jun.-23

Illumina2Nanopore: A Pipeline for Adapting Illumina Libraries and Implementing a Clustering-Based Error Correction Method

Álvaro Herrero Reiriz

Illumina2Nanopore: A Pipeline for Adapting Illumina Libraries and Implementing a Clustering-Based Error Correction Method

PROYECTO
presentado para optar
al Título de Grado en Ingeniería Biomédica por
Álvaro Herrero Reiriz

bajo la dirección de la
Dra. Lorea Blázquez

Y bajo la supervisión del profesor
Ángel Rubio Díaz-Cordovés

biodonostia
health research institute

tecnun
Universidad de Navarra

Donostia-San Sebastián, junio 2023

Tecnun Universidad de Navarra | ESCUELA DE INGENIERÍA INGENIARITZA ESKOLA SCHOOL OF ENGINEERING

Proyecto Fin de Grado

INGENIERÍA BIOMÉDICA

Illumina2Nanopore: A Pipeline for Adapting Illumina Libraries and Implementing a Clustering-Based Error Correction Method

Álvaro Herrero Reiriz

San Sebastián, junio de 2023

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 ABSTRACT

In this project, a comprehensive pipeline was developed to enable the sequencing of Illumina libraries with Oxford Nanopore sequencers and providing accurate gene expression quantification. The pipeline encompassed various steps, including sample and library demultiplexing, as well as Unique Molecular Identifier (UMI) deduplication. Additionally, a novel tool named UMIclusterer was created to provide deduplication by read clustering based on UMI Hamming distance and genomic coordinates, while incorporating error correction through global alignment (Needleman-Wunsch) and a quality- and abundance-based scoring system.

To evaluate the performance of UMIclusterer, a comparison was conducted using a set of human samples processed with three different methods: a standard no-deduplication pipeline, a standard UMItools deduplication approach, and the proposed UMIclusterer method. The results demonstrated that UMIclusterer outperformed the other methods in terms of accuracy and reliability. However, it should be noted that the sequencing depth was not sufficient, and further improvements could be achieved by increasing the depth of sequencing.

2 RESUMEN

En este proyecto, se ha desarrollado un *pipeline* integral que permite la secuenciación de bibliotecas Illumina con secuenciadores Nanopore, incluyendo la cuantificación de la expresión génica de las muestras secuenciadas. El proceso incluye diferentes pasos, incluida la demultiplexación por muestra y librería, así como la deduplicación mediante el uso de Identificadores Moleculares Únicos (UMI). Además, se ha publicado una nueva herramienta llamada UMIclusterer que proporciona un método para la deduplicación mediante agrupación de *reads* basada en la distancia de Hamming entre UMI y coordenadas genómicas, incorporando al mismo tiempo corrección de errores mediante alineamiento global (Needleman-Wunsch) y un sistema de puntuación basado en la abundancia y calidad de las bases en la secuencia.

Para evaluar el rendimiento del UMIclusterer, se a llevado a cabo una comparación utilizando un conjunto de muestras humanas procesadas con tres métodos diferentes: un *pipeline* estándar sin deduplicación, uno estándar usando UMItools y otro utilizando la herramienta UMIclusterer propuesta. Los resultados muestran que el UMIclusterer supera a los demás métodos en términos de *indels* y errores. Sin embargo, hay que señalar que la profundidad de secuenciación no fue todo lo grande que se esperaba, y por lo tanto se podrían conseguir mejores resultados aumentando la misma.

3 INTRODUCTION

Bioinformatics has revolutionized the analysis of genomic data, enabling researchers to unravel complex biological processes and gain insights into gene expression patterns.

RNA sequencing (RNA-seq) has emerged as a powerful tool for studying gene expression by providing a comprehensive view of the transcriptome across different tissues in our body. Traditional RNA-seq methods often rely on short-read sequencing technologies, such as Illumina sequencing, which involves some limitations, namely having a maximum sequencing length (around 150-250 nucleotides) and being dependent on external laboratories to carry out the sequencing, extending the waiting time for results.

In recent years, Oxford Nanopore Technologies' sequencing technology has gained attention due to its ability to generate long-read sequences in real-time. The long reads offered by Nanopore sequencing enable the capture of full-length RNA transcripts and the ability to detect alternative splicing events, making it a promising approach for comprehensive transcriptomic analysis. Moreover, it also allows for direct RNA sequencing, avoiding intermediate PCR duplication.

