PlasLR Enables Adaptation of Plasmid Prediction for Error-Prone Long Reads

Anuradha Wickramarachchi[1], Vijini Mallawaarachchi[1], Lianrong Pu[2,3,*] and Yu Lin[1,*]

1 School of Computing, College of Engineering and Computer Science, Australian National University, Canberra, ACT 0200, Australia
2 College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350116, China
3 School of Computer Science, Tel Aviv University, Tel Aviv 6997801, Israel

* lianrong.pu@gmail.com or yu.lin@anu.edu,au

Abstract

Plasmids are extra-chromosomal genetic elements commonly found in bacterial cells that support many functional aspects including environmental adaptations. The identification of these genetic elements is vital for the further study of function and behaviour of the organisms. However it is challenging to separate these small sequences from longer chromosomes within a given species. Machine learning approaches have been successfully developed to classify assembled contigs into two classes (plasmids and chromosomes). However, such tools are not designed to directly perform classification on long and error-prone reads which have been gaining popularity in genomics studies. Assembling complete plasmids is still challenging for many long-read assemblers with a mixed input of long and error-prone reads from plasmids and chromosomes. In this paper, we present PlasLR, a tool that adapts existing plasmid detection approaches to directly classify long and error-prone reads. PlasLR makes use of both the composition and coverage information of long and error-prone reads. We evaluate PlasLR on multiple simulated and real long-read datasets with varying compositions of plasmids and chromosomes. Our experiments demonstrate that PlasLR substantially improves the accuracy of plasmid detection on top of the state-of-the-art plasmid detection tools. Moreover, we show that using PlasLR before long-read assembly helps to enhance the assembly quality in terms of plasmid recovery and near complete chromosome assembly from metagenomic datasets. The source code is freely available at `https://github.com/anuradhawick/PlasLR`.

Introduction

Plasmids are extra-chromosomal genetic elements which allow their hosts to rapidly adapt and survive under changing environmental conditions [1, 2]. These small and widespread genetic elements can be commonly found in bacterial cells [3]. Plasmids are responsible for a significant amount of genetic variation within populations which ensures persistence towards changes in the environment and various selective pressures. Plasmids also play an important role in horizontal gene transfer among unrelated bacterial species to spread genes related to virulence, resistance to different classes of antibiotics and heavy metal resistance [2, 4, 5]. Hence, it is important to identify and study plasmids present in environmental samples [5].

High-throughput sequencing such as next-generation sequencing (NGS) and third-generation sequencing (TGS) technologies have allowed us to sequence and analyse bacterial genomes including both their chromosomes and plasmids [6, 7]. However, when sequencing an entire sample, reads originating from chromosomes and plasmids are mixed together. Although NGS produces highly accurate reads, direct classification of NGS reads is not possible as their typical length (100-300bp) may not be long enough to contain sufficient genomic signatures [8]. Many tools such as plasmidSPAdes [9], Recycler [10], PLACNET [11] and PLACNETw [12] have been developed to directly reconstruct plasmid sequences from raw NGS reads. For example, plasmidSPAdes [9] makes use of the coverage information in the assembly graph to recover contigs originating from plasmids. Recycler [10] reconstructs plasmids as cycles in the assembly graph using coverage and length properties [10]. PLACNET [11] and PLACNETw [12] (a web tool based on PLACNET) create a network of contig interactions to reconstruct plasmids with the help of manual interaction.

Another interesting strategy to detect plasmids from NGS reads is to first assemble the short reads into much longer sequences called *contigs* and then classify these contigs. For example, cBar [13] and mlplasmids [14] both use k-mer profiles of sequences and different classification models to classify chromosomal and plasmid contigs. PlasmidFinder [15] is a web application that compares contigs with a database of replicon sequences to identify plasmids. More recently, two supervised-learning approaches, PlasFlow [16] and PlasClass [17], have achieved the best performance in classifying plasmid contigs found in metagenomics samples. Table 1 denotes a comparison of these tools to classify contigs (assembled from NGS reads) originating from plasmids or chromosomes.

Table 1. Comparison of currently available plasmid detection tools which classify plasmid and chromosomal contigs.

Tool	Inputs provided	Uses coverage	Uses sequence features	Method of classification
cBar [13]	Contigs	✗	✓	Decision tree, Bayes network, SVM, SMO, nearest neighbour
PlasmidFinder [15]	Contigs	✗	✓	Similarity comparison with a database of replicon sequences
mlplasmids [14]	Contigs	✗	✓	Logistic regression, Bayesian classifier, decision trees, RF, SVM
PlasFlow [16]	Contigs	✗	✓	Neural networks
PlasClass [17]	Contigs	✗	✓	Logistic regression

Third-generation sequencing (TGS) technologies such as Pacific Biosciences (PacBio) and Oxford Nanopore Technologies (ONT) have become popular because TGS reads are orders of magnitudes longer than NGS reads. As TGS reads are long enough to span over many repetitive regions, plasmid sequences have been reconstructed by assembling TGS reads [18–21]. However, recent benchmarks show that assembling complete plasmids is still challenging for many long-read assemblers from a mixed input of TGS reads from plasmids and chromosomes [22, 23]. As TGS reads are as long as contigs assembled from NGS reads, it becomes interesting to explore if it is possible to directly classify TGS reads originating from plasmids or chromosomes before assembly. Separating TGS reads before assembly has the potential to improve the assembly quality because it allows long-read assemblers to assemble plasmids and chromosomes independently (using different appropriate parameters). However, the above contig-classification tools (in Table 1) are designed to be used directly with NGS assemblies. Such contigs are usually accurate and have a higher base quality. However, TGS reads are much more error-prone compared to contigs obtained from NGS reads. Therefore, there remains a need to design and test plasmid classification tools directly for TGS reads.

