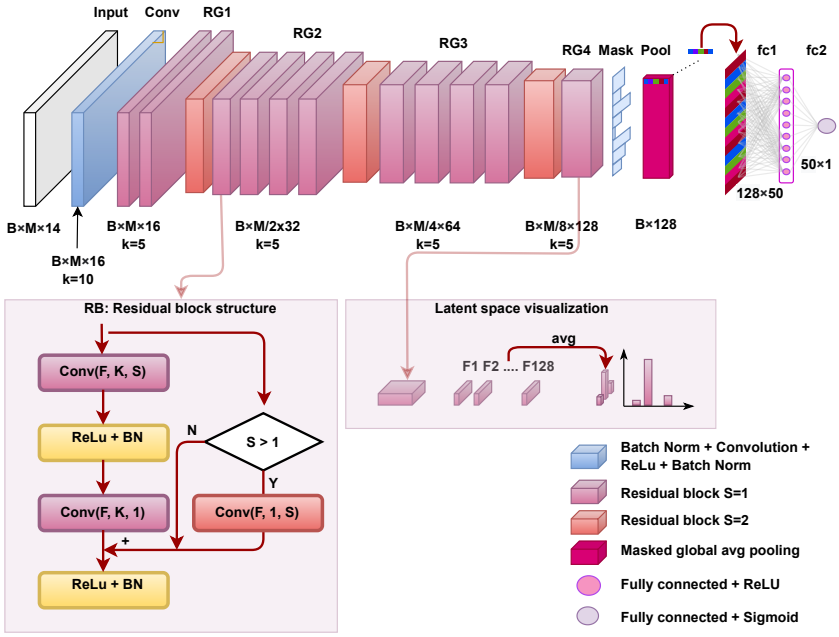

Fig 2. ResMiCo architecture. The input is first passed through multiple convolutional layers; then the convolved result is masked to eliminate the effect of padding and passed through an average pooling layer, followed by two fully connected layers of sizes 128×50 and 50×1. The convolutional part consists of a simple convolution, followed by four residual groups (RG) with 2, 5, 5, and 2 residual blocks, respectively. The bottom of the figure depicts the structure of a residual block with a given number of features (F), kernel size (K), and stride (S). The first convolution in RG2, RG3, and RG4 halves the input size (using a stride of S=2) and doubles the number of filters, gradually from 16 to 128. B denotes the batch size, and M represents the maximum contig length. Overall, ResMiCo has 562,573 parameters, of which 559,441 are trainable.

200 and an initial learning rate of 0.0001 with exponential decay of 0.8 when plateauing at evaluation. Gradients were clipped to a norm of 1 and a value of 0.5.

Model selection

We used 10% of the *n9k-train* dataset as a validation set for model selection (*n9k-valid*). AUPRC on the validation set was computed every second epoch; if the score improved, a corresponding model was saved. The ResMiCo model described in this section achieved the highest AUPRC on the *n9k-valid* dataset at epoch 44. The list of optimized hyperparameters and the attempted values are provided in Table S2.

Feature selection

Since ResMiCo uses a larger number of features than both DeepMAsED and metaMIC, it is important to understand the amount that each feature, particularly features unique to ResMiCo, contributes to model predictions. Borrowed from game theory, Shapley values provide a principled way of explaining the predictions of machine learning models. We approximated the Shapley values using the Deep Shap algorithm [34], a refined version of DeepLIFT [35].

In order to be able to compute Deep Shap coefficients, we had to make some adjustments to ResMiCo's architecture: the input size was fixed, the padding was not masked, and the global average pooling layer was replaced by local pooling with a window covering the whole length. Deep Shap requires as input background samples as well as samples for which the predictions will be explained. We randomly sampled 200 contigs for the background and 200 contigs for explanations (100 correctly assembled and 100 misassembled) from the *n9k-novel* dataset.

Figure 3 shows features ranked by their importance. For comparison, we also marked features present in the ResMiCo pipeline that were used by metaMIC and DeepMAsED. Feature names are explained in Table S1. The top 14 features were selected as input to the ResMiCo model. We included at least one feature of each kind: mapping quality, alignment score, etc. Limiting the number of features resulted in significantly reduced training time.

