MitoChime: A Machine-Learning Pipeline for Detecting PCR-Induced Chimeras in Mitochondrial Illumina Reads

A Special Project Proposal
Presented to
the Faculty of the Division of Physical Sciences and Mathematics
College of Arts and Sciences
University of the Philippines Visayas
Miag-ao, Iloilo

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science in Computer Science

by

Duranne Duran
Yvonne Lin
Daniella Pailden

Adviser
Francis D. Dimzon, Ph.D.

December 5, 2025

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The rapid advancement of next-generation sequencing (NGS) technologies has transformed genomic research by enabling high-throughput and cost-effective DNA analysis (Metzker, 2010). Among current platforms, Illumina sequencing remains the most widely adopted, capable of producing millions of short reads that can be assembled into reference genomes or analyzed for genetic variation (Bentley et al., 2008; Glenn, 2011). Despite its high base-calling accuracy, Illumina sequencing is prone to artifacts introduced during library preparation, particularly polymerase chain reaction (PCR)-induced chimeras, which are artificial hybrid sequences that do not exist in the true genome (Judo, Wedel, & Wilson, 1998).

PCR chimeras form when incomplete extension products from one template anneal to an unrelated DNA fragment and are extended, creating recombinant reads (Qiu et al., 2001). In mitochondrial genome assembly, such artifacts are especially problematic because the mitochondrial genome is small, circular, and often repetitive (Boore, 1999; Cameron, 2014). Even a small number of chimeric or misjoined reads can reduce assembly contiguity and introduce false junctions during organelle genome reconstruction (Dierckxsens, Mardulyn, & Smits, 2017; Hahn, Bachmann, & Chevreux, 2013; Jin et al., 2020). Existing assembly tools such as GetOrganelle and MITObim assume that input reads are largely free of such artifacts (Hahn et al., 2013; Jin et al., 2020). Consequently, undetected chimeras may produce fragmented assemblies or misidentified organellar boundaries. To ensure accurate reconstruction of mitochondrial genomes, a reliable method for detecting and filtering PCR-induced chimeras before assembly is essential.

This study focuses on mitochondrial sequencing data from the genus *Sardinella*, a group of small pelagic fishes widely distributed in Philippine waters. Among them, *Sardinella lemuru* (Bali sardinella) is one of the country's most abundant and economically important species, providing protein and livelihood to coastal communities (Labrador, Agmata, Palermo, Ravago-Gotanco, & Pante, 2021; Willette, Bognot, Mutia, & Santos, 2011). Accurate mitochondrial assemblies are critical for understanding its population genetics, stock structure, and evolutionary history. However, assembly pipelines often encounter errors or fail to complete due to undetected chimeric reads. To address this gap, this research introduces MitoChime, a machine learning pipeline designed to detect and filter PCR-induced chimeric reads using both alignment-based and sequence-derived statistical features. The tool aims to provide bioinformatics laboratories, particularly the Philippine Genome Center Visayas (PGC Visayas), with an efficient solution for improving mitochondrial genome reconstruction.

1.2 Problem Statement

While NGS technologies have revolutionized genomic data acquisition, the accuracy of mitochondrial genome assembly remains limited by artifacts produced during PCR amplification. These chimeric reads can distort assembly graphs and cause misassemblies, with particularly severe effects in small, circular mitochondrial genomes (Boore, 1999; Cameron, 2014). Existing assembly pipelines such as GetOrganelle, MITObim, and NOVOPlasty assume that sequencing reads are free of such artifacts (Dierckxsens et al., 2017; Hahn et al., 2013; Jin et al., 2020). At PGC Visayas, several mitochondrial assemblies have failed or yielded incomplete contigs despite sufficient coverage, suggesting that undetected chimeric reads compromise assembly reliability. Meanwhile, existing chimera detection tools such as UCHIME and VSEARCH were developed primarily for amplicon-based community analysis and rely heavily on reference or taxonomic comparisons (Edgar, Haas, Clemente, Quince, & Knight, 2011; Rognes, Flouri, Nichols, Quince, & Mahé, 2016). These approaches are unsuitable for single-species organellar data, where complete reference genomes are often unavailable. Therefore, there is a pressing need for a reference-independent, data-driven tool capable of detecting and filtering PCR-induced chimeras in mitochondrial sequencing datasets.

1.3 Research Objectives

1.3.1 General Objective

This study aims to develop and evaluate a machine learning-based pipeline (MitoChime) that detects PCR-induced chimeric reads in *Sardinella lemuru* mitochondrial sequencing data in order to improve the quality and reliability of downstream mitochondrial genome assemblies.

1.3.2 Specific Objectives

Specifically, the study aims to:

1. construct simulated *Sardinella lemuru* Illumina paired-end datasets containing both clean and PCR-induced chimeric reads,
2. extract alignment-based and sequence-based features such as k-mer composition, junction complexity, and split-alignment counts from both clean and chimeric reads,
3. train, validate, and compare supervised machine-learning models for classifying reads as clean or chimeric,
4. determine feature importance and identify indicators of PCR-induced chimerism,
5. integrate the optimized classifier into a modular and interpretable pipeline deployable on standard computing environments at PGC Visayas.

1.4 Scope and Limitations of the Research

This study focuses on detecting PCR-induced chimeric reads in Illumina paired-end mitochondrial sequencing data from *Sardinella lemuru*. The decision to restrict the taxonomic scope to a single species is based on four considerations: to limit interspecific variation in mitochondrial genome size, GC content, and repetitive regions so that differences in read patterns can be attributed more directly to PCR-induced chimerism; to align the analysis with relevant *S. lemuru* sequencing projects at PGC Visayas; to take advantage of the availability of *S. lemuru* mitochondrial assemblies and raw datasets in public repositories such as the National Center for Biotechnology Information (NCBI), which facilitates reference selection and benchmarking; and to develop a tool that directly supports local studies on *S. lemuru* population structure and fisheries management.

The study emphasizes `wgsim`-based simulations and selected empirical mitochondrial datasets from *S. lemuru*. It excludes naturally occurring chimeras, nuclear mitochondrial pseudogenes (NUMTs), and large-scale assembly rearrangements in nuclear genomes. Feature extraction is restricted to low-dimensional alignment and sequence statistics, such as k-mer frequency profiles, GC content, read length, soft and hard clipping metrics, split-alignment counts, and mapping quality, rather than high-dimensional deep learning embeddings. This design keeps model behaviour interpretable and ensures that the pipeline can be run on standard workstations at PGC Visayas. Testing on long-read platforms (e.g., Nanopore, PacBio) and other taxa is outside the scope of this project; the implemented pipeline is evaluated only on short-read *S. lemuru* datasets.

Other limitations in this study include the following: simulations with varying error rates were not performed, so the effect of different sequencing errors on model performance remains unexplored; alternative parameter settings, including k-mer lengths and microhomology window sizes, were not systematically tested, which could affect the sensitivity of both k-mer and microhomology feature detection as well as the identification of chimeric junctions; and the machine-learning models rely on supervised training with labeled examples, which may limit their ability to detect novel or unexpected chimeric patterns.

1.5 Significance of the Research

This research provides both methodological and practical contributions to mitochondrial genomics and bioinformatics. First, MitoChime detects PCR-induced chimeric reads prior to genome assembly, with the goal of improving the contiguity and correctness of *Sardinella lemuru* mitochondrial assemblies. Second, it replaces informal manual curation with a documented workflow, improving automation and reproducibility. Third, the pipeline is designed to run on computing infrastructures commonly available in regional laboratories, enabling routine use at facilities such as PGC Visayas. Finally, more reliable mitochondrial assemblies for *S. lemuru* provide a stronger basis for downstream applications in the field of fisheries and genomics.

Chapter 2

Review of Related Literature

This chapter presents an overview of the literature relevant to the study. It discusses the biological and computational foundations underlying mitochondrial genome analysis and assembly, as well as existing tools, algorithms, and techniques related to chimera detection and genome quality assessment. The chapter aims to highlight the strengths, limitations, and research gaps in current approaches that motivate the development of the present study.

2.1 The Mitochondrial Genome

Mitochondrial genome (mtDNA) is a small, typically circular molecule found in most eukaryotes. It encodes essential genes involved in oxidative phosphorylation and energy metabolism. Because of its conserved structure, mtDNA has become a valuable genetic marker for studies in population genetics and phylogenetics (Anderson et al., 1981; Boore, 1999). In animal species, the mitochondrial genome ranges from 15–20 kilobase and contains 13 protein-coding genes, 22 tRNAs, and two rRNAs arranged compactly without introns (Gray, 2012). In comparison to nuclear DNA, the ratio of the number of copies of mtDNA is higher and has simple organization which make it particularly suitable for genome sequencing and assembly studies (Dierckxsens et al., 2017).

2.1.1 Mitochondrial Genome Assembly

Mitochondrial genome assembly refers to the reconstruction of the complete mitochondrial DNA (mtDNA) sequence from raw or fragmented sequencing reads. It is conducted to obtain high-quality, continuous representations of the mitochondrial genome that can be used for a wide range of analyses, including species identification, phylogenetic reconstruction, evolutionary studies, and investigations of mitochondrial diseases. Because mtDNA evolves rapidly, its assembled sequence provides valuable insights into population structure, lineage divergence, and adaptive evolution across taxa (Boore, 1999). Compared to nuclear genome assembly, assembling the mitochondrial genome is often considered more straightforward but still encounters technical challenges such as the formation of chimeric reads. Commonly used tools for mitogenome assembly such as GetOrganelle and MITObim operate under the assumption of organelle genome circularity, and are vulnerable when chimeric reads disrupt this circular structure, resulting in assembly errors (Hahn et al., 2013; Jin et al., 2020).