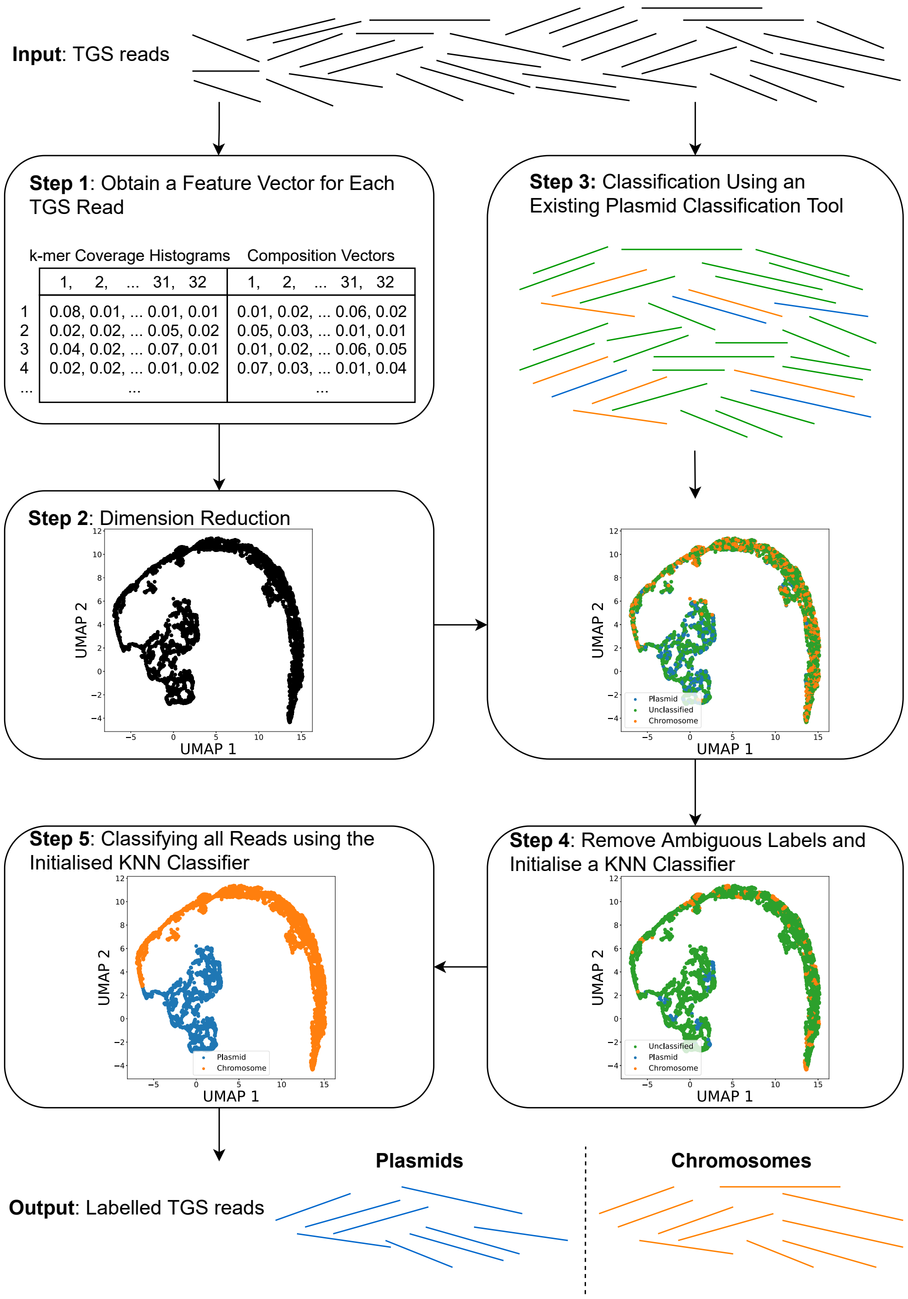
In this paper, we propose PlasLR, a tool that adapts existing contig-classification tools for plasmids to directly classify long and error-prone (TGS) reads. PlasLR makes use of both the composition information and k-mer coverage information of long reads. To the best of our knowledge, PlasLR is the first plasmid classification tool that is designed to directly handle long reads. Our experiments demonstrate that PlasLR achieves high accuracy of plasmid detection on top of state-of-the-art plasmid classification tools. Moreover, we show that using PlasLR before long-read assembly also helps to enhance the assembly quality.

Materials and methods

The contig-classification tools for plasmids (in Table 1) have been successfully applied to classify contigs assembled from NGS reads. However, they are not robust on direct classification of all TGS reads (by treating TGS reads as contigs) because TGS reads are more error-prone than contigs assembled from NGS reads. However, these tools are capable of producing a subset of confident classification at higher confident thresholds, *i.e.* chromosomes closer to probability 0 and plasmids closer to probability 1. Hence, there exists a significant compromise between precision and recall depending on the chosen threshold. Therefore, PlasLR employs such high confident classifications as initial results along with new features to make a semi-supervised classifier to classify error prone long reads.

PlasLR takes raw TGS reads (either PacBio or ONT) as the input. The minimum read length accepted by PlasLR is 1000bp. If we directly apply an existing contig-classification tool to classify all the long reads, many long reads may receive ambiguous or incorrect labels (plasmid or chromosome) as these long reads are not as accurate as the default input (contigs assembled from NGS reads). The intuition is to select and retain the most confident labels (plasmid or chromosome) from a subset of long reads and propagate these labels to other long reads originating from the same chromosome or plasmid. Previous studies have shown that plasmids and chromosomes have different oligonucleotide compositions which have been used in plasmid classification [13, 24–26]. The oligonucleotide composition is thus a potential feature for long reads because long reads from the same chromosome or plasmid tend to have similar oligonucleotide composition. Plasmids may also have different coverages compared to chromosomes and such coverage variations have been used in plasmid reconstruction [9]. Although it is challenging to estimate the coverage of a chromosome or plasmid that a long read belongs to, the k-mer coverage histogram of a long read has been shown to approximate such coverage and has been used in clustering long reads [27]. Therefore, PlasLR extracts both oligonucleotide composition and k-mer coverage histogram as features of a long read and propagates reliable labels from a subset of long reads to others based on these features.

More specifically, the complete workflow is presented in Figure 1. In Step 1, the k-mer coverage histograms and the trinucleotide frequency vectors are computed respectively and concatenated together for each read. In Step 2 the concatenated vectors are subjected to dimension reduction using Principal Component Analysis (PCA) [28], UAMP (default choice) [29] and OpenTSNE [30]. In Step 3, TGS reads are classified using an existing tool and only the high-confident labels (plasmid or chromosome) are retained. In Step 4, nearest neighbours in the UMAP plots are used to detect and remove some ambiguous labels, and a K-Nearest Neighbour (KNN) classifier [31] is initialised on the remaining labelled reads. Finally, in Step 5, all the unlabelled reads are labelled using the KNN classifier. Details of each step are explained in the following sections.