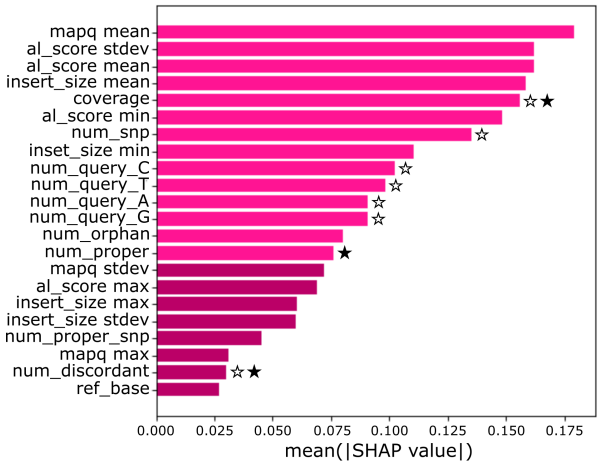

Fig 3. Feature ranked by their importance. The lighter color marks features used by the ResMiCo model. We denote features used by DeepMAsED and metaMIC with a star outline and filled star, respectively. `mapq` and `al_score` are mapping quality and alignment score, as defined by Bowtie2. `num_snp` is the number of SNVs among aligned reads relative to the reference. `num_query_[ATGC]` is the base composition of aligned reads at the target position. `num_orphan` is the number of aligned reads in which only one of the pairs aligns properly. `num_proper` is the number of read pairs that align properly, as defined by Bowtie2. `num_proper_snp` is properly aligned reads with a SNV relative to the reference at the target position. `ref_base` is the reference base `[ATGC]` at the target position.

Predictions

To predict the misassembly probability for contigs longer than 20,000 base pairs, we split the contig into chunks of 20,000 base pairs, with a stride (overlap) of 500 bases. The prediction for the contig was obtained by selecting the maximum score across all chunks. For contigs shorter than 20,000 base pairs, the entire contig is given as input.

Comparison to the state of the art

We analyzed the performance of our proposed model in relation to the following existing methods:

- metaMIC [36] – applied with default parameters, except the minimum contig length was reduced from 5 kbp to 1 kbp, in order to test on the same data as the other methods;
- DeepMAsED [18] – we followed the feature generation scheme and the trained model provided by the authors;
- ALE [37] – we aggregated four positional sub-scores that ALE outputs (depth, place, insert, and k-mer log-likelihoods) with the same thresholds defined in [18]. The contig misassembly probability is computed as the number of positions with the sub-score below the threshold divided by contig length;
- Random – we assigned a random misassembly probability to each contig. This results in a horizontal line on a precision-recall curve plot with a precision equal to the prevalence of misassemblies in the dataset.

Since all datasets suffer from a class imbalance in the detriment of positive samples (misassembled contigs, Table 2), we selected the area under the precision-recall curve (AUPRC) as a metric to measure performance, rather than the area under the receiver operator curve (AUROC) [38]. However, AUPRC is not invariant to the prevalence of positive samples, so we used AUROC to compare the model's performance across datasets with different percentages of positive samples.

Code and data availability

ResMiCo is publicly available at (`https://github.com/leylabmpi/ResMiCo`). The computationally intensive feature extraction library was written in C++, and the data simulation pipeline was implemented using Snakemake [39]. The deep learning model was built using Tensorflow [40].

The *n9k-train* and *n9k-novel* datasets are publicly available at `http://ftp.tue.mpg.de/ebio/projects/ResMiCo/`.

Results

ResMiCo outperforms existing models and is robust to metagenome novelty

We first tested all models against the *n9k-novel* dataset, which consisted of family-level taxonomically novel genomes relative to any in the training dataset. ResMiCo outperformed DeepMAsED, ALE, and metaMIC by a large margin, with an AUPRC of 0.76 versus 0.25 for DeepMAsED, the second-best performing model (Fig 4**a.**). Note that DeepMAsED's AUPRC score dropped from 0.57 (reported in [18]) to 0.25 due to a higher variability within the *n9k-novel* test set. Importantly, ResMiCo's AUPRC did not substantially differ between the training validation (0.73) and the *n9k-novel* dataset (0.76), thereby demonstrating that the model is robust to taxonomic novelty. ResMiCo's AUPRC score typically varied from 0.6 to 0.8 across the various simulation parameter combinations (Fig S6). The most challenging settings for ResMiCo were a low community richness and a low sequencing depth. We also found that the contig length distribution explained much of the variability in ResMiCo model performance. For the simulations with a median contig length longer than 2000bp, the AUPRC was between 0.4 and 0.6 (Fig S5).

We next evaluated ResMiCo on the CAMI gut, oral, and skin metagenome datasets, which are commonly used simulation datasets for the evaluation metagenomics analysis

tools. The CAMI datasets differed substantially from *n9k-train* and *n9k-novel* in regards to coverage (sequencing depth) and class imbalance (percent misassemblies) (Table 2). Moreover, reference genomes used for the *n9k-train* and *n9k-novel* datasets were selected from the entire GTDB, while the CAMI datasets consisted of biome-specific reference genomes [24]. Regardless of these differences, ResMiCo's performance remained largely unaffected, and the model still clearly outperformed all competitors (Fig 4**b.**–**d.**). Given that the 5 synthetic datasets differ substantially in true positive rates (Table 2), we computed the AUROC score, which is unaffected by such differences. Fig 4**e.** shows that the AUROC remains relatively constant across the *n9k-train* validation, *n9k-novel*, and the CAMI datasets.