2.2 PCR Amplification and Chimera Formation

PCR plays an important role in NGS library preparation, as it amplifies target DNA fragments for downstream analysis. However as previously mentioned, the amplification process can also introduce chimeric reads which compromises the quality of the input reads supplied to sequencing or assembly workflows. Chimeras typically arise when incomplete extension occurs during a PCR cycle. This causes the DNA polymerase to switch from one template to another and generate hybrid recombinant molecules (Judo et al., 1998). Artificial chimeras are produced through such amplification errors, whereas biological chimeras occur naturally through genomic rearrangements or transcriptional events.

In the context of amplicon-based sequencing, the presence of chimeras can inflate estimates of genetic or microbial diversity and may cause misassemblies during genome reconstruction. Qin et al. (2023) has reported that chimeric sequences may account for more than 10% of raw reads in amplicon datasets. This artifact tends to be most prominent among rare operational taxonomic units (OTUs) or singletons, which are sometimes misinterpreted as novel diversity, further causing the complication of microbial diversity analyses (Gonzalez, Zimmermann, & Saiz-Jimenez, 2004). As such, determining and minimizing PCR-induced chimera formation is vital for improving the quality of mitochondrial genome assemblies, and ensuring the reliability of amplicon sequencing data.

2.3 Existing Traditional Approaches for Chimera Detection

Several computational tools have been developed to identify chimeric sequences in NGS datasets. These tools generally fall into two categories: reference-based and de novo approaches. Reference-based chimera detection, also known as database-dependent detection, is one of the earliest and most widely used computational strategies for identifying chimeric sequences in amplicon-based community studies. These methods rely on the comparison of each query sequence against a curated, high-quality database of known, non-chimeric reference sequences (Edgar et al., 2011).

On the other hand, the de novo chimera detection, also referred to as reference-free detection, represents an alternative computational paradigm that identifies chimeric sequences without reliance on external reference databases. This method infer chimeras based on internal relationships among the sequences present within the dataset itself, making it particularly advantageous in studies of under explored or taxonomically diverse communities where comprehensive reference databases are unavailable or incomplete (Edgar, 2016; Edgar et al., 2011). The underlying assumption on this method is that during PCR, true biological sequences are generally more abundant as they are amplified early and dominate the read pool, whereas chimeric sequences appear later and are generally less abundant. The de novo approach leverage this abundance hierarchy, treating the most abundant sequences as supposed parents and testing whether less abundant sequences can be reconstructed as mosaics of these templates. Compositional and structural similarity are also evaluated to check whether different regions of a candidate sequence correspond to distinct high-abundance sequences.

In practice, many modern bioinformatics pipelines combine both paradigms sequentially: an initial de novo step identifies dataset-specific chimeras, followed by a reference-based pass that removes remaining artifacts relative to established databases (Edgar, 2016). These two methods of detection form the foundation of tools such as UCHIME and later UCHIME2.

2.3.1 UCHIME

UCHIME is one of the most widely used computational tools for detecting chimeric sequences in amplicon sequencing data, as it serves as a critical quality control step to prevent the misinterpretation of PCR artifacts as novel biological diversity. The algorithm operates by searching for a model (M) where a query (Q) sequence can be perfectly explained as a combination of two parent sequences, denoted as A and B (Edgar et al., 2011).

In reference mode, UCHIME divides the query into four chunks and maps them to a trusted chimeric-free database to identify candidate parents. It then constructs a three-way alignment to calculate a score based on "votes." A "Yes" vote indicates the query aligns with parent A in one region and parent B in another, while a "No" vote penalizes the score if the query diverges from the expected chimeric model. In de novo mode, the algorithm operationalizes the abundance skew principle described in Section 2.3. Instead of using an external database, UCHIME dynamically treats the sample's own high-abundance sequences as a reference database, testing if lower-abundance sequences can be reconstructed as mosaics of these internal ancestors (Edgar et al., 2011).

Despite its high sensitivity, UCHIME has inherent limitations rooted in sequence divergence and database quality. The algorithm struggles to detect chimeras formed from parents that are very closely related, specifically when the sequence divergence between parents is less than roughly 0.8%, as the signal-to-noise ratio becomes too low to distinguish a crossover event from sequencing error (Edgar et al., 2011). Furthermore, in reference mode, the accuracy is strictly bound by the completeness of the database; if true parents are absent, the tool may fail to identify the chimera or produce false positives. Similarly, the de novo mode relies on the assumption that parents are present and sufficiently more abundant in the sample, which may not hold true in unevenly amplified samples or complex communities.

2.3.2 UCHIME2

Building upon the original algorithm, UCHIME2 was developed to address the nuances of high-resolution amplicon sequencing. A key contribution of the UCHIME2 study was the critical re-evaluation of chimera detection benchmarks. In the UCHIME2 paper (Edgar, 2016) and the UCHIME in practice website (Edgar, n.d), the author has noted that the accuracy results reported in the original UCHIME paper were “highly over-optimistic” because they relied on unrealistic benchmark designs where parent sequences were assumed to be 100% known and present. UCHIME2 introduced more rigorous testing (the CHSIMA benchmark), revealing that “fake models,” where a valid biological sequence perfectly mimics a chimera of two other valid sequences, are far more common than previously assumed. This discovery suggests that error-free detection is impossible in principle (Edgar, 2016).Another notable improvement is the introduction of multiple application-specific modes that allow users to tailor the algorithm's performance to the characteristics of their datasets. The following parameter presets: denoised, balanced, sensitive, specific, and high-confidence, enable researchers to optimize the balance between sensitivity and specificity according to the goals of their analysis.

However despite these advancements, the practical application of UCHIME2 requires caution. The author explicitly advises against using UCHIME2 as a stand-alone tool in standard OTU clustering or denoising pipelines. Using UCHIME2 as an independent filtering step in these workflows is discouraged, as it often results in significantly higher error rates, increasing both false positives (discarding valid sequences) and false negatives (retaining chimeras) (Edgar, 2016).

2.3.3 CATch

As previously mentioned, UCHIME (Edgar et al., 2011) relied on alignment-based sequences in amplicon data. However, researchers soon observed that different algorithms often produced inconsistent predictions. A sequence might be identified as chimeric by one tool but classified as non-chimeric by another, resulting in unreliable filtering outcomes across studies.

To address these inconsistencies, Mysara, Saeys, Leys, Raes, and Monsieurs (2015) developed the Classifier for Amplicon Tool Chimeras (CATCh), which represents the first ensemble machine learning system designed for chimera detection in 16S rRNA amplicon sequencing. Rather than depending on a single detection strategy, CATCh integrates the outputs of several established tools, including UCHIME, ChimeraSlayer, DECIPHER, Pintail, and Perseus. The individual scores and binary decisions generated by these tools are used as input features for a supervised learning model. The algorithm employs a Support Vector Machine (SVM) with a Pearson VII Universal Kernel (PUK) to determine optimal weightings among the input features and to assign each sequence a probability of being chimeric.

Benchmarking in both reference-based and de novo modes demonstrated significant performance improvements. CATCh achieved sensitivities of approximately 85 percent in reference-based mode and 92 percent in de novo mode, with corresponding specificities of approximately 96 percent and 95 percent. These results indicate that CATCh detected 7 to 12 percent more chimeras than any individual algorithm while maintaining high precision.

2.3.4 ChimPipe

Among the available tools for chimera detection, ChimPipe is a pipeline developed to identify chimeric sequences such as biological chimeras. It uses both discordant paired-end reads and split-read alignments to improve the accuracy and sensitivity of detecting biological chimeras (Rodriguez-Martin et al., 2017). By combining these two sources of information, ChimPipe achieves better precision than methods that depend on a single type of indicator.

The pipeline works with many eukaryotic species that have available genome and annotation data (Rodriguez-Martin et al., 2017). It can also predict multiple isoforms for each gene pair and identify breakpoint coordinates that are useful for reconstructing and verifying chimeric transcripts. Tests using both simulated and real datasets have shown that ChimPipe maintains high accuracy and reliable performance.

ChimPipe lets users adjust parameters to fit different sequencing protocols or organism characteristics. Experimental results have confirmed that many chimeric transcripts detected by the tool correspond to functional fusion proteins, demonstrating its utility for understanding chimera biology and its potential applications in disease research (Rodriguez-Martin et al., 2017).

2.4 Machine Learning Approaches for Chimera and Sequence Quality Detection

Traditional chimera detection tools rely primarily on heuristic or alignment-based rules. Recent advances in machine learning (ML) have demonstrated that models trained on sequence-derived features can effectively capture compositional and structural patterns in biological sequences. Although most existing ML systems such as those used for antibiotic resistance prediction, taxonomic classification, or viral identification are not specifically designed for chimera detection, they highlight how data-driven models can outperform similarity-based heuristics by learning intrinsic sequence signatures. In principle, ML frameworks can integrate indicators such as k-mer frequencies, GC-content variation and split-alignment metrics to identify subtle anomalies that may indicate a chimeric origin (Arango et al., 2018; Liang, Bible, Liu, Zou, & Wei, 2020; Ren et al., 2020).

2.4.1 Feature-Based Representations of Genomic Sequences

Feature extraction converts DNA sequences into numerical representations suitable for machine-learning models. One approach is k-mer frequency analysis, which counts short nucleotide sequences within a read (Vervier, Mahé, Tournoud, Veyrieras, & Vert, 2015). High-frequency k-mers, including simple repeats such as “AAAAAA,” can highlight repetitive or unusual regions that may occur near chimeric junctions. Comparing k-mer patterns across adjacent parts of a read can help identify such regions, while GC content provides an additional descriptor of local sequence composition (Ren et al., 2020).

Alignment-derived features further inform junction detection. Long-read tools such as Sniffles (Sedlazeck et al., 2018) use split alignments to locate breakpoints across extended sequences, whereas short-read aligners like Minimap2 (Li, 2018) report supplementary and secondary alignments that indicate local discontinuities. Split alignments, where parts of a read map to different regions, can reveal template-switching events. These features complement k-mer profiles and enhance detection of potentially chimeric reads, even in datasets with incomplete references.