Input: TGS reads
Step 1: Obtain a Feature Vector for Each TGS Read
k-mer Coverage Histograms
Composition Vectors
Step 2: Dimension Reduction
Step 3: Classification Using an Existing Plasmid Classification Tool
Step 4: Remove Ambiguous Labels and Initialise a KNN Classifier
Step 5: Classifying all Reads using the Initialised KNN Classifier
UMAP 1
UMAP 2
Plasmid
Unclassified
Chromosome
Output: Labelled TGS reads
Plasmids
Chromosomes

	1, 2, ... 31, 32	1, 2, ... 31, 32
1	0.08, 0.01, ... 0.01, 0.01	0.01, 0.02, ... 0.06, 0.02
2	0.02, 0.02, ... 0.05, 0.02	0.05, 0.03, ... 0.01, 0.01
3	0.04, 0.02, ... 0.07, 0.01	0.01, 0.02, ... 0.06, 0.05
4	0.02, 0.02, ... 0.01, 0.02	0.07, 0.03, ... 0.01, 0.04
...	...	...

Fig 1. The Workflow of PlasLR. Long reads are provided as inputs for PlasLR. In Step 1, the k-mer coverage histograms and the trinucleotide frequency vectors are computed respectively and concatenated together for each read. In Step 2 the concatenated vectors are subjected to dimension reduction using UMAP. In Step 3, TGS reads are classified using an existing tool and only the high-confident labels (plasmid or chromosome) are retained. In Step 4, nearest neighbours in the UMAP plot are used to detect and remove some ambiguous labels, and a KNN classifier is initialised on the remaining labelled reads. Finally, in Step 5, all the unlabelled reads are labelled using the KNN classifier.

Step 1: Obtain a Feature Vector for Each TGS Read

In real-world genomic samples, the species (chromosomes and plasmids) are of varying abundances but contigs from the same chromosome or plasmid are likely to have similar coverages. In other words, the reads should overlap and stack on each other to reflect the actual coverages. Intuitively, this can be achieved using an all-vs-all alignment approach. However, such a naive approach would be computationally expensive as the number of reads can be very large. Therefore, a k-mer based approach is used in PlasLR for coverage estimation of long reads [27].

For a k-mer in a long read, the coverage of this k-mer is defined as its number of appearances in all the reads. Despite its high error rates, each long read may still contain some (unknown) error-free k-mers and the coverages of such k-mers correlate with the coverage of the chromosome or plasmid that the long read belongs to. As erroneous k-mers typically have lower coverages compared to error-free k-mers, the coverage histogram of all k-mers in a long read indicates the coverage of the chromosome or plasmid that it belongs to. Similar to previous studies on long read analysis [32–34], we chose k=15 in PlasLR such that there are enough error-free k-mers in most long reads to provide accurate coverage information while reducing k-mer collisions. The computation of k-mer coverage histograms starts with counting all the k-mers in the whole set of TGS reads by DSK [35]. The k-mer counts are then stored in a lookup table for efficient lookup. For each read, all its k-mers are converted to an array of counts by using the above lookup table. By default, PlasLR focuses on k-mers with coverage from 1 to 320 and a k-mer coverage histogram is generated by dividing the coverage range 1X to 320X into 32 bins, with each bin having a size of 10 (*i.e.*, [1-10],[11-20],...,[311-320]). The k-mers with counts more than 320 will be accumulated to the last bin. The choice of these variables are provided in PlasLR. This histogram is normalised by the total number of k-mers observed in the read. This 32-dimension vector is chosen as the first half of the feature vector.

Microbial genomes carry signatures of oligonucleotide frequencies that are unique to each species [36]. These signatures are preserved within a species across its genome and varies between different species [37]. Moreover, previous studies have shown that chromosomes and plasmids have different oligonucleotide compositions [24–26]. Given the longer lengths of TGS reads, despite their error prone nature, the reads carry closely similar oligonucleotide frequencies to that of their underlying chromosomes/plasmids. While PlasFlow and PlasClass use the composition information from 3-mers to 7-mers in contigs, PlasLR only uses 3-mers (trinucleotides or trimers) to compute frequency vectors for the long reads because higher error rates in long reads affect more k-mers as k increases. 3-mer frequency vectors have been successfully utilised in metagenomics binning of long reads [27]. For each long read, we count the occurrences of all 64 3-mers and then combine the counts of any 3-mer and its reverse complement. This results in a vector of size 32, which is then normalised by the sum of elements in the vector. Now, we concatenate the coverage histograms and trinucleotide frequency vectors to derive a vector with 64 dimensions for each long read as shown in Figure 1 Step 1.

Step 2: Dimension Reduction

UMAP (Uniform Manifold Approximation and Projection) [29] is used to project the concatenated vectors of Step 1 into the two-dimensional space. UMAP is a technique that is used to reduce the dimensionality of the data while preserving most of the variation among data points in the dataset. UMAP tries to embed data in lower dimensions such that the fuzzy topological structure is conserved [29]. Hence, PlasLR uses two UMAP components to visualise (refer to Step 2 of Figure 1) and classify long reads into the two classes.

Figure 2 denotes the UMAP plots of 64-dimension vectors of simulated PacBio reads originating from the *Aquifex aeolicus* chromosome (NCBI accession number NC_000918, 20x coverage) and the *Lactococcus lactis* plasmid (NCBI accession number NC_000906, 200x coverage) and **Sim-2C5P** dataset. We include the ground-truth information as coloured points in Figure 2 to demonstrate a locality of the chromosomal and plasmid sequences in the UMAP plot. Note that PlasLR provides users with the ability to select dimensionality reduction algorithm (PCA, UMAP, OpenTSNE [30]).

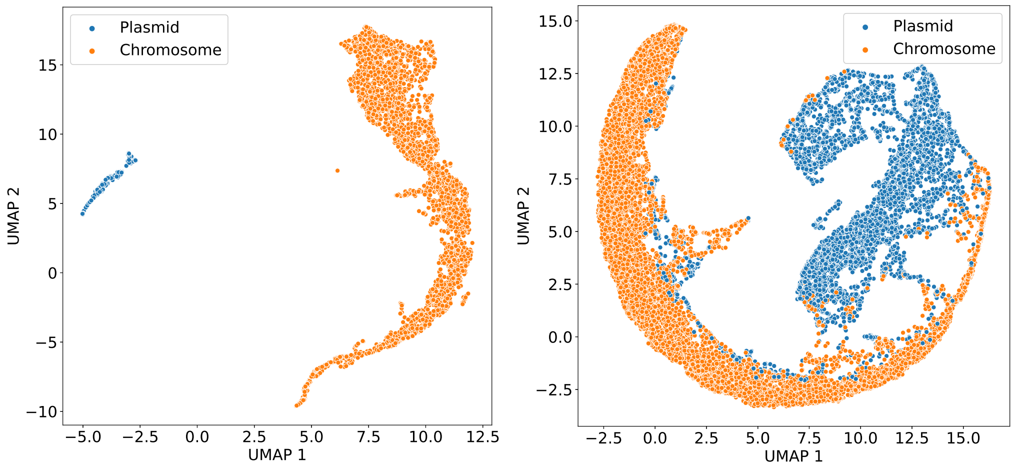


Fig 2. Example UMAP decomposition plots of concatenated coverage histograms and composition vectors. (a) single species with 1 chromosome and 1 plamid, (b) Sim-2C5P dataset. The UMAP decomposition plot of concatenated coverage histograms and 3-mer frequency vectors of a set of simulated PacBio reads originating from one *Aquifex aeolicus* chromosome (20X coverage) and one *Lactococcus lactis* plasmid (200X coverage) is shown on left. In the right, a **Sim-2C5P** dataset which is more complicated is demonstrated.