However, Nanopore sequencing has also some downsides when compared to its counterparts. Among other things, it has a high tendency to insert non-existent nucleotides, especially in areas of low complexity such as polyA tails; as well as a significantly lower accuracy (up to 96% at best) compared to Illumina (up to 99.999%).

Bearing all this in mind, its evident that Nanopore could offer a fast alternative for diagnosis and prototyping of laboratory protocols without the need for long waiting times, provided that its accuracy and stability can be increased. For this reason, in this work I present a protocol to adapt existing Illumina libraries or libraries to be sequenced both in Illumina and Nanopore machines in parallel (using Illumina to verify Nanopore results afterwards). At the same time, a novel tool called UMIclusterer has been developed to provide a solution to the limited accuracy by clustering reads by their UMI Hamming distance and genomic distance, and then performing error correction by creating a consensus read out of each cluster.

In the next section of this document, a brief background of the technologies involved in this project will be given. After that, the pipeline developed in order to obtain demultiplexed and deduplicated BAM files will be broken down part by part. A Nextflow implementation optimized for HPC clusters is included, which is available on GitHub along with Docker containers with all the required dependencies. Then, the UMIclusterer tool will be compared to other deduplication tools already available, and all the processes that this tool carries out will also be explained. The code is also available on GitHub.

4 BACKGROUND

Advances in DNA and RNA sequencing technologies have revolutionized genomics and medicine, allowing for quick and cost-effective studies of genetic material. Oxford Nanopore Techonologies' sequencers are an example of what is to come in the close future in terms of flexibility and speed.

Nanopore sequencing is a single-molecule sequencing technique that employs the principle of passing DNA or RNA molecules through a nanopore, a small pore in a membrane. Through this pore, a current is passed, playing the role of a molecular sensor that detects changes in the current as nucleotides pass through. These current fluctuations are used to determine the sequence of the genetic material with the help of AI.

Illumina sequencing, on the other hand, employs a parallel sequencing-by-synthesis approach. Unlike Nanopore, Illumina relies on fluorescently labeled nucleotides and optical detection. This method offers a tremendously high throughput and a remarkably high accuracy of around 99.999%.

The appeal of Nanopore lies in the following key points:

Real-time sequencing: the basecalling process can be performed as the DNA or RNA molecule passes through the nanopore. This real-time feature allows for rapid and on-site analysis of samples, making it suitable for various applications, such as infectious disease surveillance and field research.

Long-read capability: Nanopore sequencing technology can generate long sequencing reads, typically spanning thousands of bases, enabling the sequencing of long fragments of DNA or RNA. This characteristic is particularly advantageous in resolving complex genomic regions, structural variations, and repetitive sequences that are challenging to analyze using other sequencing methods.

Direct detection of modified bases: Unlike other sequencing technologies, Nanopore sequencing can directly detect modifications to individual nucleotides, such as DNA methylation or RNA modifications. This capability provides valuable insights into epigenetic modifications and regulatory mechanisms, expanding the potential applications of Nanopore sequencing in various fields.

However, these come at the cost of a relatively high error rate, specially in low complexity areas, where the low fluctuation in the signal makes it difficult to determine the exact quantity of nucleotides in that fragment. Another limitation of Nanopore is the lower throughput compared to other technologies like Illumina. This is due to the fact that individual molecules must get through pores and the parallelization of the process is very limited compared again to Illumina, where millions of strands can be sequenced at the same time.

Table 1: Summary of the comparison of Illumina Sequencing and Nanopore Sequencing

	Illumina Sequencing	**Nanopore Sequencing**
Principle	Sequencing-by-synthesis	Single-molecule sequencing
Read Length	Short reads (10^2 bases)	Long reads (10^3 - 10^4 bases)
Real-time Analysis	Not available	Available
Throughput	10^6 - 10^9 reads per run	Lower 10^4 - 10^6 reads per run
Accuracy	99.999%	96%
Modified Base Detection	No	Yes
Portable	No	Yes

5 MATERIALS AND METHODS

5.1 Sample Preparation

5.1.1 Sample source and description

The samples used in this study were derived from modified HEK (Human Embryonic Kidney) cells, as well as mouse cells, some of them containing the Huntington disease mutation. Huntington disease is a neurodegenerative disorder caused by an expanded CAG repeat in the huntingtin (HTT) gene, leading to the production of a mutant huntingtin protein. This mutation results in the progressive degeneration of neurons and is associated with various motor, cognitive, and psychiatric symptoms The Huntington's Disease Collaborative Research Group (1993).