Microhomology, or short sequences shared between adjacent segments, is another biologically meaningful feature. Its length, typically a few to tens of base pairs, has been linked to microhomology-mediated repair and template-switching mechanisms (Sfeir & Symington, 2015). In PCR-induced chimeras, short identical sequences at junctions provide a clear signature of chimerism. Measuring the longest exact overlap at each breakpoint complements k-mer and alignment features and helps identify reads that are potentially chimeric.

2.5 Synthesis of Chimera Detection Approaches

To provide an integrated overview of the literature discussed in this chapter, Table 2.1 summarizes the major chimera detection studies, their methodological approaches, and their known limitations.

Table 2.1: Comparison of Chimera Detection Methods

Methods	Approach	Limitations
Reference-based Chimera Detection	Compares query sequences against curated, non-chimeric reference databases; identifies mosaic sequences by evaluating similarity to known templates.	Depends heavily on completeness and quality of reference databases; often fails when novel taxa or missing parent sequences are present; reduced accuracy for low-divergence chimeras.
De novo Chimera Detection	Identifies chimeras using only internal dataset relationships; relies on abundance patterns and compositional similarity; reconstructs sequences as mosaics of high-abundance parents.	Assumes true sequences are more abundant—fails when amplification bias distorts abundance; struggles with evenly abundant parental sequences; can misclassify highly similar true variants.
UCHIME	Alignment-based chimera detection; segments query sequence, identifies parent candidates, performs 3-way alignment, and computes chimera scores; supports both reference-based and de novo modes.	Accuracy inflated in original benchmarks; suffers under incomplete databases; poor performance on low-divergence chimeras; sensitive to sequencing errors; misclassifies when parents are missing.
UCHIME2	Improved initial UCHIME benchmarking; offers multiple sensitivity/specificity modes; more robust with incomplete references; higher sensitivity.	Cannot achieve perfect accuracy due to "perfect fake models"; genuine variants may be indistinguishable from artificial recombinants; theoretical detection limit remains.
CATCh	First ML ensemble tool for 16S chimera detection; integrates outputs of UCHIME, ChimeraSlayer, DECIPHER, Pintail, Perseus via SVM classifier; significantly improves sensitivity and specificity.	Depends on performance of underlying tools; ML model limited to features they output; ensemble can still misclassify in datasets with extreme novelty or low coverage.
ChimPipe	Pipeline for detecting fusion genes and transcript-derived chimeras in RNA-seq; uses discordant paired-end reads and split-alignments; predicts isoforms and breakpoint coordinates.	Designed for RNA-seq, not amplicons; needs high-quality genome and annotation; computationally heavier; limited to organisms with reference genomes.

Across existing studies, no single approach reliably detects all forms of chimeric sequences, particularly those generated by PCR-induced template switching in mitochondrial genomes. Reference-based tools perform poorly when parental sequences are absent; de novo methods rely strongly on abundance assumptions; alignment-based systems show reduced sensitivity to low-divergence chimeras; and ensemble methods inherit the limitations of their component algorithms. RNA-seq–oriented pipelines likewise do not generalize well to organelle data. Although machine learning approaches offer promising feature-based detection, they are rarely applied to mitochondrial genomes and are not trained specifically on PCR-induced organelle chimeras. These limitations indicate a clear research gap: the need for a specialized, feature-driven classifier tailored to mitochondrial PCR-induced chimeras that integrates k-mer composition, split-alignment signals, and microhomology features to achieve more accurate detection than current heuristic or alignment-based tools.

Chapter 3

Research Methodology

This chapter outlines the steps involved in completing the study, including data gathering, generating simulated mitochondrial Illumina reads, preprocessing and indexing the data, developing a feature extraction pipeline to extract key features, applying machine learning algorithms for chimera detection, and validating and comparing model performance.

3.1 Research Activities

As illustrated in Figure 3.1, this study carried out a sequence of procedures to detect PCR-induced chimeric reads in mitochondrial genomes. The process began with collecting a mitochondrial reference sequence of *Sardinella lemuru* from the National Center for Biotechnology Information (NCBI) database, which was used as a reference for generating simulated clean and chimeric reads. These reads were subsequently indexed and mapped. The resulting collections then passed through a feature extraction pipeline that extracted k-mer profiles, supplementary alignment (SA) features, and microhomology information to prepare the data for model construction. The machine learning model was trained using the processed input, and its precision and accuracy were assessed. It underwent tuning until it reached the desired performance threshold, after which it proceeded to validation and will undergo testing.

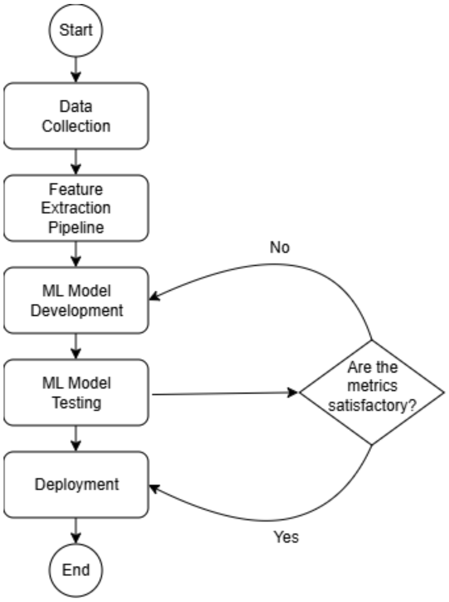


Figure 3.1: Process Diagram of Special Project

3.1.1 Data Collection

The mitochondrial genome reference sequence of *S. lemuru* was obtained from the NCBI database (accession number NC_039553.1) in FASTA format. This sequence served as the basis for generating simulated reads for model development.

This step was scheduled to begin in the first week of November 2025 and expected to be completed by the end of that week, with a total duration of approximately one (1) week.

Data Preprocessing

To reduce manual repetition, all steps in the simulation and preprocessing pipeline were executed using a custom script in Python (Version 3.11). The script runs each stage, including read simulation, reference indexing, mapping, and alignment processing, in a fixed sequence.

Sequencing data were simulated from the NCBI reference genome using `wgsim` (Version 1.13). First, a total of 10,000 paired-end fragments were simulated, producing 20,000 reads (10,000 forward and 10,000 reverse) from the the original reference (`original_reference.fasta`) and and designated as clean reads using the command:

```
wgsim -1 150 -2 150 -r 0 -R 0 -X 0 -e 0.001 -N 10000 \
       original_reference.fasta ref1.fastq ref2.fastq
```

The command parameters are as follows:

- `-1` and `-2`: read lengths of 150 base pairs for each paired-end read.
- `-r`, `-R`, `-X`: mutation rate, fraction of indels, and indel extension probability, all set to a default value of 0.
- `-e`: base error rate, set to 0.001 to simulate realistic sequencing errors.
- `-N`: number of read pairs, set to 10,000.

Chimeric sequences were then generated from the same NCBI reference using a separate Python script. Two non-adjacent segments were randomly selected such that their midpoint distances fell within specified minimum and maximum thresholds. The script attempts to retain microhomology, or short identical sequences at segment junctions, to mimic PCR-induced template switching. The resulting chimeras were written to `chimera_reference.fasta`, with headers recording segment positions and microhomology length. The `chimera_reference.fasta` was processed with `wgsim` to simulate 10,000 paired-end fragments, generating 20,000 chimeric reads (10,000 forward reads in `chimeric1.fastq` and 10,000 reverse reads in `chimeric2.fastq`) using the command format.

Next, a `minimap2` index of the reference genome was created using:

```
minimap2 -d ref.mmi original_reference.fasta
```

Minimap2 (Version 2.28) is a tool used to map reads to a reference genome. The index `ref.mmi` of the original reference sequence is required by `minimap2` for efficient read mapping. Mapping allows extraction of alignment features from each read, which were used as input for the machine learning model. The simulated clean and chimeric reads were then mapped to the reference index as follows:

```
minimap2 -ax sr -t 8 ref.mmi ref1.fastq ref2.fastq > clean.sam
```

```
minimap2 -ax sr -t 8 ref.mmi \
chimeric1.fastq chimeric2.fastq > chimeric.sam
```

Here, `-ax sr` specifies short-read alignment mode, and `-t 8` uses 8 CPU threads. The resulting clean and chimeric SAM files contain the alignment positions of each read relative to the original reference genome.

The SAM files were then converted to BAM format, sorted, and indexed using `samtools` (Version 1.20):

```
samtools view -bS clean.sam -o clean.bam
samtools view -bS chimeric.sam -o chimeric.bam

samtools sort clean.bam -o clean.sorted.bam
samtools index clean.sorted.bam

samtools sort chimeric.bam -o chimeric.sorted.bam
samtools index chimeric.sorted.bam
```

BAM files are the compressed binary version of SAM files, which enables faster processing and reduced storage. Sorting arranges reads by genomic coordinates, and indexing allows detection of SA as a feature for the machine learning model.

The total number of simulated reads was expected to be 40,000. The final collection of reads contained 19,984 clean reads and 20,000 chimeric reads (39,984 entries in total), providing a roughly balanced distribution between the two classes. After alignment with `minimap2`, only 19,984 clean reads remained because unmapped reads were not included in the BAM file. Some sequences failed to align due to the 5% error rate defined during `wgsim` simulation, which produced mismatches that caused certain reads to fall below the aligner's matching threshold.

This whole process is scheduled to start in the second week of November 2025 and is expected to be completed by the last week of November 2025, with a total duration of approximately three (3) weeks.