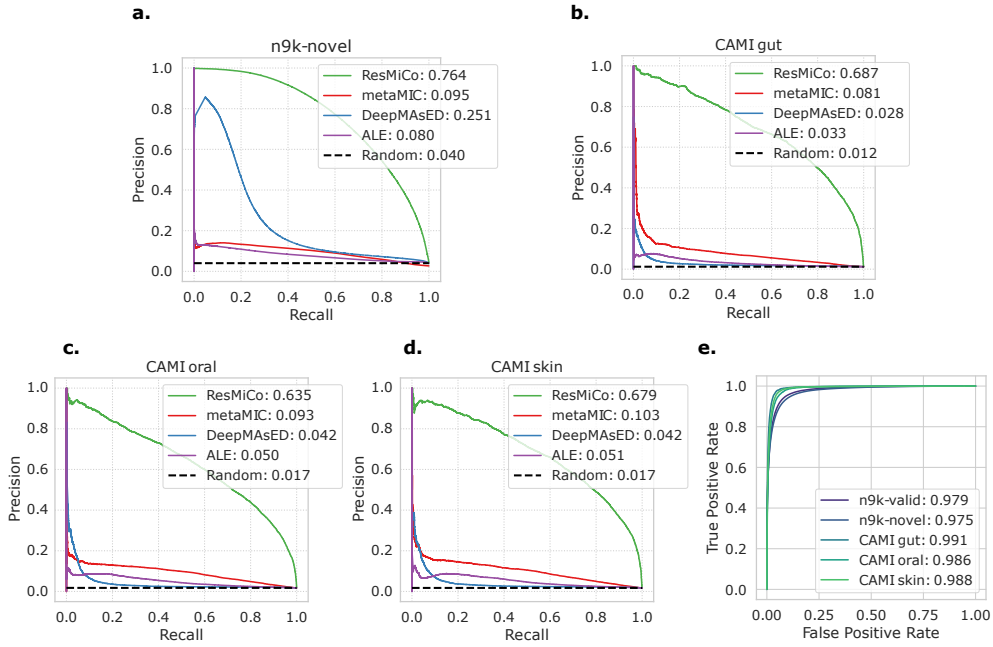

Fig 4. ResMiCo performance evaluation. Precision-recall curves and the corresponding AUPRC scores for ResMiCo and four baseline methods (metaMIC, DeepMAsED, ALE, Random) applied on the **a.** *nk9-novel*, **b.** CAMI gut, **c.** CAMI oral, and **d.** CAMI skin datasets. **e.** Receiver operating characteristic curve and the corresponding AUROC scores for ResMiCo applied on five datasets: *n9k-train* (validation set only), *nk9-novel*, and three CAMI datasets

Improvement of assembly quality after filtering ResMiCo-identified misassemblies

A primary function of ResMiCo is to identify misassembled contigs so that they can be removed from the assembly. To illustrate the effects of such filtering, we discarded contigs in *n9k-novel* with a ResMiCo score of > 0.8, which corresponds to a recall and precision of 0.72 and 0.65, respectively. This filtering resulted in a reduction of the true error rate from 4% to 1% while keeping the contiguity metrics virtually unmodified (Table 3).

Dataset	True error rate	N contigs	N50	Mean length	Median length
Original	0.040	6779977	3919	2911	1510
ResMiCo	**0.012**	6478409	3886	2904	1521

Table 3. Low-quality contig filtering. Statistics before and after filtering low-quality contigs with the ResMiCo model applied on the *n9k-novel* test set. The ResMiCo score threshold was set to > 0.8.

Assembly optimization based on ResMiCo-identified error rates

Since ResMiCo generates a score for each contig, we could use the number of contigs with a score above a certain threshold (we used > 0.8 in our experiments) to estimate the misassembly rate of a given contig set. The estimated misassembly rate could then be used to optimize metagenome assembler parameters (e.g., k-mer lengths) for real metagenomes, which lack ground-truth data. Assembler hyperparameters are generally optimized simply based on total continuity (e.g., N50) or possibly via CheckM after binning contigs into MAGs. However, such methods do not directly assess contig assembly accuracy. In order to use ResMiCo for this application, model performance must be robust to assembler hyperparameter settings outside of the training distribution. We tested ResMiCo's performance as an oracle for assembler performance by simulating datasets in a similar fashion as *n9k-novel*, but with 6 different k-mer length hyperparameter settings for both MEGAHIT and metaSPAdes (see Methods). For each of the 6 k-mer length combinations, we generated akin to *n9k-novel* but utilized only 2 community richness (50 and 3000) and 2 sequencing depth (8m and 20m) settings. The rest of the simulation parameters were fixed: genome abundance distribution with $\sigma = 1$, read lengths of 150 bps, insert size distribution of mean=270 & sd=50, and the "HiSeq 2500 error" error profile. The percentage of actual misassembled contigs differed from $< 1\%$ to 30% depending on the assembler and the chosen k-mer set (Fig 5). We then compared the percentage of misassembled contigs with the percentage estimated by ResMiCo for each of the four community richness/sequencing depth combinations (6 k-mer sets per combination).