Step 3: Classification Using an Existing Plasmid Classification Tool

As discussed above, PlasLR makes use of plasmid classification results from existing contig-classifcation tools. In this paper, we have selected two of the latest plasmid detection tools, PlasFlow [16] and PlasClass [17] to obtain the initial classifications. PlasFlow and PlasClass are shown to outperform earlier tools in recent benchmarks [16,17] and provide confidence levels for the output labels predicted. This facilitates PlasLR to perform downstream tasks by filtering classifications based these confidence levels.

Note that the initial classification results of either PlasFlow or PlasClass consist of 3 classes, plasmid, chromosome and unclassified. Most (>50%) TGS reads will be in the unclassified class as only high-confident assignments of plasmids and chromosomes are selected. Refer to Section "Tools Used for Initial Classification" for more details on selecting high-confident assignments in both PlasFlow and PlasClass. The Step 3 of Figure 1 demonstrates the labels of plasmids and chromosomes incorporated into the two dimensional UMAP plot, and will be used in subsequent refinement and propagation.

Step 4: Remove Ambiguous Labels and Initialise a KNN Classifier

Using the set of labelled reads from Step 3, nearest neighbours (50 neighbours by default) are computed for each read using the lower dimensional representation. The label of a read (plasmid or chromosome) is defined to be *ambiguous* if more than 20% of its labelled nearest neighbours have a different label. PlasLR removes those ambiguous labels and set them to be unclassified. Then, a K-Nearest Neighbour (KNN) classifier [31] is initialised on the remaining labelled points (refer to Step 4 of Figure 1). In this step, the KNN indexes the labelled points thus enabling predictions on unseen points using the K (K=50) closest labelled data points.

Step 5: Classifying all Reads using the Initialised KNN Classifier

The initialised KNN classifier is used to predict the probability that a read is a plasmid. We label the reads exceeding 0.5 probability threshold as plasmids and vice versa. Step 5 of Figure 1 demonstrates the visualisation of assigning unclassified reads to the two classes using the KNN classifier. PlasLR finally classifies all the input TGS reads into two classes (plasmid and chromosome).

Experimental Setup

Datasets

We used the following TGS datasets in our experiments.

1. We simulated several datasets with genomic abundances obtained from a log normal distribution following the model in [17]. The plasmid copy numbers were taken from a geometric distribution where the probability of success is $min(1, log(L)/10)$ and L is the length of the plasmid. This was performed to amplify the abundance of short plasmids [17]. The following three datasets are simulated using SimLoRD [38]. The error rate was set to 10% and the minimum read length was set to 1000bp. The detailed information about species (chromosomes and plasmids) is available in Section 1 of the Supplementary Material.

    - **Sim-2C5P**: Contains 1 species with a total of 2 chromosomes and 5 plasmids.
    - **Sim-4C11P**: Contains 2 species with a total of 4 chromosomes and 11 plasmids.
    - **Sim-10C16P**: Contains 5 species with a total of 10 chromosomes and 16 plasmids.

2. Zymo GridION dataset (*Zymo-GridION-EVEN-BB-SN*) from the ZymoBIOMICS Microbial Community Standards [39] was used, which consists of real Oxford Nanopore (ONT) reads. In order to obtain datasets with a considerable portion of plasmid content, we subsample reads from each of the three species *Escherichia coli*, *Staphylococcus aureus* and *Salmonella enterica*. We refer to the three datasets as **Zymo-EC**, **Zymo-SA** and **Zymo-SE** respectively. We randomly subsampled upto 10,000 reads from each plasmid and chromosome class.

Table 2. Information about the datasets in the experiments.

Dataset	Read type	Total number of reads	Average read length (bp)	Number of plasmid reads	Number of chromosomal reads
Sim-2C5P	PacBio	20250	8246	6720	13530
Sim-4C11P	PacBio	119957	8200	35110	84847
Sim-10C16P	PacBio	199983	8200	32223	168760
Zymo-EC	ONT	207442	6836	6437	10000
Zymo-SA	ONT	375654	4058	10000	10000
Zymo-SE	ONT	199974	6785	2850	10000

Tools Used for Initial Classification

We choose PlasFlow and PlasClass as initial classifiers for plasmids and chromosomes since they have outperformed other tools in recent evaluations [16, 17].

PlasFlow outputs probabilities for 26 different classes under the phylum of each chromosome and plasmid. Furthermore, the results contain a single field labelling the classification in the form *Class.phylum*. Please refer to Section 2 of the Supplementary Material for more details.

PlasClass outputs single-valued probabilities to indicate how probable a sequence belongs to a plasmid. PlasLR chooses the thresholds such that 5% of reads are plasmids if such an amount of reads exists above 70% confidence. Similarly up to 20% reads are chosen to be plasmids below 50% confidence. The selection of probability thresholds is further discussed in Section 3 of the Supplementary Material.

Evaluation Criteria

Different classification tools (PlasFlow, PlasClass and PlasLR) are evaluated based on the precision, recall and F1 score of plasmid and chromosome classifications separately. For this purpose, we define the following terms.

- TP_p: the number of actual plasmid sequences that were classified as plasmid (true positives for plasmids)
- TP_c: the number of actual chromosomal sequences that were classified as chromosomal (true positives for chromosomes)
- FP_p: the number of non-plasmid sequences that were classified as plasmid (false positives for plasmids)
- FP_c: the number of non-chromosomal sequences that were classified as chromosomal (false positives for chromosomes)
- FN_p: the number of plasmid sequences that were not classified as plasmid (false negatives for plasmids)
- FN_c: the number of chromosomal sequences that were not classified as chromosomal (false negatives for chromosomes)

The micro-averaged precision, recall and F1 score are calculated as follows. Note that FN_p and FN_c include the number of unclassified chromosomal and plasmid sequences, respectively. We use the following equations to evaluate the classification of chromosomes.

$$Chromosome\ Precision\ (\%) = \frac{TP_c}{TP_c + FP_c} \tag{1}$$

$$Chromosome\ Recall\ (\%) = \frac{TP_c}{TP_c + FN_c} \tag{2}$$

For the evaluation of plasmids the following equations were used;

$$Plasmid\ Precision\ (\%) = \frac{TP_p}{TP_p + FP_p} \tag{3}$$

$$Plasmid\ Recall\ (\%) = \frac{TP_p}{TP_p + FN_p} \tag{4}$$

For both the classes we compute the F1-score using the following equation;

$$F1\ score\ (\%) = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{5}$$

For the reads in the real dataset (with unknown ground truth), the reads are mapped to the known reference genomes using Minimap 2 [40] and the chromosome or plasmid that results in the best alignment is considered as the origin of the read.