3.1.2 Feature Extraction Pipeline

This stage directly follows the previous alignment phase, utilizing the resulting `BAM` files (specifically `chimeric.sorted.bam` and `clean.sorted.bam`). A custom Python script was created to efficiently process each primary-mapped read to extract the necessary set of analytical features, which are then compiled into a structured feature matrix in `TSV` format. The pipeline's core functionality relies on libraries, namely `Pysam` (Version 0.22) for the robust parsing of `BAM` structures and `NumPy` (Version 1.26) for array operations and computations. The pipeline focuses on three principal features that collectively capture biological signatures associated with PCR-induced chimeras: (1) Supplementary alignment flag (SA count), (2) k-mer composition difference, and (3) microhomology.

Supplementary Alignment Flag

Split-alignment information was derived from the `SA` (Supplementary Alignment) tag embedded in each primary read of the `BAM` file. This tag is typically associated with reads that map to multiple genomic locations, suggesting a chimeric structure. To extract this information, the script first checked whether the read carried an `SA:Z` tag. If present, the tag string was parsed using the function `parse_sa_tag`, yielding a structure for each alignment containing the reference name, mapped position, strand, mapping quality, and number of mismatches.

After parsing, the function `sa_feature_stats` was applied to establish the fundamental split indicators, `has_sa` and `sa_count`. Along with these initial counts, the function synthesized a summarization by aggregating metrics related to the structure and reliability of the split alignments.

K-mer Composition Difference

Chimeric reads often comprise fragments from distinct genomic regions, resulting in a compositional discontinuity between segments. Comparing k-mer frequency profiles between the left and right halves of a read allows for the detection of such abrupt compositional shifts, independent of alignment information.

The script implemented this by inferring a likely junction breakpoint using the function `infer_breakpoints`, prioritizing the boundaries defined by soft-clipping operations in the `CIGAR` string. If no clipping was present, the midpoint of the alignment or the read length was utilized as a fallback. The read sequence was then divided into left and right segments at this inferred breakpoint, and k-mer frequency profiles ($k = 5$) were generated for both halves, ignoring any k-mers containing ambiguous 'N' bases. The resulting k-mer frequency vectors will be normalized and compared using the functions `cosine_difference` and `js_divergence`.

Microhomology

The workflow for extracting the microhomology feature also started by utilizing the `infer_breakpoints` similar to the k-mer workflow. Once a breakpoint was established, the script scanned a ±40 base pair window surrounding the breakpoint and used the function `longest_suffix_prefix_overlap` to identify the longest exact suffix-prefix overlap between the left and right read segments. This overlap, which represents consecutive bases shared at the junction, was recorded as the `microhomology_length` in the dataset. The 40-base pair window was chosen to ensure that short shared sequences at or near the breakpoint were captured, without including distant sequences that are unrelated. Additionally, the GC content of the overlapping sequence was calculated using the function `gc_content`, which counts guanine (G) and cytosine (C) bases within the detected microhomology and divides by the total length, yielding a proportion between 0 and 1, and was stored under the `microhomology_gc` attribute. Short microhomologies, typically 3-20 base pairs in length, are recognized signatures of PCR-induced template switching (Peccoud et al., 2018).

A k-mer length of 6 was used to capture patterns within the same 40-base pair window surrounding each breakpoint. These profiles complement microhomology measurements and help identify junctions that are potentially chimeric.

To ensure correctness and adherence to best practices, bioinformatics experts at the PGC Visayas will be consulted to validate the pipeline design, feature extraction logic, and overall data integrity. This stage of the study was scheduled to begin in the third week of November 2025 and conclude by the first week of December 2025, with an estimated total duration of approximately three (3) weeks.

3.1.3 Machine Learning Model Development

After feature extraction, the per-read feature matrices for clean and chimeric reads were merged into a single dataset. Each row corresponded to one paired-end read, and columns encoded alignment-structure features (e.g., supplementary alignment count and spacing between segments), CIGAR-derived soft-clipping statistics (e.g., left and right soft-clipped length, total clipped bases), k-mer composition discontinuity between read segments, and microhomology descriptors near candidate junctions. The resulting feature set was restricted to quantities that can be computed from standard BAM/FASTQ files in typical mitochondrial sequencing workflows.

The labelled dataset was randomly partitioned into training (80%) and test (20%) subsets using stratified sampling to preserve the 1:1 ratio of clean to chimeric reads. Model development and evaluation were implemented in Python (Version 3.11) using the `scikit-learn`, `xgboost`, `lightgbm`, and `catboost` libraries. A broad panel of classification algorithms was then benchmarked on the training data to obtain a fair comparison of different model families under identical feature conditions. The panel included: a trivial dummy classifier, L2-regularized logistic regression, a calibrated linear support vector machine (SVM), k-nearest neighbours, Gaussian Naïve Bayes, decision-tree ensembles (Random Forest, Extremely Randomized Trees, and Bagging with decision trees), gradient boosting methods (Gradient Boosting, XGBoost, LightGBM, and CatBoost), and a shallow multilayer perceptron (MLP).

For each model, five-fold stratified cross-validation was performed on the training set. In every fold, four-fifths of the data were used for fitting and the remaining one-fifth for validation. Mean cross-validation accuracy, precision, recall, F1-score for the chimeric class, and area under the receiver operating characteristic curve (ROC–AUC) were computed to summarize performance and rank candidate methods. This baseline screen allowed comparison of linear, probabilistic, neural, and ensemble-based approaches and identified tree-based ensemble and boosting models as consistently strong performers relative to simpler baselines.

3.1.4 Model Benchmarking, Hyperparameter Optimization, and Evaluation

Model selection and refinement proceeded in two stages. First, the cross-validation results from the broad panel were used to identify a subset of competitive models for more detailed optimization. Specifically, ten model families were carried forward: L2-regularized logistic regression, calibrated linear SVM, Random Forest, ExtraTrees, Gradient Boosting, XGBoost, LightGBM, CatBoost, Bagging with decision trees, and a shallow MLP. This subset spans both linear and nonlinear decision boundaries, but emphasizes ensemble and boosting methods, which showed superior F1 and ROC–AUC in the initial benchmark.

Second, hyperparameter optimization was conducted for each of the ten selected models using randomized search with five-fold stratified cross-validation (`RandomizedSearchCV`). For tree-based ensembles, the search space included the number of trees, maximum depth, minimum samples per split and leaf, and the fraction of features considered at each split. For boosting methods, key hyperparameters such as the number of boosting iterations, learning rate, tree depth, subsampling rate, and column subsampling rate were tuned. For the MLP, the number and size of hidden layers, learning rate, and L_2 regularization strength were varied. In all cases, the primary optimisation criterion was the F1-score of the chimeric class, averaged across folds.

For each model family, the hyperparameter configuration with the highest mean cross-validation F1-score was selected as the best-tuned estimator. These tuned models were then refitted on the full training set and evaluated once on the held-out test set to obtain unbiased estimates of performance. Test-set metrics included accuracy, precision, recall, F1-score for the chimeric class, and ROC–AUC. Confusion matrices and ROC curves were generated for the top-performing models to characterise common error modes, such as false negatives (missed chimeric reads) and false positives (clean reads incorrectly labelled as chimeric). The final model or small set of models for downstream interpretation was chosen based on a combination of test-set F1-score, ROC–AUC, and practical considerations such as model complexity and ease of deployment within a feature extraction pipeline.

3.1.5 Feature Importance and Interpretation

To relate model decisions to biologically meaningful signals, feature-importance analyses were performed on the best-performing tree-based models. Two complementary approaches were used. First, built-in importance measures from ensemble methods (e.g., split-based importances in Random Forest and Gradient Boosting) were examined to obtain an initial ranking of features based on their contribution to reducing impurity. Second, model-agnostic permutation importance was computed on the test set by repeatedly permuting each feature column while keeping all others fixed and measuring the resulting decrease in F1-score. Features whose permutation led to a larger performance drop were interpreted as more influential for chimera detection.

For interpretability, individual features were grouped into four conceptual families: (i) supplementary alignment and alignment-structure features (e.g., SA count, spacing between alignment segments, strand consistency), (ii) CIGAR-derived soft-clipping features (e.g., left and right soft-clipped length, total clipped bases), (iii) k-mer composition discontinuity features (e.g., cosine distance and Jensen–Shannon divergence between k-mer profiles of read segments), and (iv) microhomology descriptors (e.g., microhomology length and local GC content around putative breakpoints). Aggregating permutation importance scores within each family allowed assessment of which biological signatures contributed most strongly to the classifier's performance. This analysis provided a basis for interpreting the trained models in terms of known mechanisms of PCR-induced template switching and for identifying which alignment- and sequence-derived cues are most informative for distinguishing chimeric from clean mitochondrial reads.

3.1.6 Validation and Testing

Validation will involve both internal and external evaluations. Internal validation was achieved through five-fold cross-validation on the training data to verify model generalization and reduce variance due to random sampling. External validation will be achieved through testing on the 20% hold-out dataset derived from the simulated reads, which will be an unbiased benchmark to evaluate how well the trained models generalized to unseen data. All feature extraction and preprocessing steps were performed using the same feature extraction pipeline to ensure consistency and comparability across validation stages.

Comparative evaluation was performed across all candidate algorithms, including a trivial dummy classifier, L2-regularized logistic regression, a calibrated linear SVM, k-nearest neighbours, Gaussian Naïve Bayes, decision-tree ensembles, gradient boosting methods, and a shallow MLP. This evaluation determined which models demonstrated the highest predictive performance and computational efficiency under identical data conditions. Their metrics were compared to identify which algorithms were most suitable for further refinement.

3.1.7 Documentation

Comprehensive documentation was maintained throughout the study to ensure transparency and reproducibility. All stages of the research, including data gathering, preprocessing, feature extraction, model training, and validation, were systematically recorded in a `.README` file in the GitHub repository. For each analytical step, the corresponding parameters, software versions, and command line scripts were documented to enable exact replication of results.