ResMiCo was able to accurately rank the assemblies for all four simulation parameter combinations, achieving a Pearson correlation of 0.9. (Fig 5 **a.**). In contrast, the N50 value does not correlate with the error rate and is a complementary measure of assembly quality. (Fig 5 **b.**). Consequently, we propose that ResMiCo can be used to rank assembler parameters for real-world metagenome data and identify parameters leading to the lowest misassembly rate.

Latent space visualization

To get an intuition on how ResMiCo internally represents data, we studied the output of the global average pooling layer. At that point, the input data is mapped into a 128-dimensional space. We used UMAP [41] with default parameters to project the embeddings into a two-dimensional space. UMAP was fitted on the *n9k-train*, *n9k-novel*, and *CAMI-gut* datasets. We used 10,000 randomly sampled contigs from each of the three datasets.

The latent space visualization indicates that *n9k-train* has more variability (due to the extensive set of parameters used in the simulations) than *CAMI-gut*, which is concentrated in a small subspace, while misassembled contigs from both datasets are generally clustered together (Fig 6**a.,b.**). Note that since ResMiCo has two fully connected layers following the visualized global average pooling, the two classes are not expected to be completely separable at this stage. Both community richness and average contig coverage strongly partition the latent space (Fig 6**c.,d.**).

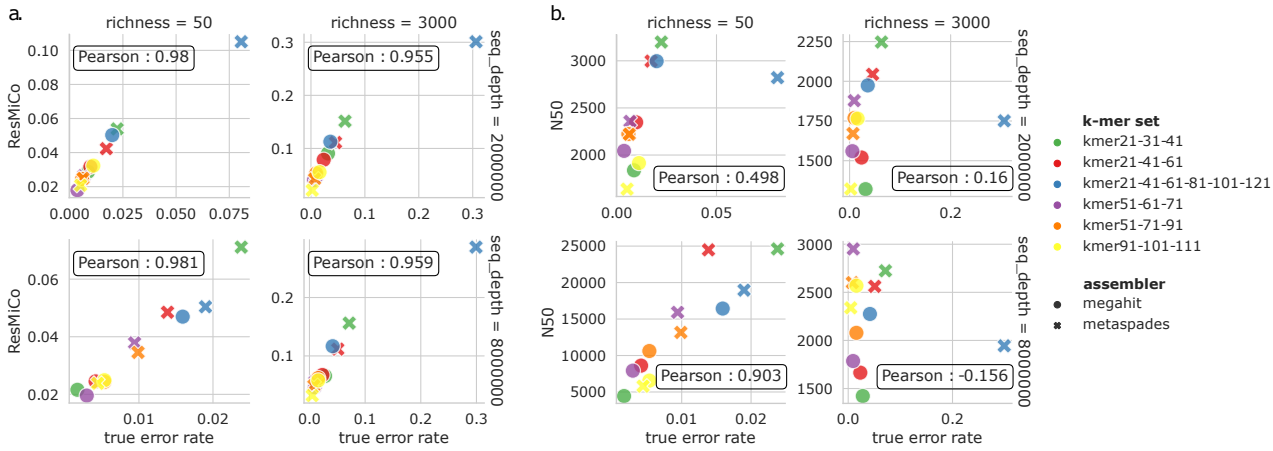

Fig 5. Misassembly (error) rate produced by MEGAHIT and metaSPAdes assemblers with six different k-mer sets. k-mer set names denote the k-mer lengths used for the assembly. **a.** ResMiCo-identified error rate (y-axis) correlates with the true error rate. **b.** N50 size and the true error rate are orthogonal measures of the metagenome quality.