Results

Classification Results on Long Reads

We executed PlasFlow [16] and PlasClass [17] and plotted precision-vs-recall curves to examine the performance of each tool as the decision boundary changes. The resulting trade-offs between precision and recall are demonstrated in Figure 3 for the Zymo-SA dataset (details about this dataset can be found in Section 3.1). As expected, the higher the recall PlasFlow and PlasClass achieve (*i.e.*, classify more TGS reads), the lower the precision their classification results have. Hence, it is evident that mere parameter setting of a decision threshold is insufficient to improve the overall classification of results (Note that both PlasFlow and PlassClass provide a probability to assign each input sequence to either plasmids or chromosomes). When we select higher thresholds on such probabilities, both PlasFlow and PlassClass are able to generate relatively accurate classification results for a small subset of input sequences. In Figure 3, we demonstrates the PlasLR precision-vs-recall curves which demonstrate a significant shift of precision and recall from the initial results.

We present the results of PlasLR on classification of long reads into two classes (plasmid and chromosome). Table 3 denotes the comparison of results of PlasFlow [16], PlasClass [17] and PlasLR (on top of PlasFlow and PlasClass) respectively. Note that the probabilites given by PlasClass, PlasFlow and PlasLR are converted into respective classes by considering the decision boundary of 0.5.

According to the F1-scores highlighted in Table 3, we can see that PlasLR has improved the classification results based on the initial results obtained from PlasFlow and PlasClass. Similar to Figure 3, PlasLR has shown significant improvements over the recall while maintaining a high precision (similar to or even better than the precision of the high-confident subset of reads initially classified by PlasFlow or PlasClass). For initial classification results used in PlasLR, please refer to Section 4 of the Supplementary Material. The improvement on precision in PlasLR is gained by removing ambiguous labels while the improvement on recall of PlasLR is achieved by

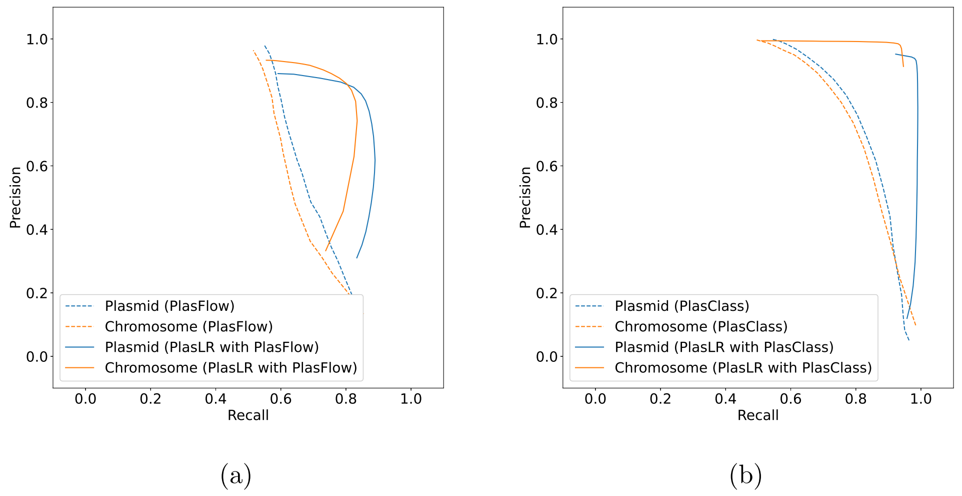


Fig 3. The trade-off between precision and recall for (a) PlasFlow results and for (b) PlasClass results on the Zymo-SA dataset. The precision-recall curve is generated by varying the probability threshold in PlasFlow and PlasClass, respectively. The classification results of PlasLR on top of PlasFlow and PlasClass achieve high precision and recall. Precision and recall are computed separately for plasmid and chromosomal sequences using $TP/(TP + FP)$ and $TP/(TP + FN)$ respectively. Unclassified reads are considered under FN for recall.

Table 3. Comparison of classification results of PlasFlow [16], PlasClass [17] and PlasLR (on top of PlasFlow and PlasClass). Please refer to Section 4 of the Supplementary Material for raw results presented in read counts used by PlasLR as input.

Dataset	Evaluation Criteria	**PlasFlow**		**PlasLR** with **PlasFlow** result		**PlasClass**		**PlasLR** with **PlasClass** result	
		Chromosome	Plasmid	Chromosome	Plasmid	Chromosome	Plasmid	Chromosome	Plasmid
Sim-2C5P	Precision	87.75%	73.50%	94.25%	75.28%	82.40%	56.76%	90.93%	61.30%
	Recall	26.84%	47.31%	85.40%	89.51%	24.75%	45.34%	73.25%	85.30%
	F1-Score	41.10%	57.56%	**89.61%**	**81.78%**	38.06%	50.41%	**81.14%**	**71.33%**
Sim-4C11P	Precision	93.94%	65.81%	94.62%	63.16%	94.66%	50.91%	94.17%	43.51%
	Recall	27.27%	54.71%	78.45%	89.22%	26.80%	61.58%	50.27%	92.48%
	F1-Score	42.28%	59.75%	**85.78%**	**73.97%**	41.77%	55.74%	**65.55%**	**59.18%**
Sim-10C16P	Precision	96.64%	24.45%	94.42%	33.73%	96.62%	31.49%	97.25%	24.77%
	Recall	23.28%	30.22%	72.01%	77.01%	23.66%	53.73%	47.91%	92.68%
	F1-Score	37.52%	27.03%	**81.71%**	**46.92%**	38.02%	**39.71%**	**64.20%**	39.09%
Zymo-EC	Precision	95.15%	80.79%	93.68%	81.22%	98.29%	89.26%	92.35%	91.30%
	Recall	32.95%	33.65%	86.46%	90.94%	34.59%	11.50%	94.61%	87.82%
	F1-Score	48.95%	47.51%	**89.93%**	**85.80%**	51.17%	20.37%	**93.47%**	**89.52%**
Zymo-SA	Precision	72.59%	79.82%	73.04%	88.50%	89.74%	94.96%	76.43%	98.96%
	Recall	31.23%	24.29%	90.25%	69.25%	39.38%	14.28%	99.18%	71.77%
	F1-Score	43.67%	37.25%	**80.74%**	**77.70%**	54.74%	24.82%	**86.33%**	**83.20%**
Zymo-SE	Precision	98.67%	70.80%	98.03%	69.71%	99.80%	94.78%	95.71%	91.72%
	Recall	26.79%	42.46%	88.39%	93.75%	29.21%	22.95%	97.59%	85.90%
	F1-Score	42.14%	53.08%	**92.96%**	**79.96%**	45.19%	36.95%	**96.64%**	**88.72%**