The repository structure followed standard research data management practices, with clear directories for datasets and scripts. Computational environments were standardized using Conda, with an environment file (`environment.arm.yml`) specifying dependencies and package versions to maintain consistency across systems.

For manuscript preparation and supplementary materials, Overleaf (LaTeX) was used to produce publication-quality formatting and consistent referencing. f

3.2 Calendar of Activities

Table 3.1 presents the project timeline in the form of a Gantt chart, where each bullet point corresponds to approximately one week of planned activity.

Table 3.1: Timetable of Activities

Activities (2025)	**Nov**	**Dec**	**Jan**	**Feb**	**Mar**	**Apr**	**May**
Data Collection and Simulation	● ● ●●						
Feature Extraction Pipeline	●●	●					
Machine Learning Development			●●	● ● ●●	● ● ●●	●●	
Testing and Validation						●●	● ● ●●
Documentation	● ● ●●	● ● ●●	● ● ●●	● ● ●●	● ● ●●	● ● ●●	● ● ●●

Chapter 4

Results and Discussion

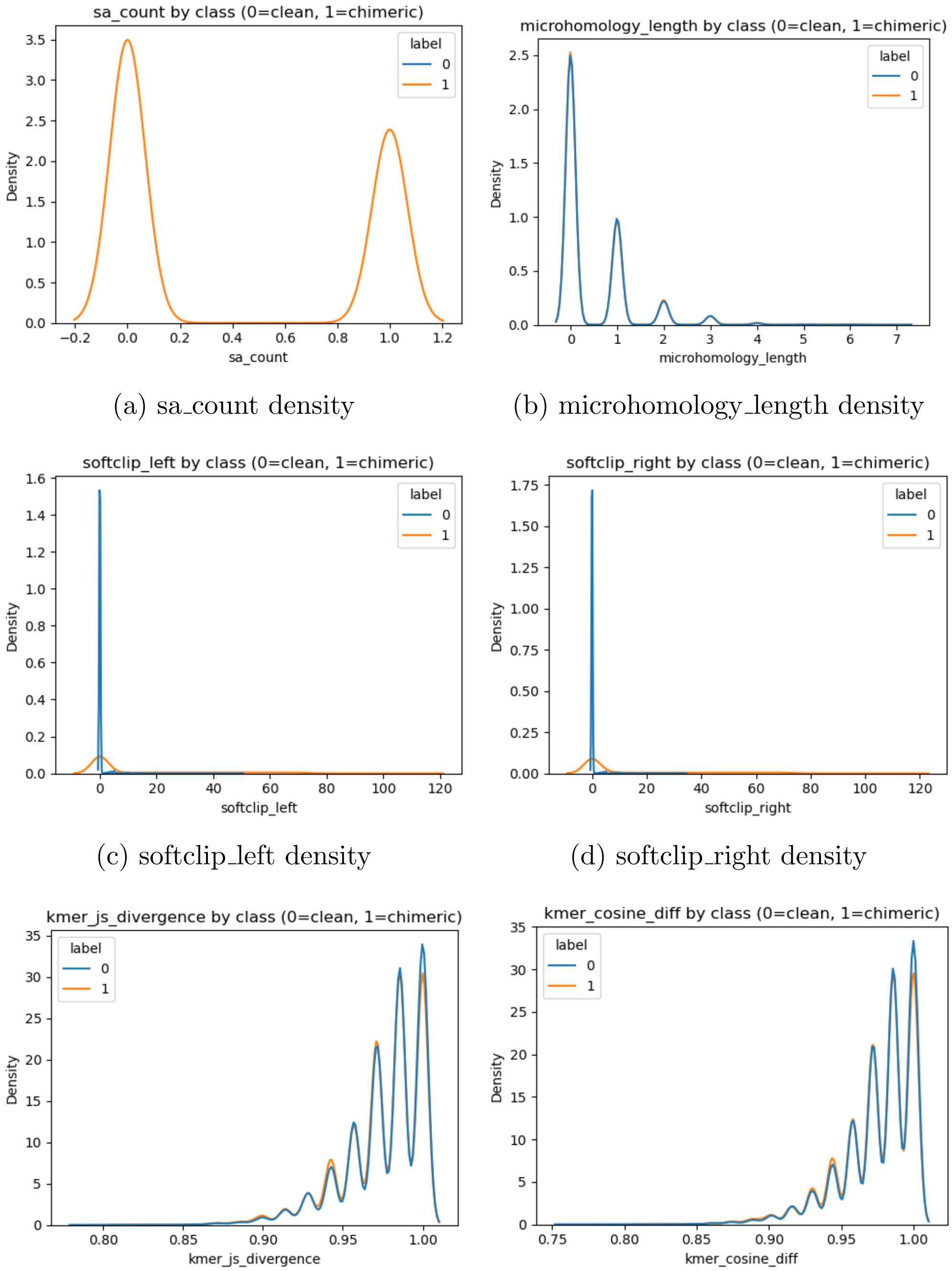
4.1 Descriptive Analysis of Features

This chapter presents the performance of the proposed feature set and machine-learning models for detecting PCR-induced chimeric reads in simulated mitochondrial Illumina data. We first describe the behaviour of the main features, then compare baseline classifiers, assess the effect of hyperparameter tuning, and finally analyse feature importance in terms of individual variables and biologically motivated feature families.

The final dataset contained 31,986 reads for training and 7,997 reads for testing, with classes balanced (approximately 4,000 clean and 4,000 chimeric reads in the test split).

4.1.1 Univariate Distributions

The kernel density plots in Figures 4.1a–4.1f collectively show that alignment-based features provide the strongest separation between clean and chimeric reads. The distribution of `sa_count` (Figure 4.1a) is distinctly bimodal, with clean reads concentrated near zero and chimeric reads peaking around one, reflecting the frequent presence of supplementary alignments in chimeras. A similar pattern of clear separation is observed in `softclip_left` and `softclip_right` (Figures 4.1c and 4.1d), where clean reads cluster tightly at zero while chimeric reads display broad, long-tailed distributions, consistent with extensive soft clipping when a read spans multiple genomic locations. In contrast, `microhomology_length` (Figure 4.1b) shows substantial overlap between classes, with both distributions sharply concentrated near zero and exhibiting smaller secondary peaks at short integer lengths, indicating limited discriminative value under the simulated conditions. Finally, the k-mer–based features `kmer_js_divergence` and `kmer_cosine_diff` (Figures 4.1e and 4.1f) exhibit highly overlapping, multimodal distributions with both classes peaking near 1.0; although chimeric reads appear slightly less concentrated at the highest similarity values, the separation is weak overall.



(a) sa_count density

(b) microhomology_length density

(c) softclip_left density

(d) softclip_right density

(e) kmer_js_divergence density

(f) kmer_cosine_diff density

Figure 4.1: Kernel density plots of six key features comparing clean and chimeric reads.

4.2 Baseline Classification Performance

Table 4.1 summarises the performance of eleven classifiers trained on the engineered feature set using five-fold cross-validation and evaluated on the held-out test set. All models were optimised using default hyperparameters, without dedicated tuning.

The dummy baseline, which always predicts the same class regardless of the input features, achieved an accuracy of 0.50 and test F1-score of 0.67. This reflects the balanced class distribution and provides a lower bound for meaningful performance.

Across other models, test F1-scores clustered in a narrow band between approximately 0.74 and 0.77 and ROC–AUC values between 0.82 and 0.84. Gradient boosting, CatBoost, LightGBM, XGBoost, bagging trees, random forest, and multilayer perceptron (MLP) all produced very similar scores, with CatBoost and gradient boosting slightly ahead (test F1 $\approx$ 0.77, ROC–AUC $\approx$ 0.84). Linear models (logistic regression and calibrated linear SVM) performed only marginally worse (test F1 $\approx$ 0.74), while Gaussian Naive Bayes lagged behind with substantially lower F1 ($\approx$ 0.65) despite very high precision for the chimeric class.

Table 4.1: Performance of baseline classifiers on the held-out test set.

model	test_accuracy	test_precision	test_recall	test_f1	test_roc_auc
dummy_baseline	0.500000	0.500000	1.000000	0.667000	0.500000
logreg_l2	0.789000	0.945000	0.614000	0.744000	0.821000
linear_svm_calibrated	0.789000	0.945000	0.614000	0.744000	0.820000
random_forest	0.788000	0.894000	0.654000	0.755000	0.834000
extra_trees	0.788000	0.901000	0.647000	0.753000	0.824000
gradient_boosting	0.802000	0.936000	0.648000	0.766000	0.840000
xgboost	0.800000	0.929000	0.650000	0.765000	0.839000
lightgbm	0.799000	0.926000	0.650000	0.764000	0.838000
catboost	0.803000	0.936000	0.650000	0.767000	0.839000
knn	0.782000	0.892000	0.642000	0.747000	0.815000
gaussian_nb	0.741000	0.996000	0.483000	0.651000	0.819000
bagging_trees	0.792000	0.900000	0.657000	0.760000	0.837000
mlp	0.789000	0.931000	0.625000	0.748000	0.819000

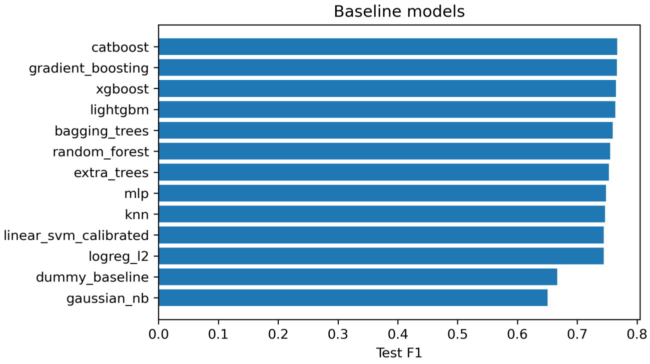


Figure 4.2: Test F1 of all baseline classifiers, showing that no single model clearly dominates and several achieve comparable performance.