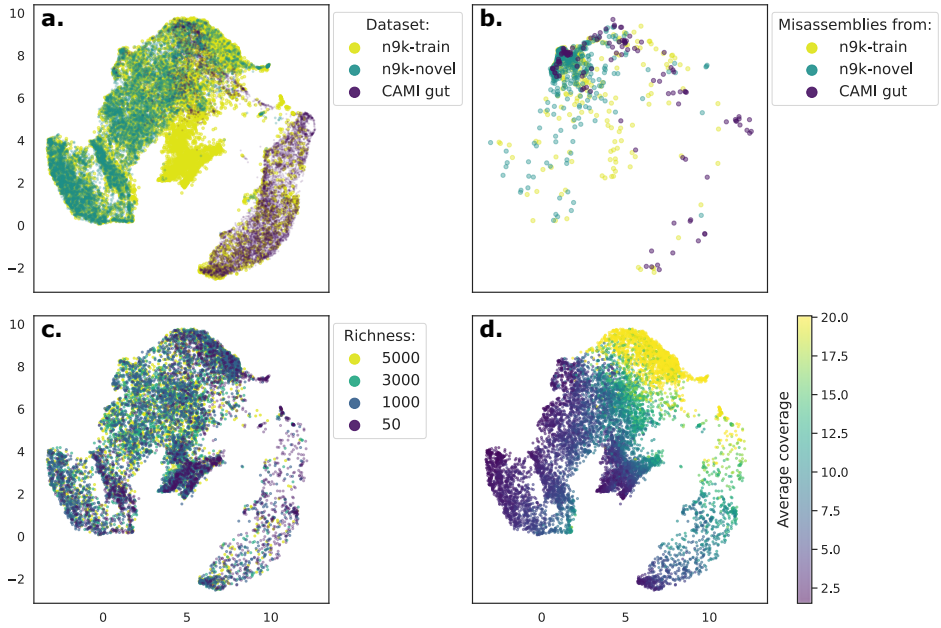

Fig 6. Contig embeddings learned by ResMiCo, projected using UMAP. The first row shows **a.** all contig embeddings and **b.** misassembled contig embeddings for the *n9k-train*, the *n9k-novel*, and the *CAMI. gut* datasets. In the second row, contigs from the *n9k-train* dataset are colored **c.** by the richness of the simulated community they originated from and **d.** by their average coverage.

ResMiCo detects a 3-7% misassembly rate in real-world metagenomes

We applied ResMiCo to published gut metagenome datasets from multiple studies in order to assess the prevalence of misassembled contigs in publicly available metagenomic data. We utilized three datasets: *UHGG*, *TwinsUK*, and *Animal-gut*. *UHGG* consisted of a random subset of gut metagenomes associated with MAGs in the UHGG database, while *TwinsUK* and *Animal-gut* consisted of gut metagenomes from westernized adults

and a broad taxonomic diversity of vertebrates, respectively (see Methods).

ResMiCo detected an average of 3%, 3%, and 7% misassembled contigs across all metagenome assemblies in the *TwinsUK*, and *Animal-gut*, and *UHGG* datasets, respectively. Overall, we evaluated 4,955,309 contigs, of which 4% are misassembled according to ResMiCo's predictions (≥ 0.8).

At the per-metagenome level, ResMiCo detected $3.1\% \pm 0.5$, $4.9\% \pm 6.5$, and $7.9\% \pm 4.8$ (s.d.) missassembled contigs in the *TwinsUK*, and *Animal-gut*, and *UHGG* datasets, respectively (Table S7). Averaged across all metagenomes, the misasssembly rate was $4.7\% \pm 4.2$ (s.d.). The high variability among samples and datasets suggests that sample-specific factors (e.g., community taxonomic complexity or variability among NGS library preparations) can substantially influence misassembly rates.

Discussion

We addressed the problem of reference-free metagenome quality evaluation by developing ResMiCo, a deep residual neural network that enables accurate misassembled contig identification. ResMiCo provides an efficient data generation pipeline (see Supplementary Methods), which transforms raw reads and contigs into positional features that are utilized by a residual neural network to predict if a given contig was misassembled. The ResMiCo model was trained and tested on datasets of unprecedented size and complexity (*n9k-train* and *n9k-novel* contain 143 Gbps of assembled contigs), which we have made freely available as a resource for further model development and benchmarking (see Methods). These datasets can be expanded, or new datasets can be generated with ResMiCo's dataset simulation pipeline, which allows for straightforward, efficient data generation on high-performance computing systems.

ResMiCo achieved a 0.76 AUPRC score on the taxonomically novel holdout test set (*n9k-novel*), which is an exceptional improvement over the state of the art (Fig 4**a.**). The robustness of ResMiCo to family-level taxonomic novelty clearly demonstrates that it can be applied to metagenomes that include substantial taxonomic novelty, such as gut microbiomes from poorly studied animal species [28].

When tested on the CAMI gut, oral, and skin datasets, ResMiCo showed similarly high performance and again substantially outperformed the state of the art (Fig 4**b.-d.**). These results show that ResMiCo can generalize to third-party, biome-specific datasets, despite our use of a biome-agnostic training dataset consisting of genomes randomly selected from the entire GTDB.