applying the initialised KNN classifier to unclassified reads on the UMAP plot. Although the precision drops in a few cases, PlasLR improves on the recall, F1 score and percentage of plasmid reads recovered on the datesets. Note that in dataset **Sim-10C5P** PlasLR has compromised a bit of plasmid F1-score in improving that of chromosomes. The initial results of dataset are relatively poor due to the mere complexity of the composition. However, the improvements on the Zymo datasets are significant due to the relative simplicity of the datasets.

Improved Assemblies on Long Reads

After classification of TGS reads into the two classes, we assembled the classified chromosomal and plasmid reads separately using metaFlye [41] (available in Flye [32]). metaFlye is a popular metagenomic long-read assembler which has shown best performance in assembling plasmids in a recent benchmark [22]. Moreover, we also assemble the complete set of TGS reads (without classification) using metaFlye as the baseline for comparison.

Table 4 demonstrates the assembly results for metaFlye, for all possible assembly approaches along with PlasLR results on top of both PlasFlow and PlasClass. The results show that PlasLR classification can improve the genome fraction (computed by MetaQUAST [42]) and the number of plasmids recovered by metaFlye. The chromosomal assemblies are significantly improved over the non-PlasLR approaches. Note that assembly performance gain is significant wherever there is a higher number of plasmids. Furthermore, poor chromosomal assembly in non-PlasLR approaches indicates the classification of chromosomal reads into plasmid class. This may result in plasmid assemblies whose contig set contain partial chromosomal assemblies. PlasLR mitigate this situation by pushing chromosomal and plasmid reads into appropriate bins facilitating proper assembly. This is evident in dataset **Sim-10C5P** where PlasLR with PlasFlow demonstrates the most number of sequences assembled (5 chromosomes and 13 plasmids). Note that, although genome fractions are comparable in some PlasLR results (especially in **Zymo-EC**, **Zymo-SA** and **Zymo-SE** datasets), each assembled class may contain multiple contigs that corresponds to the opposite class which might result in misleading downstream analysis. PlasLR supports the mitigation of such false positive plasmid/chromosome assemblies by accurate classification of long reads as indicated in Table 3 (refer to **Zymo-EC**, **Zymo-SA** and **Zymo-SE** datasets).

These improvements were achieved by classifying TGS reads before assembly, where metaFlye can assemble plasmids and chromosomes independently. The independent assembly of chromosomes and plasmids allows metaFlye to estimate more appropriate assembly parameters for plasmids and chromosomes, respectively.

Implementation

The source code for the experiments was implemented using Python 3.6.10 and C++ 9.3.0 (standard 17) and run on a Darwin system with macOS Catalina 10.15.3, 16G memory and Quad-Core Intel Core i7 CPU @ 2.5GHz with 4 CPU cores.

In order to reduce the memory consumption by the k-mers, each k-mer is encoded into its binary form using the encoding A=00, C=01, T=10 and G=11. During the computation of the k-mer coverage histograms, numeric values of the k-mers are stored as a lookup table with the number of times each k-mer occurs in the data set as the value for the k-mer.

PlasLR has the peak memory consumption during Step 1. The lookup table used in Step 1 utilises 4^{15} indices holding 64-bit (4-byte) double precision values. This results in a memory occupation of 8GB. All the other steps have a space complexity of $2 \times N \times 4$ bytes where N is the total number of reads. All steps of PlasLR are performed using

Table 4. Comparison of metaFlye assembly results with and without using PlasLR to classify TGS reads. The genome fractions are computed using MetaQUAST [42] while the number of plasmids recovered is the number of plasmids which have a genome fraction greater than 85%. Genome fractions are computed separately on the assemblies of plasmid and chromosome classes. Where genome fraction is defined as the total number of aligned bases as a percentage of genome size.

Dataset	Classification before assembly	metaFlye assembly			
		Genome fraction		Sequences recovered	
		Chromosome	Plasmid	Chromosome	Plasmid
Sim-2C5P	Raw reads	82.19%	96.21%	0	4
	PlasFlow	88.30%	97.81%	2	5
	PlasClass	65.64%	99.46%	0	5
	PlasLR with PlasFlow result	**95.99%**	**99.63%**	**2**	**5**
	PlasLR with PlasClass result	94.13%	99.40%	**2**	**5**
Sim-4C11P	Raw reads	**99.10%**	88.34%	**4**	8
	PlasFlow	73.83%	85.91%	1	9
	PlasClass	79.26%	85.56%	2	8
	PlasLR with PlasFlow result	93.20%	91.00%	3	**10**
	PlasLR with PlasClass result	96.19%	**92.08%**	**4**	9
Sim-10C16P	Raw reads	47.47%	39.63%	4	3
	PlasFlow	57.30%	**96.00%**	3	**13**
	PlasClass	64.04%	95.70%	3	**13**
	PlasLR with PlasFlow result	68.35%	95.88%	**5**	**13**
	PlasLR with PlasClass result	**86.41%**	75.50%	6	10
Zymo-EC	Raw reads	89.95%	100.00%	**1**	**1**
	PlasFlow	94.57%	100.00%	**1**	**1**
	PlasClass	98.82%	100.00%	**1**	**1**
	PlasLR with PlasFlow result	94.46%	100.00%	**1**	**1**
	PlasLR with PlasClass result	**99.00%**	**100.00%**	**1**	**1**
Zymo-SA	Raw reads	95.66%	66.63%	**1**	2
	PlasFlow	95.49%	99.96%	**1**	**3**
	PlasClass	97.51%	99.99%	**1**	**3**
	PlasLR with PlasFlow result	**98.52%**	**100.00%**	**1**	**3**
	PlasLR with PlasClass result	98.39%	99.93%	**1**	**3**
Zymo-SE	Raw reads	96.60%	**100.00%**	**1**	**1**
	PlasFlow	94.93%	**100.00%**	**1**	**1**
	PlasClass	99.58%	**100.00%**	**1**	**1**
	PlasLR with PlasFlow result	97.23%	**100.00%**	**1**	**1**
	PlasLR with PlasClass result	**99.70%**	**100.00%**	**1**	**1**

multi threading (8 threads by default). Our PlasLR implementation uses the python scikit-learn library implementation of the KNN classifier (known as *KNeighborsClassifier*) and PCA [43]. Furthermore, UMAP-learn [29] (used as the default dimensionality reducer) and OpenTSNE [30] were obtained from the repositories of respective authors of work.