4.3 Effect of Hyperparameter Tuning

To assess whether performance could be improved further, ten model families underwent randomised hyperparameter search (Chapter 3). The tuned metrics are summarised in Table 4.2. Overall, tuning yielded modest but consistent gains for tree-based ensembles and boosting methods, while leaving linear models essentially unchanged or slightly worse.

CatBoost, gradient boosting, LightGBM, XGBoost, random forest, bagging trees, and MLP all experienced small increases in test F1 (typically ΔF1 ≈ 0.002–0.009) and ROC–AUC (up to ΔAUC ≈ 0.008). After tuning, CatBoost remained the best performer with test accuracy 0.802, precision 0.924, recall 0.658, F1-score 0.769, and ROC–AUC 0.844. Gradient boosting achieved almost identical performance (F1 0.767, AUC 0.843). Random forest and bagging trees also improved to F1 scores around 0.763 with AUC ≈ 0.842.

Table 4.2: Performance of tuned classifiers on the held-out test set.

model	test_accuracy	test_precision	test_recall	test_f1	test_roc_auc
logreg_l2_tuned	0.788000	0.946000	0.612000	0.743000	0.818000
linear_svm_calibrated_tuned	0.788000	0.944000	0.612000	0.743000	0.818000
random_forest_tuned	0.797000	0.915000	0.655000	0.763000	0.842000
extra_trees_tuned	0.794000	0.910000	0.652000	0.760000	0.837000
gradient_boosting_tuned	0.802000	0.928000	0.654000	0.767000	0.843000
xgboost_tuned	0.799000	0.922000	0.653000	0.765000	0.839000
lightgbm_tuned	0.801000	0.930000	0.651000	0.766000	0.842000
catboost_tuned	0.802000	0.924000	0.658000	0.769000	0.844000
bagging_trees_tuned	0.798000	0.922000	0.650000	0.763000	0.842000
mlp_tuned	0.790000	0.934000	0.625000	0.749000	0.821000

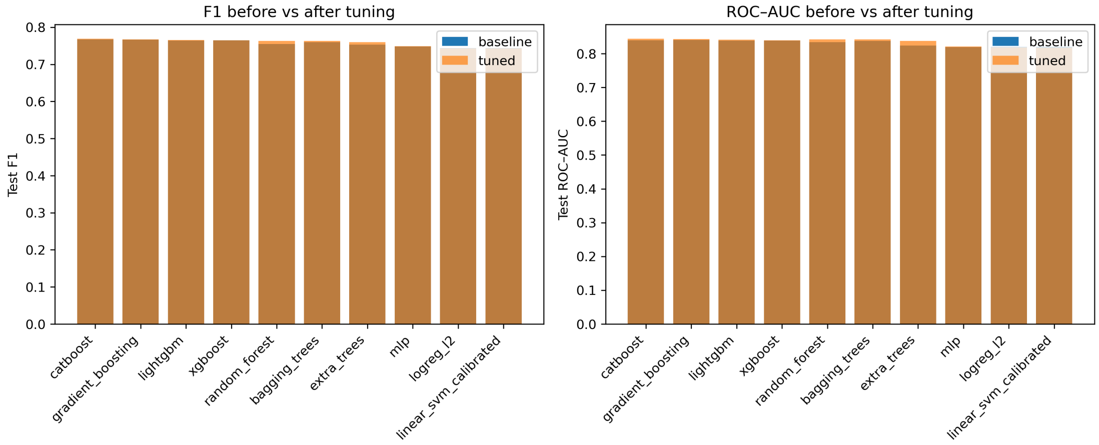


Figure 4.3: Comparison of test F1 (left) and ROC–AUC (right) for baseline and tuned models. Hyperparameter tuning yields small but consistent gains, particularly for tree-based ensembles.

Because improvements are small and within cross-validation variability, we interpret tuning as stabilising and slightly refining the models rather than fundamentally altering their behaviour or their relative ranking.

4.4 Detailed Evaluation of Representative Models

For interpretability and diversity, four tuned models were selected for deeper analysis: CatBoost (best-performing boosted tree), scikit-learn gradient boosting (canonical gradient-boosting implementation), random forest (non-boosted ensemble baseline), and L2-regularised logistic regression (linear baseline). All models were trained on the engineered feature set and evaluated on the same held-out test data.

4.4.1 Confusion Matrices and Error Patterns

Classification reports and confusion matrices for the four models reveal consistent patterns. CatBoost and gradient boosting both reached overall accuracy of approximately 0.80 with similar macro-averaged F1 scores (~ 0.80). For CatBoost, precision and recall for clean reads were 0.73 and 0.95, respectively, while for chimeric reads they were 0.92 and 0.66 ($F1 = 0.77$). Gradient boosting showed nearly identical trade-offs.

Random forest attained slightly lower accuracy (0.80) and chimeric F1 (0.76), whereas logistic regression achieved the lowest accuracy among the four (0.79) and chimeric F1 (0.74), although it provided the highest chimeric precision (0.95) at the cost of lower recall (0.61).

Across all models, errors were asymmetric. False negatives (chimeric reads predicted as clean) were more frequent than false positives. For example, CatBoost misclassified 1 369 chimeric reads as clean but only 215 clean reads as chimeric. This pattern indicates that the models are conservative: they prioritise avoiding spurious chimera calls at the expense of missing some true chimeras. Depending on downstream application, alternative decision thresholds or cost-sensitive training could be explored to adjust this balance.

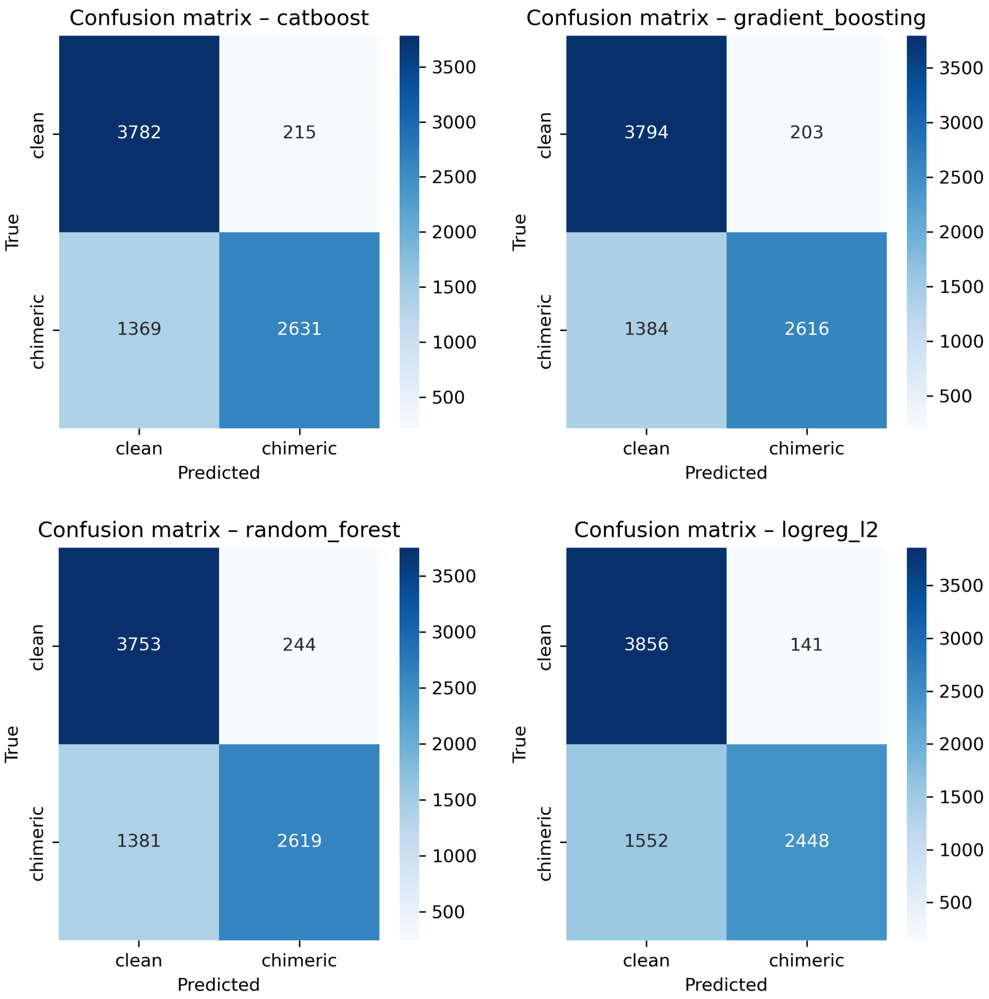


Figure 4.4: Confusion matrices for the four representative models on the held-out test set. All models show more false negatives (chimeric reads called clean) than false positives.

4.4.2 ROC and Precision–Recall Curves

Receiver operating characteristic (ROC) and precision–recall (PR) curves (Figure 4.5) further support the similarity among the top models. The three tree-based ensembles (CatBoost, gradient boosting, random forest) achieved ROC–AUC values of approximately 0.84 and average precision (AP) around 0.88. Logistic regression performed slightly worse (AUC ≈ 0.82, AP ≈ 0.87) but still substantially better than random guessing.

The PR curves show that precision remains above 0.9 across a broad range of recall values (up to roughly 0.5–0.6), after which precision gradually declines. This behaviour indicates that the models can assign very high confidence to a subset of chimeric reads, while more ambiguous reads can only be recovered by accepting lower precision.