Among the HEK cells used, there were wild-type cells, expressing the normal HTT gene without any expanded CAG repeats, as well as other modified versions of these cells that will not be specified as they are not the subject of this study. Mouse cells were composed of wild-type cells and mutation-induced knock-in and YAC cells. In addition, two non-template control samples were included. These controls are essential for assessing background noise, potential contamination, or artifacts introduced during the experimental process. The non-template control samples lacked any genetic material and were used to ensure the specificity and accuracy of the sequencing and analysis methods.

The objective of this study was to analyze the canonical and cryptic polyadenylation events in the samples. Polyadenylation is a crucial post-transcriptional modification involving the addition of a poly(A) tail to mRNA molecules. Aberrant polyadenylation events, both canonical (normal) and cryptic (anomalous), can have significant implications for gene expression regulation and disease pathogenesis.

5.1.2 RNA Preparation

After the RNA extraction, sample-specific 5 nucleotide long barcodes were added, allowing for multiplexing of multiple samples in a single sequencing run. In addition, Illumina i7 indexes were included during library preparation to enable the sequencing of additional libraries.

To enhance the accuracy of quantification and error correction, unique molecular identifiers (UMIs) consisting of 10 nucleotide oligos were added to each RNA sample. The incorporation of UMIs enabled the identification and removal of PCR duplicates during the data processing pipeline.

Illumina standard P5 and P3 adapters were employed, making the library available to be also sequenced with Illumina technology.These adapters facilitate the attachment of the sequencing primers on the flow cell. The use of standardized Illumina adapters ensured compatibility with different sequencing platforms and downstream analysis pipelines.

Following adapter ligation, the RNA samples were subjected to a targeted PCR protocol. This targeted PCR amplification step selectively amplifies the regions of interest, minimizing the amplification of non-targeted regions and reducing potential biases. The targeted PCR protocol employed specific primers designed to amplify the desired RNA sequences.

Table 2: Summary of key read elements.

Element	Description
5’ adapter	P5
3’ adapter	P3
UMI	10 nt
Barcode	5 nt
Index	i7

5.2 Illumina Sequencing

By following this protocol, samples can be sequenced both in Nanopore and Illumina sequencers. The first method will give fast in-situ results, while the second will give more accurate reads but will take longer to be available. The mouse library was sequenced both with Illumina and Nanopore to provide an example comparison.

5.3 Nanopore Sequencing

Nanopore sequencing was performed using the MinION Mk1C device, FLO-MIN106 flowcell, and R9.4.1 chemistry. The DNA samples derived from RNA PCR products were prepared with the SQK-LSK110 ligation kit, and left sequencing overnight. Basecalling was performed using the super high accuracy (SUP) model on Guppy version 6.4.6, the most performant AI model available at the moment.

5.4 Data Processing Pipeline

The following section will present an overview of the goals of this protocol and highlight the third-party tools involved, followed by a step-by-step explanation of each of the sub-processes to be performed.

5.4.1 Challenges

One of the primary challenges is extracting original reads out of joined reads, artifacts and noisy insertions inside of or following low complexity areas. At the same time, accurately determining the strand orientation of raw sequencing reads is crucial for downstream processing.

Another significant challenge is filtering out reads with high levels of noise. During the sequencing process, various sources of noise, such as sequencing errors or low-quality reads, can be introduced. Filtering out these is essential to maintain the accuracy and reliability of downstream analyses, as they can lead to false interpretations and erroneous results.

Ensuring the correct demultiplexing of the samples is another important requirement. Demultiplexing involves assigning sequencing reads to their respective samples based on their sample-specific barcodes. Accurate demultiplexing is critical to prevent cross-contamination and ensure that the data from each sample is correctly analyzed independently. To do so, it is paramount to identify the position of the barcode in the sequenced read.