ResMiCo is primarily designed to increase the quality of existing assemblies, and we demonstrated that filtering out contigs with high scores from the holdout test set resulted in a fourfold decrease in the true error rate without a substantial decline in contiguity (Table 3).

When applied to the real-world metagenome datasets, ResMiCo detected $4.7\% \pm 4.2$ (s.d.) misassembled contigs per metagenome. This estimate is substantially higher than the 1% misassembly rate previously estimated via DeepMAsED [18], which may be due to differences in model accuracy and the increased number and variety of real-world metagenomes used for our estimation.

We also show that ResMiCo can be applied to select assembler parameters corresponding to the best assembly accuracy for a given unlabeled dataset (Fig 5). Researchers can thus optimize assembler parameters for obtaining high accuracy on their specific real-world metagenomes without relying on benchmarks from simulated datasets [24].

ResMiCo's vastly improved performance relative to other reference-free misassembly detection methods is likely due to three main factors. First, ResMiCo was trained on a very large and varied dataset. Even after re-implementing the *samtools pileup* algorithm

to gain a 10x speed improvement, generating the *n9k-train* dataset required nearly 20,000 CPU hours to produce all 2,400 simulation parameter combinations (Table 1), resulting in 41.6M contigs that total 123 Gbps. In contrast, the DeepMAsED training dataset is 80x smaller, while no such dataset information is available for metaMIC. The UMAP projections of the contig embeddings for the *n9k-train*, the *n9k-novel*, and the *CAMI-gut* datasets (Fig6**a**) show that ResMiCo's training data comprises a substantial portion of the input space. Second, ResMiCo was trained on a larger number of carefully selected features; although ResMiCo uses only the top 14 features in Fig 3, we generated and tested a total of 23 features (Table S1) before selecting the 14 best performing ones. In comparison, DeepMAsED used 8 features for training, while metaMIC used 4 types of features, and as we have shown, both models failed to identify the features most relevant to misassembly detection. Third, introducing residual blocks, combined with the larger dataset, allowed us to train a deeper convolutional model, which has been shown to have better performance relative to traditional, shallower CNNs [20]. Long range (up to 20,000 bp) signal from raw positional features is transformed by the residual blocks, such that ResMiCo is able to internally detect a breakpoint (Fig S2) and identify misassembled contigs based on the strength of this signal.

While our extensive evaluations showed that ResMiCo is robust to many sources of dataset novelty, we did find that the model is sensitive to the mean insert size distribution (Table S5). Therefore, prior to evaluation, we removed all metagenomes falling substantially outside of the *n9k-train* insert size distribution. While filtering by insert size may lead to a biased selection of real-world metagenomes, Illumina sequencing libraries often vary substantially in fragment length and subsequently insert size distribution, even within the same sequencing run. Such variation can result from inaccuracies in DNA quantification and reagent aliquoting. ResMiCo automatically detects if the insert size distributions of the evaluated data differ substantially from those during training and warns users that results may be less accurate.

Improving performance robustness across changing distributions, sometimes termed out-of-distribution (o.o.d.) generalization [42–44], could be an interesting direction for follow-up work. While deep learning has produced impressive results in a range of domains, its reliance on independent and identically distributed (i.i.d.) training data can be a problem [45–47]. Although a range of attempts exist to improve o.o.d. generalization, empirical risk minimization still is the method of choice in practice [48, 49], especially when used with a dataset of maximal diversity, as done in our study.

Besides improving o.o.d., there are some other areas for further improvement. First, more research is needed to evaluate the quality of contigs assembled from error-prone long reads (e.g., Oxford Nanopore). Second, it is worth investigating if ResMiCo can be adapted to indicate the location of breakpoints in misassembled contigs. Third, rather than using binary labels, ResMiCo could be trained on misassembly type (e.g., inversion or translocation) to provide more detailed predictions.

In summary, ResMiCo is a major advancement in the challenge of reference-free metagenome quality assessment. Existing methods addressing this problem have not been widely used, likely due to concerns regarding whether such approaches can generalize to real-world datasets. Our extensive testing shows that ResMiCo generalizes well across a large parameter space that includes taxonomy, community abundances, and many sequencing parameters. Wide adoption of ResMiCo could substantially improve metagenome assembly quality for individual studies and databases, which is critical for obtaining accurate biological insights from metagenomic data.

Author's contributions

OM, DD and NY prepared the manuscript. OM and DD wrote, trained and evaluated the deep learning model. NY wrote the data generation pipeline and created all datasets. DD wrote the feature extraction library. OM worked on model interpretability and application scenarios. NY, BS and GR provided supervision and feedback. BS, GR, and RL provided funding acquisition and resources.