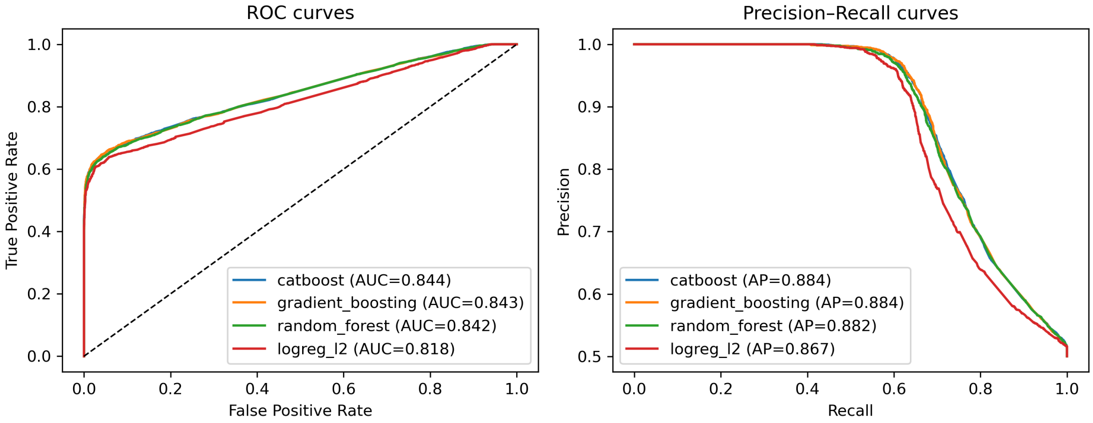


Figure 4.5: ROC (left) and precision–recall (right) curves for the four representative models on the held-out test set. Tree-based ensembles cluster closely, with logistic regression performing slightly but consistently worse.

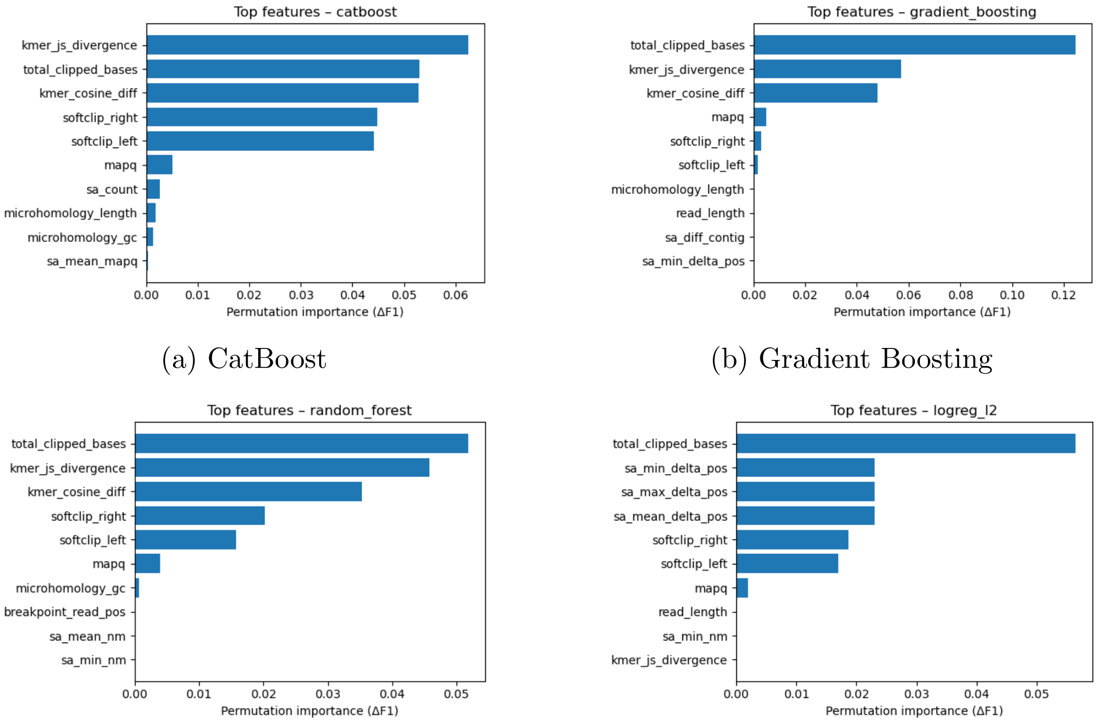
4.5 Feature Importance and Biological Interpretation

4.5.1 Permutation Importance of Individual Features

To understand how each classifier made predictions, feature importance was quantified using permutation importance. In this approach, the values of a single feature are randomly shuffled, and the resulting drop in F_1 score (ΔF_1) reflects how strongly the model depends on that feature. Greater decreases in F_1 indicate stronger reliance on that feature. This analysis was applied to four representative models: CatBoost, Gradient Boosting, Random Forest, and L_2-regularized Logistic Regression.

As shown in Figure 4.6, the total number of clipped bases consistently provides a strong predictive signal, particularly in Random Forest, Gradient Boosting, and L_2-regularized Logistic Regression. CatBoost differs by assigning the highest importance to k-mer divergence metrics such as `kmer_js_divergence`, which capture subtle sequence changes resulting from structural variants or PCR-induced chimeras. Soft-clipping features (`softclip_left` and `softclip_right`) provide additional context around breakpoints, complementing these primary signals in all models except Gradient Boosting. L_2-regularized Logistic Regression relies more on alignment-based split-read metrics when breakpoints are simple, but it is less effective at detecting complex rearrangements that introduce novel sequences.

Overall, these results indicate that accurate detection of chimeric reads relies on both alignment-based signals and k-mer compositional information. Explicit microhomology features contribute minimally in this analysis, and combining both alignment-based and sequence-level features enhances model sensitivity and specificity.



(a) CatBoost

(b) Gradient Boosting

(c) Random Forest

(d) L_2-regularized Logistic Regression

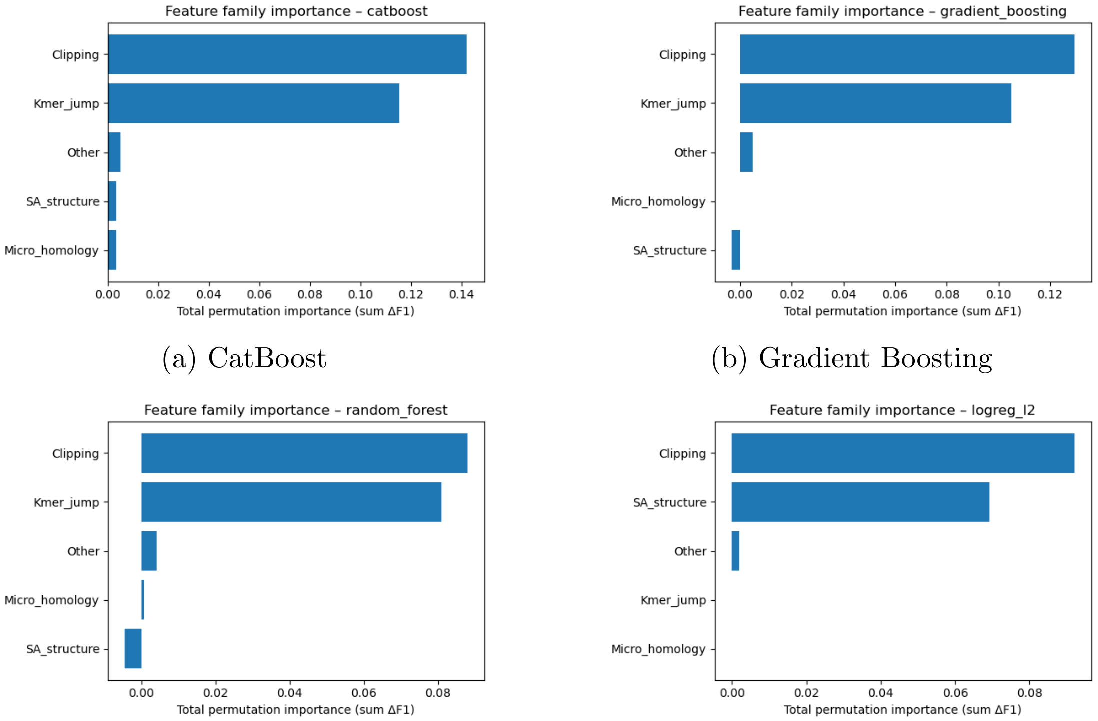
Figure 4.6: Permutation-based feature importance for four representative classifiers. Clipping and k-mer composition features are generally the strongest predictors, whereas microhomology and other alignment metrics contribute minimally.

4.5.2 Feature Family Importance

To evaluate the contribution of broader biological signals, features were grouped into five families: SA_structure (supplementary alignment and segment metrics, e.g., `has_sa`, `sa_count`, `sa_min_delta_pos`, `sa_mean_nm`), Clipping (`softclip_left`, `softclip_right`, `total_clipped_bases`, `breakpoint_read_pos`), Kmer_jump (`kmer_cosine_diff`, `kmer_js_divergence`), Micro_homology, and Other (e.g., `mapq`).

Aggregated analyses reveal consistent patterns across models. In CatBoost, the Clipping family has the largest cumulative contribution (0.14), followed by Kmer_jump (0.12), with Other features contributing modestly (0.005) and SA_structure (0.003) and Micro_homology (0.003) providing minimal predictive power. Gradient Boosting shows a similar trend, with Clipping (0.13) dominating, Kmer_jump (0.11) secondary, and the remaining families contributing negligibly. Random Forest integrates both Clipping (0.088) and Kmer_jump (0.08) effectively, while SA_structure, Micro_homology, and Other remain minor contributors. L_2-regularized Logistic Regression emphasizes Clipping (0.09) and SA_structure (0.07), with Kmer_jump and Micro_homology having minimal impact.