The pipeline also aims to address the bias introduced by the PCR amplification. PCR bias occurs when certain sequences are preferentially amplified over others, leading to biased representation and potential distortions in the final data, especially when trying to quantify expression. By implementing strategies such as unique molecular identifiers (UMIs), the pipeline aims to minimize bias and improve the accuracy of downstream analyses.

Lastly, maintaining data quality, reliability and traceability throughout each step of the pipeline is a primary goal. Quality control measures are implemented to assess the quality of the sequencing data at various stages. These quality control steps help identify and trace back any potential issues, ensuring that the data generated is of high quality and suitable for reliable interpretation.

5.4.2 Bioinformatics tools used for data analysis and quality control

The generated sequencing data underwent rigorous quality control (QC) using a set of bioinformatics tools. FastQC (Andrews et al. (2012)) was employed for initial quality assessment, providing valuable insights into read quality, base composition, and sequence duplication levels. To summarize and visualize the QC results, MultiQC (Ewels et al. (2016)) was used to generate comprehensive QC reports.

SeqKit (Shen et al. (2016)) was utilized for various sequence analysis and manipulation tasks, such as generating file stat summaries and computing reverse complements. The tool Cutadapt (Martin (2011)) played a crucial role in adapter removal, ensuring accurate downstream analysis by eliminating adapter contamination, as well as the primary tool for demultiplexing purposes.

For unique molecular identifier (UMI) processing, UMI-tools (Smith et al. (2017)) was employed both to extract UMI sequences and to collapse PCR duplicates as a baseline model. The processed reads were then aligned to the reference genome using the STAR aligner (Dobin et al. (2012)) for short reads and the Minimap2 aligner (Li (2018)) for long reads, which provided accurate alignment of the sequencing reads to their respective genomic locations.

Samtools (Li et al. (2009)) was utilized for manipulation and processing of Sequence Alignment/Map (SAM) and Binary Alignment/Map (BAM) files, including sorting, indexing, and filtering. The tool featureCounts (Liao et al. (2013)) was employed to quantify read counts for specific transcripts (different polyadenilation sites).

Several Python libraries were also used for data analysis and statistical computations. The pySam library (Li et al. (2009)) facilitated interaction with SAM/BAM files in Python, allowing for efficient data extraction and manipulation. The SciPy (Virtanen et al. (2020)) and NumPy (Harris et al. (2020)) libraries were employed for advanced statistical analysis and calculations, including matrix operations, applying machine learning algorithms and computing pairwise distances of custom functions. In addition, Plotly (Plotly Technologies Inc. (2015)) was used to automate plot generation to facilitate result visualization.

The specific versions of all of the above are listed on the appendix A.

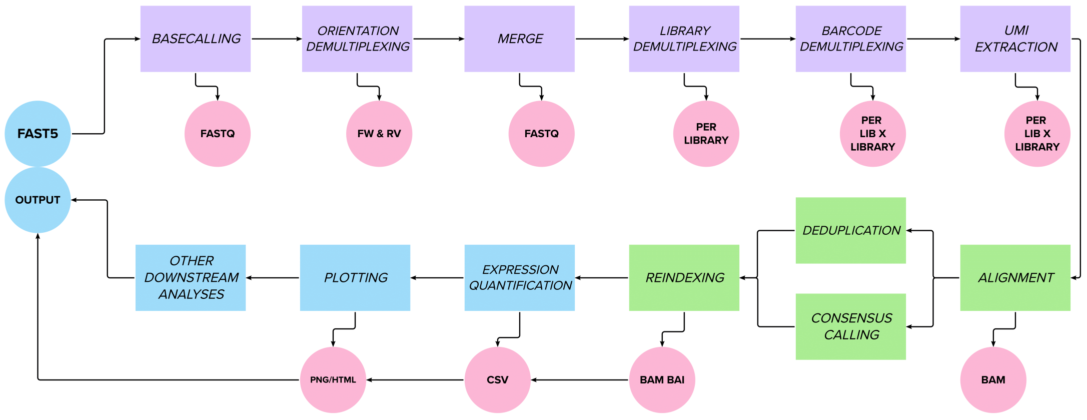

Figure 1: Flowchart of the processing pipeline. In purple, the preprocessing steps; in green the processing and in blue the downstream analyses.