Funding

OM and DD were supported by ETH core funding (to GR). OM was also supported by the Max Planck ETH Center for Learning Systems. DD was partially funded by ETH-SFA PHRT project #106. This work was supported by the Department of Microbiome Science at the Max Planck Institute for Biology.

References

1. Gaio D, Anantanawat K, To J, Liu M, Monahan L, Darling AE. Hackflex: low cost Illumina Nextera Flex sequencing library construction. bioRxiv. 2021;doi:10.1101/779215.

2. Hennig BP, Velten L, Racke I, Tu CS, Thoms M, Rybin V, et al. Large-Scale Low-Cost NGS Library Preparation Using a Robust Tn5 Purification and Tagmentation Protocol. G3 Genes—Genomes—Genetics. 2018;8(1):79–89. doi:10.1534/g3.117.300257.

3. Nurk S, Meleshko D, Korobeynikov A, Pevzner PA. metaSPAdes: a new versatile metagenomic assembler. Genome research. 2017;27(5):824–834.

4. Li D, Liu CM, Luo R, Sadakane K, Lam TW. MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. Bioinformatics. 2015;31(10):1674–1676.

5. Nayfach S, Shi ZJ, Seshadri R, Pollard KS, Kyrpides NC. New insights from uncultivated genomes of the global human gut microbiome. Nature. 2019;568(7753):505–510.

6. Almeida A, Mitchell AL, Boland M, Forster SC, Gloor GB, Tarkowska A, et al. A new genomic blueprint of the human gut microbiota. Nature. 2019;568(7753):499–504.

7. Pasolli E, Asnicar F, Manara S, Zolfo M, Karcher N, Armanini F, et al. Extensive unexplored human microbiome diversity revealed by over 150,000 genomes from metagenomes spanning age, geography, and lifestyle. Cell. 2019;176(3):649–662.

8. Qin J, Li R, Raes J, Arumugam M, Burgdorf KS, Manichanh C, et al. A human gut microbial gene catalogue established by metagenomic sequencing. nature. 2010;464(7285):59–65.

9. Xiao L, Feng Q, Liang S, Sonne SB, Xia Z, Qiu X, et al. A catalog of the mouse gut metagenome. Nature biotechnology. 2015;33(10):1103–1108.

10. Breitwieser FP, Lu J, Salzberg SL. A review of methods and databases for metagenomic classification and assembly. Briefings in bioinformatics. 2019;20(4):1125–1136.

11. Almeida A, Nayfach S, Boland M, Strozzi F, Beracochea M, Shi ZJ, et al. A unified catalog of 204,938 reference genomes from the human gut microbiome. Nature biotechnology. 2021;39(1):105–114.

12. Parks DH, Chuvochina M, Waite DW, Rinke C, Skarshewski A, Chaumeil PA, et al. A standardized bacterial taxonomy based on genome phylogeny substantially revises the tree of life. Nature biotechnology. 2018;36(10):996–1004.

13. Youngblut ND, Ley RE. Struo2: efficient metagenome profiling database construction for ever-expanding microbial genome datasets. PeerJ. 2021;9:e12198.

14. Lapidus AL, Korobeynikov AI. Metagenomic data assembly–the way of decoding unknown microorganisms. Frontiers in Microbiology. 2021;12:653.

15. Mikheenko A, Saveliev V, Gurevich A. MetaQUAST: evaluation of metagenome assemblies. Bioinformatics. 2016;32(7):1088–1090.

16. Koren S, Phillippy AM. One chromosome, one contig: complete microbial genomes from long-read sequencing and assembly. Current opinion in microbiology. 2015;23:110–120.

17. Ayling M, Clark MD, Leggett RM. New approaches for metagenome assembly with short reads. Briefings in bioinformatics. 2020;21(2):584–594.

18. Mineeva O, Rojas-Carulla M, Ley RE, Schölkopf B, Youngblut ND. DeepMAsED: evaluating the quality of metagenomic assemblies. Bioinformatics. 2020;36(10):3011–3017.

19. Lai S, Pan S, Coelho LP, Chen WH, Zhao XM. metaMIC: reference-free Misassembly Identification and Correction of de novo metagenomic assemblies. bioRxiv. 2021;.

20. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2016. p. 770–778.

21. Jaganathan K, Kyriazopoulou Panagiotopoulou S, McRae JF, Darbandi SF, Knowles D, Li YI, et al. Predicting Splicing from Primary Sequence with Deep Learning. Cell. 2019;176(3):535–548.e24.

22. Parks DH, Chuvochina M, Waite DW, Rinke C, Skarshewski A, Chaumeil PA, et al. A standardized bacterial taxonomy based on genome phylogeny substantially revises the tree of life. Nat Biotechnol. 2018;36(10):996–1004.