Both feature-level and aggregated analyses indicate that detection of chimeric reads in this dataset relies primarily on alignment disruptions (Clipping) and k-mer compositional shifts (Kmer_jump), which often arise from PCR-induced recombination events, while explicit microhomology features contribute minimally.



(a) CatBoost

(b) Gradient Boosting

(c) Random Forest

(d) L_2-regularized Logistic Regression

Figure 4.7: Aggregated feature family importance across four models. Clipping and k-mer compositional shifts are consistently the dominant contributors, while SA_structure, Micro_homology, and other features contribute minimally.

4.6 Summary of Findings

After removing trivially discriminative metadata, all models performed substantially better than the dummy baseline, with test F1-scores around 0.76 and ROC-AUC values near 0.84. Hyperparameter tuning yielded modest improvements, with boosting methods, particularly CatBoost and gradient boosting, achieving the highest performance. Confusion matrices and precision-recall curves indicate that these models prioritise precision for chimeric reads while accepting lower recall, which a conservative strategy appropriate for scenarios where false positives are costly.

Feature importance analyses revealed that alignment disruptions, such as clipping, and abrupt k-mer composition changes accounted for most predictive power. In contrast, microhomology metrics and supplementary alignment descriptors contributed minimally. These results indicate that features based on read alignment and k-mer composition are sufficient to train classifiers for detecting mitochondrial PCR-induced chimera reads, without needing additional quality-score or positional information in the conditions tested.

References

Anderson, S., Bankier, A., Barrell, B., Bruijn, M., Coulson, A., Drouin, J., … Young, I. (1981, 04). Sequence and organization of the human mitochondrial genome. *Nature*, *290*, 457-465. doi: 10.1038/290457a0

Arango, G., Garner, E., Pruden, A., Heath, L., Vikesland, P., & Zhang, L. (2018, 02). Deeparg: A deep learning approach for predicting antibiotic resistance genes from metagenomic data. *Microbiome*, *6*. doi: 10.1186/s40168-018-0401-z

Bentley, D. R., Balasubramanian, S., Swerdlow, H. P., Smith, G. P., Milton, J., Brown, C. G., … Smith, A. J. (2008). Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, *456*(7218), 53–59. doi: 10.1038/nature07517

Boore, J. L. (1999). Animal mitochondrial genomes. *Nucleic Acids Research*, *27*(8), 1767–1780. doi: 10.1093/nar/27.8.1767

Cameron, S. L. (2014). Insect mitochondrial genomics: Implications for evolution and phylogeny. *Annual Review of Entomology*, *59*, 95–117. doi: 10.1146/annurev-ento-011613-162007

Dierckxsens, N., Mardulyn, P., & Smits, G. (2017). Novoplasty: de novo assembly of organelle genomes from whole genome data. *Nucleic Acids Research*, *45*(4), e18. doi: 10.1093/nar/gkw955

Edgar, R. C. (2016). Uchime2: improved chimera prediction for amplicon sequencing. *bioRxiv*. Retrieved from https://api.semanticscholar.org/CorpusID:88955007

Edgar, R. C. (n.d). *Uchime in practice.* Retrieved from https://www.drive5.com/usearch/manual7/uchime_practical.html

Edgar, R. C., Haas, B. J., Clemente, J. C., Quince, C., & Knight, R. (2011). Uchime improves sensitivity and speed of chimera detection. *Bioinformatics*, *27*(16), 2194–2200. doi: 10.1093/bioinformatics/btr381

Glenn, T. C. (2011). Field guide to next-generation dna sequencers. *Molecular Ecology Resources*, *11*(5), 759–769. doi: 10.1111/j.1755-0998.2011.03024.x

Gonzalez, J. M., Zimmermann, J., & Saiz-Jimenez, C. (2004, 09). Evaluating putative chimeric sequences from pcr-amplified products. *Bioinformatics*, *21*(3), 333-337. Retrieved from https://doi.org/10.1093/bioinformatics/bti008 doi: 10.1093/bioinformatics/bti008

Gray, M. W. (2012). Mitochondrial evolution. *Cold Spring Harbor perspectives in biology*, *4*. Retrieved from https://doi.org/10.1101/cshperspect.a011403 doi: 10.1101/cshperspect.a011403

Hahn, C., Bachmann, L., & Chevreux, B. (2013). Reconstructing mitochondrial genomes directly from genomic next-generation sequencing reads—a baiting and iterative mapping approach. *Nucleic Acids Research*, *41*(13), e129. doi: 10.1093/nar/gkt371

Jin, J.-J., Yu, W.-B., Yang, J., Song, Y., dePamphilis, C. W., Yi, T.-S., & Li, D.-Z. (2020). Getorganelle: a fast and versatile toolkit for accurate de novo assembly of organelle genomes. *Genome Biology*, *21*(1), 241. doi: 10.1186/s13059-020-02154-5

Judo, M. S. B., Wedel, W. R., & Wilson, B. H. (1998). Stimulation and suppression of pcr-mediated recombination. *Nucleic Acids Research*, *26*(7), 1819–1825. doi: 10.1093/nar/26.7.1819

Labrador, K., Agmata, A., Palermo, J. D., Ravago-Gotanco, R., & Pante, M. J. (2021). Mitochondrial dna reveals genetically structured haplogroups of bali sardinella (sardinella lemuru) in philippine waters. *Regional Studies in Marine Science*, *41*, 101588. doi: 10.1016/j.rsma.2020.101588

Li, H. (2018, 05). Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, *34*(18), 3094-3100. Retrieved from https://doi.org/10.1093/bioinformatics/bty191 doi: 10.1093/bioinformatics/bty191

Liang, Q., Bible, P. W., Liu, Y., Zou, B., & Wei, L. (2020, 02). Deepmicrobes: taxonomic classification for metagenomics with deep learning. *NAR Genomics and Bioinformatics*, *2*(1), lqaa009. Retrieved from https://doi.org/10.1093/nargab/lqaa009 doi: 10.1093/nargab/lqaa009

Metzker, M. L. (2010). Sequencing technologies — the next generation. *Nature Reviews Genetics*, *11*(1), 31–46. doi: 10.1038/nrg2626

Mysara, M., Saeys, Y., Leys, N., Raes, J., & Monsieurs, P. (2015). Catch, an ensemble classifier for chimera detection in 16s rrna sequencing studies. *Applied and Environmental Microbiology*, *81*(5), 1573-1584. Retrieved from https://journals.asm.org/doi/abs/10.1128/aem.02896-14 doi: 10.1128/AEM.02896-14

Peccoud, J., Lequime, S., Moltini-Conclois, I., Giraud, I., Lambrechts, L., & Gilbert, C. (2018, 04). A survey of virus recombination uncovers canonical features of artificial chimeras generated during deep sequencing library preparation. *G3 Genes—Genomes—Genetics*, *8*(4), 1129-1138. Retrieved from https://doi.org/10.1534/g3.117.300468 doi: 10.1534/

g3.117.300468

Qin, Y., Wu, L., Zhang, Q., Wen, C., Nostrand, J. D. V., Ning, D., … Zhou, J. (2023). Effects of error, chimera, bias, and gc content on the accuracy of amplicon sequencing. *mSystems*, *8*(6), e01025-23. Retrieved from `https://journals.asm.org/doi/abs/10.1128/msystems.01025-23` doi: 10.1128/msystems.01025-23

Qiu, X., Wu, L., Huang, H., McDonel, P. E., Palumbo, A. V., Tiedje, J. M., & Zhou, J. (2001). Evaluation of pcr-generated chimeras, mutations, and heteroduplexes with 16s rrna gene-based cloning. *Applied and Environmental Microbiology*, *67*(2), 880–887. doi: 10.1128/AEM.67.2.880-887.2001

Ren, J., Song, K., Deng, C., Ahlgren, N., Fuhrman, J., Li, Y., … Sun, F. (2020, 01). Identifying viruses from metagenomic data using deep learning. *Quantitative Biology*, *8*. doi: 10.1007/s40484-019-0187-4

Rodriguez-Martin, B., Palumbo, E., Marco-Sola, S., Griebel, T., Ribeca, P., Alonso, G., … Djebali, S. (2017, 01). Chimpipe: Accurate detection of fusion genes and transcription-induced chimeras from rna-seq data. *BMC Genomics*, *18*. doi: 10.1186/s12864-016-3404-9

Rognes, T., Flouri, T., Nichols, B., Quince, C., & Mahé, F. (2016). Vsearch: a versatile open source tool for metagenomics. *PeerJ*, *4*, e2584. doi: 10.7717/peerj.2584

Sedlazeck, F., Rescheneder, P., Smolka, M., Fang, H., Nattestad, M., von Haeseler, A., & Schatz, M. (2018, 06). Accurate detection of complex structural variations using single-molecule sequencing. *Nature Methods*, *15*. doi: 10.1038/s41592-018-0001-7

Sfeir, A., & Symington, L. S. (2015). Microhomology-mediated end joining: A back-up survival mechanism or dedicated pathway? *Trends in Biochemical*

Sciences, *40*(11), 701-714. Retrieved from https://www.sciencedirect.com/science/article/pii/S0968000415001589 doi: https://doi.org/10.1016/j.tibs.2015.08.006

Vervier, K., Mahé, P., Tournoud, M., Veyrieras, J.-B., & Vert, J.-P. (2015, 11). Large-scale machine learning for metagenomics sequence classification. *Bioinformatics*, *32*(7), 1023-1032. Retrieved from https://doi.org/10.1093/bioinformatics/btv683 doi: 10.1093/bioinformatics/btv683

Willette, D., Bognot, E., Mutia, M. T., & Santos, M. (2011). *Biology and ecology of sardines in the philippines: A review* (Vol. 13; Tech. Rep. No. 1). NFRDI Technical Paper Series. Retrieved from https://nfrdi.da.gov.ph/tpjf/etc/Willette%20et%20al.%20Sardines%20Review.pdf