5.4.3 Overview of the data analysis pipeline

Here, the whole preprocessing protocol (as shown in figure 1), as well as the mapping and downstream analysis are described. Bearing in mind the sample preparation steps, the final read structure would be as shown in figure 2, where it is also shown how the structure will vary along the preprocessing stage.

Note that after each of the preprocessing steps, quality control is carried out to get an insight into how the process affected the read distribution, quality histogram and read length.

Orientation demultiplexing The first step in the pipeline, once the basecalling has been performed, is to extract actual reads and infer their orientation. To do so, an approach similar to that of the demultiplexing process has been used with Cutadapt. Using the P5 and P3 (and their reverse complements) as linked adapters (i.e. $P5 - Sequence - P3$ and $P3' - RevSequence - P5'$), reads can be distinguished from artifacts and, depending on the matched adapter (forward or reverse), they can be sent to different file. Here, the error tolerance for the adapters to match is 20% (*-e 0.2*).

Once the orientation is known, a *SeqKit stat* is performed to check the distribution of the reads across both files, which should be close to 1:1. Then, the file containing reverse reads is *reverse complemented* and concatenated to the forward file. This step is required to ensure no information is lost. Ideally, pair-end files would be generated, but with this kind of sequencing only one of the two strands of each PCR product is sequenced, and such no real pair-end read can be generated. Instead, the pipeline will go on with single-end reads.

Library demultiplexing After a unique forward-oriented file is obtained, the next step is to demultiplex the sequenced libraries into files named after their own library. This is achieved by searching for a given list of i7 Illumina indexes on the trailing 5 nucleotides of each sequence. Here, Cutadapt is again used, and an error tolerance of 20% (i.e. 1nt) is

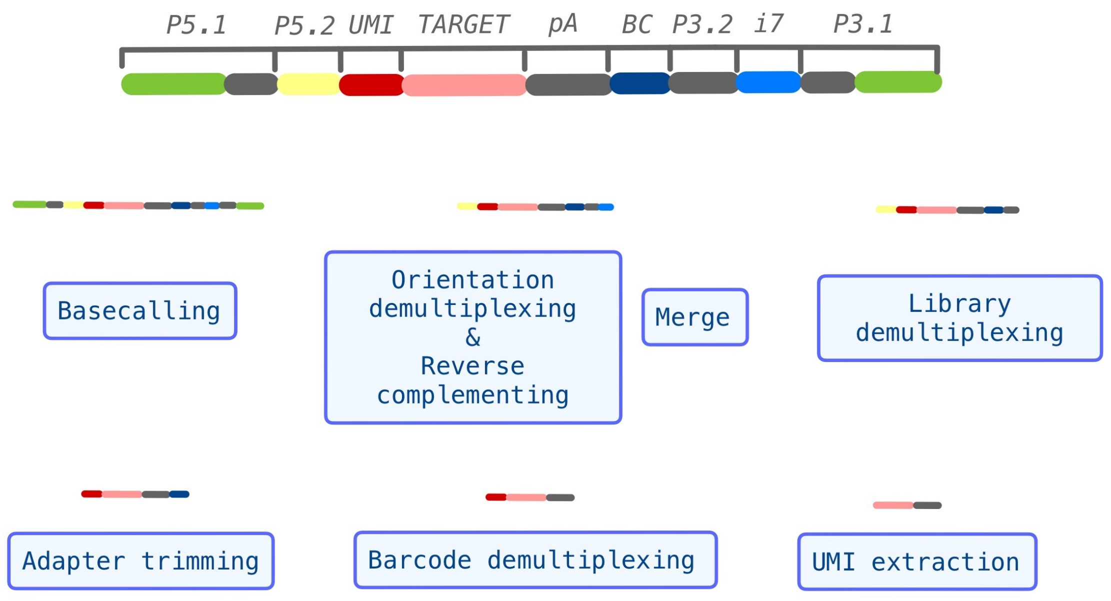

Figure 2: Read structure and evolution over preprocessing stage.

imposed, allowing for *indels*. It is not imperative to be strict yet, as the number of libraries is often limited and the probability of two different i7 indexes being similar is remarkably low.

Sample demultiplexing Once the reads are organized by library, the next thing to do is to trim the internal parts of the Illumina adapters. It is crucial not to allow *indels* in this step, as taking a nucleotide out of the barcode could lead to a virtual cross-contamination of samples.