23. Huang W, Li L, Myers JR, Marth GT. ART: a next-generation sequencing read simulator. Bioinformatics. 2012;28(4):593–594.

24. Fritz A, Hofmann P, Majda S, Dahms E, Dröge J, Fiedler J, et al. CAMISIM: simulating metagenomes and microbial communities. Microbiome. 2019;7(1):1–12.

25. Langmead B, Salzberg SL. Fast gapped-read alignment with Bowtie 2. Nat Methods. 2012;9(4):357–359.

26. Sczyrba A, Hofmann P, Belmann P, Koslicki D, Janssen S, Dröge J, et al. Critical assessment of metagenome interpretation—a benchmark of metagenomics software. Nature methods. 2017;14(11):1063–1071.

27. de la Cuesta-Zuluaga J, Spector TD, Youngblut ND, Ley RE, Bordenstein S. Genomic Insights into Adaptations of Trimethylamine-Utilizing Methanogens to Diverse Habitats, Including the Human Gut. mSystems. 2021;6(1):e00939–20. doi:10.1128/mSystems.00939-20.

28. Youngblut ND, De la Cuesta-Zuluaga J, Reischer GH, Dauser S, Schuster N, Walzer C, et al. Large-scale metagenome assembly reveals novel animal-associated microbial genomes, biosynthetic gene clusters, and other genetic diversity. Msystems. 2020;5(6):e01045–20.

29. Droop AP. fqtools: an efficient software suite for modern FASTQ file manipulation. Bioinformatics. 2016;32(12):1883–1884.

30. Jiang H, Lei R, Ding SW, Zhu S. Skewer: a fast and accurate adapter trimmer for next-generation sequencing paired-end reads. BMC Bioinformatics. 2014;15:182.

31. Ewels P, Magnusson M, Lundin S, Käller M. MultiQC: summarize analysis results for multiple tools and samples in a single report. Bioinformatics. 2016;32(19):3047–3048.

32. Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. 2015;.

33. Kingma DP, Ba J. Adam: A Method for Stochastic Optimization. 2014;.

34. Lundberg SM, Lee SI. A Unified Approach to Interpreting Model Predictions. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al., editors. Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc.; 2017.Available from: `https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf`.

35. Shrikumar A, Greenside P, Kundaje A. Learning Important Features Through Propagating Activation Differences. 2017;.

36. Lai S, Pan S, Coelho LP, Chen WH, Zhao XM. metaMIC: reference-free Misassembly Identification and Correction of de novo metagenomic assemblies; 2021.

37. Clark SC, Egan R, Frazier PI, Wang Z. ALE: a generic assembly likelihood evaluation framework for assessing the accuracy of genome and metagenome assemblies. Bioinformatics. 2013;29(4):435–443.

38. Branco P, Torgo L, Ribeiro RP. A survey of predictive modeling on imbalanced domains. ACM Computing Surveys (CSUR). 2016;49(2):1–50.

39. Köster J, Rahmann S. Snakemake—a scalable bioinformatics workflow engine. Bioinformatics. 2012;28(19):2520–2522. doi:10.1093/bioinformatics/bts480.

40. Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al.. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems; 2015. Available from: `https://www.tensorflow.org/`.

41. McInnes L, Healy J, Melville J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. 2018;.

42. Azulay A, Weiss Y. Why Do Deep Convolutional Networks Generalize so Poorly to Small Image Transformations? JMLR. 2019;.

43. Schott L, von Kugelgen J, Trauble F, Gehler P, Russell C, Bethge M, et al. Visual Representation Learning Does Not Generalize Strongly within the Same Domain. In: ICLR; 2022.

44. Hendrycks D, Basart S, Mu N, Kadavath S, Wang F, Dorundo E, et al. The Many Faces of Robustness: A Critical Analysis of Out-of-distribution Generalization. In: ICCV; 2021.

45. Ben-David S, Blitzer J, Crammer K, Pereira F. Analysis of representations for domain adaptation. In: Advances in neural information processing systems; 2007. p. 137–144.

46. Scholkopf B. Artificial intelligence: Learning to see and act. Nature. 2015;518(7540):486–487.

47. Beery S, Van Horn G, Perona P. Recognition in Terra Incognita. In: ECCV; 2018.

48. Gulrajani I, Lopez-Paz D. In Search of Lost Domain Generalization. In: ICLR; 2021.

49. Miller JP, Taori R, Raghunathan A, Sagawa S, Koh PW, Shankar V, et al. Accuracy on the Line: on the Strong Correlation Between Out-of-Distribution and In-Distribution Generalization. In: ICML; 2021.