Discussion and Conclusion

In this paper, we designed and evaluated PlasLR, a tool that adapts contig-classification tools for plasmids to TGS long reads. While existing contig-classification tools are able to accurately classify a subset of TGS long reads (by treating them as contigs), the use of PlasLR refines existing labels and extends them to all TGS reads. Moreover, we showed that the classification of long reads (into chromosomal and plasmid classes) before assembly improves the recovery of plasmids from both simulated and real metagenomic datasets.

The classification improvements are prevailing due to the combination of coverage and composition information, along with a set of initial classifications in a semi-supervised manner. Coverage information facilitates the discrimination of plasmids and chromosomes based on the copy number. Composition information predominantly enables the discrimination of sequences based on their origin species. The lower dimensional projection provides a spacial representation where the sequences with similar coverage and composition are closer, enabling the expansion of initial classification labels. Inevitably, the final results are influenced by the quality of the initial classifications. Therefore, the selection of high confident initial classification is vital for PlasLR to perform accurately. Similar to contig-based classifiers, PlasLR also faces challenges when classifying long reads from shared regions from plasmids and chromosomes. In such situations, the coverage information estimated by PlasLR may be misleading and thus result in possible mis-classifications.

As a matter of fact, PlasLR, like other tools evaluated, performs binary classification (i.e., plasmids v.s. chromosomes). A fine-grained classification is needed to provide distinctions between different plasmids and chromosomes and has the potential to be incorporated in the assembly process. Therefore, efforts will be made towards expanding the capability to detect individual plasmids and chromosomes by combining PlasLR with metagenomic binning methodologies. Furthermore, data-driven approaches will be investigated to replace empirically determined cutoffs used in ambiguous label removal. We also intend to benchmark PlasLR using different dimension-reduction techniques, initial classification tools and to extend PlasLR to perform standalone classification of plasmid sequences directly. We further consider the possibility of overlapped classification of long reads to improve its accuracy and scalability on more challenging metagenomic datasets with common sequences across different origins.

Acknowledgments

This research was undertaken with the assistance of resources and services from the National Computational Infrastructure (NCI), Australia. LP was supported by fellowships from the Edmond J. Safra Center for Bioinformatics at Tel-Aviv University, the postdoctoral fellowships from the Planning Budgeting Committee (PBC) of the Council for Higher Education (CHE) in Israel and the National Natural Science Foundation of China projects 61902072.

References

1. Thomas CM, Nielsen KM. Mechanisms of, and Barriers to, Horizontal Gene Transfer between Bacteria. Nature Reviews Microbiology. 2005;3(9):711–721. doi:10.1038/nrmicro1234.

2. Smalla K, Jechalke S, Top EM. Plasmid Detection, Characterization, and Ecology. Microbiology Spectrum. 2015;3(1).

3. Rodriguez-Beltran J, Hernandez-Beltran JCR, DelaFuente J, Escudero JA, Fuentes-Hernandez A, MacLean RC, et al. Multicopy plasmids allow bacteria to escape from fitness trade-offs during evolutionary innovation. Nature Ecology & Evolution. 2018;2(5):873–881. doi:10.1038/s41559-018-0529-z.

4. Carattoli A. Plasmids and the spread of resistance. International Journal of Medical Microbiology. 2013;303(6):298 – 304. doi:https://doi.org/10.1016/j.ijmm.2013.02.001.

5. Li AD, Li LG, Zhang T. Exploring antibiotic resistance genes and metal resistance genes in plasmid metagenomes from wastewater treatment plants. Frontiers in Microbiology. 2015;6:1025. doi:10.3389/fmicb.2015.01025.

6. Margos G, Hepner S, Mang C, Marosevic D, Reynolds SE, Krebs S, et al. Lost in plasmids: next generation sequencing and the complex genome of the tick-borne pathogen Borrelia burgdorferi. BMC Genomics. 2017;18(1):422. doi:10.1186/s12864-017-3804-5.

7. Forster SC, Kumar N, Anonye BO, Almeida A, Viciani E, Stares MD, et al. A human gut bacterial genome and culture collection for improved metagenomic analyses. Nature Biotechnology. 2019;37(2):186–192. doi:10.1038/s41587-018-0009-7.

8. Orlek A, Stoesser N, Anjum MF, Doumith M, Ellington MJ, Peto T, et al. Plasmid Classification in an Era of Whole-Genome Sequencing: Application in Studies of Antibiotic Resistance Epidemiology. Frontiers in Microbiology. 2017;8:182. doi:10.3389/fmicb.2017.00182.

9. Antipov D, Hartwick N, Shen M, Raiko M, Lapidus A, Pevzner PA. plasmidSPAdes: assembling plasmids from whole genome sequencing data. Bioinformatics. 2016;32(22):3380–3387. doi:10.1093/bioinformatics/btw493.

10. Rozov R, Brown Kav A, Bogumil D, Shterzer N, Halperin E, Mizrahi I, et al. Recycler: an algorithm for detecting plasmids from de novo assembly graphs. Bioinformatics. 2016;33(4):475–482. doi:10.1093/bioinformatics/btw651.

11. Lanza VF, de Toro M, Garcillán-Barcia MP, Mora A, Blanco J, Coque TM, et al. Plasmid Flux in Escherichia coli ST131 Sublineages, Analyzed by Plasmid Constellation Network (PLACNET), a New Method for Plasmid Reconstruction from Whole Genome Sequences. PLOS Genetics. 2014;10(12):1–21. doi:10.1371/journal.pgen.1004766.

12. Vielva L, de Toro M, Lanza VF, de la Cruz F. PLACNETw: a web-based tool for plasmid reconstruction from bacterial genomes. Bioinformatics. 2017;33(23):3796–3798. doi:10.1093/bioinformatics/btx462.

13. Zhou F, Xu Y. cBar: a computer program to distinguish plasmid-derived from chromosome-derived sequence fragments in metagenomics data. Bioinformatics. 2010;26(16):2051–2052. doi:10.1093/bioinformatics/btq299.