After that, files are once again demultiplexed as done before, now using a list linking barcodes to each corresponding sample. In this case, no *indels* are allowed, and the error tolerance is again 20%.

UMI extraction As the final preprocessing step, UMIs are extracted using UMI-tools. This process is really straightforward, as the only required argument is the number of nucleotides making the UMI (10 nt in this case). From this point onwards, a single-end platform-independent pipeline is proposed.

First Alignment Now that the reads are distributed by library and sample, and that no artificial nucleotide is present inside the reads, the alignment is carried out using either STAR or Minimap2, depending on the expected length of the reads. In this experiment, the reference genome used was the *GRCh38.p14* with *Gencode Human v41* annotations.

The arguments passed to the aligner will also depend on the experiment, as a higher error rate and especially multimapping might be acceptable in some experiments. The output of this process will be a sorted-by-coordinate BAM file.

Deduplication and error correction Once the reads have been located in the genome, it is time to deduplicate the PCR duplicates by using the UMIs. Two alternatives are available:

UMI-tools creates clusters based on the exact genomic coordinates of a read (i.e. duplicates must have the exact same position and length) and its UMI (here errors are accepted). Once the reads are grouped, the read with the higher quality is selected, and the rest of the cluster is removed.

UMIclusterer creates clusters in the same fashion UMI-tools does, but is more flexible with the genomic location. Moreover, instead of taking one read from each cluster, it creates a consensus read from all the reads on it.

If using UMI-tools, the output will be a BAM file, which should be reindexed using Samtools. Thus, the second alignment would not be necessary and should be skipped. On the other hand, if using UMIclusterer, the output will be a new FASTQ file containing the consensus-called sequences, and a second alignment would be necessary.

Second Alignment After generating a consensus read for each cluster, the new FASTQ files should be aligned again with the same tool and parameters as in the first alignment performed. Then, the BAM file should once again be indexed and passed for downstream analysis.

Downstream Analysis Now that the final BAM files have been obtained, it is time to carry out the analyses required by the experiment. In this project, the goal is to quantify the expression of specific isoforms of the human HTT gene. For this task, featureCounts will be used. It will output the counts for each of the specified isoforms across all the aligned samples.

After the counting, the output is passed to a custom Python script that will generate plots of the gene expression grouped by tissue, sample and target.

This final part can be exchanged or followed by any other analysis process, namely differential expression analysis, variant calling or alternative splicing analysis, among others.

5.5 UMIclusterer

5.5.1 Overview

UMIcluster is a custom-developed software tool designed to address specific limitations in existing deduplication tools for targeted sequencing experiments, such as UMI-tools and Gencore. These existing tools do not allow duplicates to have different lengths or coordinates, which can be problematic in certain scenarios. UMIcluster aims to overcome this limitation by providing an alternative approach to deduplication. A comparison table between UMIclusterer and other tools is shown in table 3.

One key feature of UMIcluster is the generation of consensus reads for each cluster. Unlike UMI-tools, which removes duplicates entirely, UMIcluster generates a consensus read that concentrates the most accurate representation of the duplicated sequences within a cluster. This consensus read provides a more reliable and accurate representation of the underlying target sequence.

UMIcluster is implemented entirely in Python, utilizing the C-written library of pySAM, which is an implementation of samtools. This combination of Python and pySAM allows for efficient and scalable processing of the sequencing data, enabling UMIcluster to handle large datasets.

The primary objective of UMIcluster is to offer an alternative method for creating consensus reads of PCR duplicates in Nanopore targeted sequencing experiments. By employing UMIcluster, researchers can enhance the accuracy of the sequencing data above the standard quality achieved by Nanopore sequencing alone. This improved accuracy is particularly valuable in applications where precise sequence information is crucial, such as in variant calling or downstream analysis.

Table 3: Comparison between UMIclusterer and other deduplication tools.

	UMIclusterer	**UMI-tools**	**Gencore**
Error correction	Yes	No	Yes
UMI error tolerance	Yes	Yes	Yes
Flexible read position	Yes	No	No
Single-end compatible	Yes	Yes	No

5.5.2 Algorithm implementation

BAM parsing The first step to take is to parse the input BAM files. All the mapped reads are converted to a custom data-type and grouped by contig. This will allow to process each group separately, as no cluster can have reads in more than one chromosome.

Clustering Now, each group is parsed in parallel. All the reads are now indexed, and a distance matrix D containing the pairwise distance between all the reads is calculated. Given $th_{umi} \geq 0$ (the threshold to consider two UMIs the same), and $th_{window} \geq 0$ (the window around *start* and *stop* coordinates to consider two reads potentially the same), the distance $d_{1,2}$ between two sequences $s1$ and $s2$ is defined as follows:

$$d_{1,2} = d_{umi} + d_{coord}$$

d_{umi} is calculated using the *Hamming distance* between both UMIs:

$$h = \sum_{i=1}^{n} (UMI(s1)_i \neq (UMI(s2)_i)$$

$$d_{umi} = \begin{cases} h, & \text{if } h \leq th_{umi} \\ 999, & \text{otherwise} \end{cases}$$

And d_{coord} is calculated from the mean distance between both sequences' *start* and *end* coordinates:

$$w = \frac{|s2_{start} - s1_{start}| + |s2_{end} - s1_{end}|}{2}$$

$$d_{coord} = \begin{cases} w, & \text{if } w \leq th_{window} \\ 999, & \text{otherwise} \end{cases}$$

This will generate a matrix:

$$D = \begin{bmatrix} 0 & d_{1,2} & \dots & d_{1,N} \\ d_{2,1} & 0 & \dots & d_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ d_{N,1} & d_{N,2} & \dots & 0 \end{bmatrix}$$

where N is the number of sequences within the cluster.

Once the matrix is computed, it will be fed to a *hierarchical clustering* algorithm with complete linkage, included in the Scipy library. Here, the threshold to consider a cluster will be the sum of both UMI and coordinate thresholds ($th = th_{umi} + th_{coord}$). When grouped, the reads will be back-traced, indexed and grouped inside a hash table, where the key is the cluster ID and the value is the contained reads' IDs.

Consensus calling Finally, after clustering the PCR duplicates, the consensus reads for each cluster are called. The consensus generation process involves several key steps:

1. **Padding with wildcards:** To account for deletions in the reads, they are padded with wildcard characters. This padding leverages the CIGAR generated by the aligner where different aligning events are described, as explained by Li et al. (2009).
2. **Reference-Based Alignment:** The longest read within each cluster is selected as the reference, and the remaining reads within the cluster are aligned to it. This alignment is performed using a custom implementation of the Needleman-Wunsch global sequence alignment algorithm (Needleman and Wunsch (1970)). During the alignment, wildcards are inserted as necessary to maintain the alignment.
3. **Base and Quality Extraction:** For each position in the aligned sequences, the bases and their corresponding basecalling quality scores are extracted. The basecalling quality scores are represented in the Phred scale (Ewing et al. (1998)), which assigns a score $Q = -10 \log_{10} E$ to each base, where E is the probability of the base being incorrect.

4. **Probability Score Computation:** Using the extracted bases and their quality scores, the consensus score of each nucleotide (including the possibility of a deletion) is computed as the sum of its frecuency score and quality score:

$$c = freq_{score} + qual_{score}$$

$freq_{score}$ is equal to the fraction of the bases that an individual base represents times 10 (i.e. a base that does not appear will score 0 and a base appearing in half of the reads a 5). The scoring method for $qual_{score}$ can be found in table 4.

5. **Consensus Base Selection:** The highest-rated base is selected as the consensus base for each position. Additionally, the highest Phred score among the bases in that position is chosen as the consensus quality score.

6. **Export to FASTQ:** Finally, all the consensus reads generated across the clusters contained in the BAM are exported to a FASTQ file. These files are then ready to undergo further realignment or downstream analysis.

Table 4: Basecalling quality scoring table for the consensus reads.

Condition	**Score**
$Q \geq 30$	8
$Q \geq 20$	6
$Q \geq 15$	4
$Q < 15$	2

5.5.3 Parameters

Bearing in mind the aforementioned process, there are two tunnable parameters that will determine the *greediness* of the tool:

UMI threshold this parameter sets the maximum allowed Hamming distance between two UMIs to be considered the same. The Hamming distance between two sequences is the number of individual variations needed to perform on one sequence to make it identical to the other one.

Coordinate window this parameter fixes the mean distance between the *start* coordinates and between the *end* coordinates of both sequences to be considered to come from the same original read.