14. Arredondo-Alonso S, Rogers MRC, Braat JC, Verschuuren TD, Top J, Corander J, et al. mlplasmids: a user-friendly tool to predict plasmid- and chromosome-derived sequences for single species. Microbial Genomics. 2018;4(11). doi:https://doi.org/10.1099/mgen.0.000224.

15. Carattoli A, Zankari E, García-Fernández A, Voldby Larsen M, Lund O, Villa L, et al. In Silico Detection and Typing of Plasmids using PlasmidFinder and Plasmid Multilocus Sequence Typing. Antimicrobial Agents and Chemotherapy. 2014;58(7):3895–3903. doi:10.1128/AAC.02412-14.

16. Krawczyk PS, Lipinski L, Dziembowski A. PlasFlow: predicting plasmid sequences in metagenomic data using genome signatures. Nucleic Acids Research. 2018;46(6):e35–e35. doi:10.1093/nar/gkx1321.

17. Pellow D, Mizrahi I, Shamir R. PlasClass improves plasmid sequence classification. PLOS Computational Biology. 2020;16(4):1–9. doi:10.1371/journal.pcbi.1007781.

18. Loman NJ, Quick J, Simpson JT. A complete bacterial genome assembled de novo using only nanopore sequencing data. Nature Methods. 2015;12(8):733–735. doi:10.1038/nmeth.3444.

19. George S, Pankhurst L, Hubbard A, Votintseva A, Stoesser N, Sheppard AE, et al. Resolving plasmid structures in Enterobacteriaceae using the MinION nanopore sequencer: assessment of MinION and MinION/Illumina hybrid data assembly approaches. Microbial genomics. 2017;3(8):e000118–e000118. doi:10.1099/mgen.0.000118.

20. Lemon JK, Khil PP, Frank KM, Dekker JP. Rapid Nanopore Sequencing of Plasmids and Resistance Gene Detection in Clinical Isolates. Journal of Clinical Microbiology. 2017;55(12):3530–3543. doi:10.1128/JCM.01069-17.

21. Gochez AM, Huguet-Tapia JC, Minsavage GV, Shantaraj D, Jalan N, Strauß A, et al. Pacbio sequencing of copper-tolerant Xanthomonas citri reveals presence of a chimeric plasmid structure and provides insights into reassortment and shuffling of transcription activator-like effectors among X. citri strains. BMC Genomics. 2018;19(1):16. doi:10.1186/s12864-017-4408-9.

22. Wick R, Holt K. Benchmarking of long-read assemblers for prokaryote whole genome sequencing [version 1; peer review: 4 approved]. F1000Research. 2019;8(2138). doi:10.12688/f1000research.21782.1.

23. Gargis AS, Cherney B, Conley AB, McLaughlin HP, Sue D. Rapid Detection of Genetic Engineering, Structural Variation, and Antimicrobial Resistance Markers in Bacterial Biothreat Pathogens by Nanopore Sequencing. Scientific Reports. 2019;9(1):13501. doi:10.1038/s41598-019-49700-1.

24. Wong K, Finan TM, Golding BG. Dinucleotide compositional analysis of Sinorhizobium meliloti using the genome signature: distinguishing chromosomes and plasmids. Functional & Integrative Genomics. 2002;2(6):274–281. doi:10.1007/s10142-002-0068-0.

25. Zhou F, Olman V, Xu Y. Barcodes for genomes and applications. BMC Bioinformatics. 2008;9(1):546. doi:10.1186/1471-2105-9-546.

26. Davis JJ, Olsen GJ. Modal Codon Usage: Assessing the Typical Codon Usage of a Genome. Molecular Biology and Evolution. 2009;27(4):800–810. doi:10.1093/molbev/msp281.

27. Wickramarachchi A, Mallawaarachchi V, Rajan V, Lin Y. MetaBCC-LR: metagenomics binning by coverage and composition for long reads. Bioinformatics. 2020;36(Supplement_1):i3–i11. doi:10.1093/bioinformatics/btaa441.

28. Ringnér M. What is principal component analysis? Nature Biotechnology. 2008;26(3):303–304. doi:10.1038/nbt0308-303.

29. McInnes L, Healy J, Melville J. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction; 2020.

30. Poličar PG, Stražar M, Zupan B. openTSNE: a modular Python library for t-SNE dimensionality reduction and embedding. bioRxiv. 2019;doi:10.1101/731877.

31. Mitchell TM. Machine Learning. McGraw-Hill ScienceEngineeringMath; 1997.

32. Kolmogorov M, Yuan J, Lin Y, Pevzner PA. Assembly of long, error-prone reads using repeat graphs. Nature Biotechnology. 2019;37(5):540–546.

33. Lin Y, Yuan J, Kolmogorov M, et al. Assembly of long error-prone reads using de Bruijn graphs. Proceedings of the National Academy of Sciences. 2016;113(52):E8396–E8405.

34. Ruan J, Li H. Fast and accurate long-read assembly with wtdbg2. Nature Methods. 2020;17(2):155–158. doi:10.1038/s41592-019-0669-3.

35. Rizk G, Lavenier D, Chikhi R. DSK: k-mer counting with very low memory usage. Bioinformatics. 2013;29(5):652–653.

36. Abe T, Kanaya S, Kinouchi M, Ichiba Y, Kozuki T, Ikemura T. Informatics for Unveiling Hidden Genome Signatures. Genome Research. 2003;13(4):693–702.

37. Wu YW, Simmons BA, Singer SW. MaxBin 2.0: an automated binning algorithm to recover genomes from multiple metagenomic datasets. Bioinformatics. 2015;32(4):605–607.

38. Stöcker BK, Köster J, Rahmann S. SimLoRD: Simulation of Long Read Data. Bioinformatics. 2016;32(17):2704–2706.

39. Nicholls SM, Quick JC, Tang S, Loman NJ. Ultra-deep, long-read nanopore sequencing of mock microbial community standards. GigaScience. 2019;8(5).

40. Li H. Minimap2: pairwise alignment for nucleotide sequences. Bioinformatics. 2018;34(18):3094–3100.

41. Kolmogorov M, Bickhart DM, Behsaz B, Gurevich A, Rayko M, Shin SB, et al. metaFlye: scalable long-read metagenome assembly using repeat graphs. Nature Methods. 2020;17(11):1103–1110. doi:10.1038/s41592-020-00971-x.

42. Mikheenko A, Saveliev V, Gurevich A. MetaQUAST: evaluation of metagenome assemblies. Bioinformatics. 2015;32(7):1088–1090. doi:10.1093/bioinformatics/btv697.

43. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2011;12:2825–2830.