Both parameters should be tweaked according to the preparation and nature of our samples. With longer UMIs, it seems reasonable to increase the threshold as well. The window size will also depend on the length of our target sequences and on the presence of low complexity regions in our targets (e.g. poly(A) tails). The longer the reads and with more/longer low complexity regions, the bigger the window size should be.

Note that both conditions must be met at the same time in order to include both reads in the same cluster. If the reads share location but have different UMIs, or share UMI but are located in different regions, they will be separated onto different clusters.

5.5.4 Computational resources and infrastructure

The memory requirements of the tool are relatively low, only needing around a 120% of the loaded BAM file's size as available RAM memory. It is, however, quite computationally intensive, as a lot of looping and matrix operations are performed throughout the process.

The ideal setup to run UMIclusterer is a HPC cluster with multiple available nodes or processors, as it will be easy to run many instances of the tool in parallel to process a large number of files. In this specific project, the tool was executed on a HPC cluster as part of the above mentioned pipeline.

5.6 Statistical Analysis

To assess the performance and mapping accuracy of the UMIclusterer tool in comparison to other deduplication methods, a comprehensive evaluation method was employed. The evaluation involved multiple runs on the same dataset, utilizing the duplicated data as a baseline for comparison against UMI-tools and UMIclusterer.

The primary focus of the evaluation was to analyze the CIGAR strings of all the reads within the BAM files generated from the sequencing experiment. The CIGAR strings provide valuable information about the alignment and sequence variations present in the reads. By examining the CIGAR strings, key metrics were derived to quantify the mapping accuracy and potential improvements achieved by the corrected reads.

The assessed metrics were:

- the mismatch rate (mismatches per kilobase)
- the indel rate (insertions and deletions per kilobase)
- the overall error rate (indel rate + mismatch rate)

Not only will these metrics evaluate the performance of each tool, they will also provide insights into the nature of the results by showing where the improvements are made.

6 RESULTS

The developed pipeline successfully achieved its objectives of analyzing canonical and cryptic polyadenylation events in the studied samples. By implementing the UMI-clusterer tool and employing a combination of advanced bioinformatics methods, the pipeline demonstrated its effectiveness in improving mapping accuracy and reducing indel rates.

Furthermore, the pipeline was designed with a strong focus on ensuring reproducibility and deployability. Docker containers were utilized to encapsulate all the necessary software dependencies, allowing for easy replication of the analysis environment across different systems. For high-performance computing (HPC) clusters, Singularity containers were employed, ensuring compatibility and seamless execution on various computing infrastructures. The pipeline's scalability was another key feature, made possible through the utilization of Nextflow, a workflow management system. Nextflow enabled the pipeline to handle larger datasets and efficiently distribute computational tasks across multiple compute nodes. This scalability ensured the pipeline's adaptability to diverse experimental requirements and datasets of varying sizes. An implementation is available on GitHub (link in appendix B).

Table 5: Summary of accuracy, *indel* rate and mismatch rate across human samples.

Group	**Accuracy [%]**	**Indels [indel/kB]**	**Mismatches [mm/kB]**
Baseline	98.1758% ± 0.1080	4.4996 ± 0.4273	12.3752 ± 1.2029
UMI-tools	98.1779% ± 0.1327	4.2763 ± 0.3137	12.0750 ± 1.0249
UMIclusterer	98.2414% ± 0.0985	3.8515 ± 0.4205	12.2367 ± 1.4620

Table 6: Summary of accuracy, *indel* rate and mismatch rate across mouse samples.

Group	**Accuracy [%]**	**Indels [indel/kB]**	**Mismatches [mm/kB]**
Baseline	98.6001% ± 0.0811	4.1127 ± 0.3456	9.3562 ± 1.9694
UMI-tools	98.5290% ± 0.1409	3.8565 ± 0.4079	11.4077 ± 2.1912
UMIclusterer	98.6167% ± 0.2057	2.3749 ± 0.3875	11.5109 ± 2.8434
Illumina	99.4591% ± 0.0643	0.0401 ± 0.0589	5.3693 ± 0.6416

On the other hand, the performance of UMIclusterer was also tested. In the analysis of mouse samples, Illumina sequencing demonstrated superior performance, as expected, surpassing other methods by a significant margin, as can be observed in table 6. Among the Nanopore-based approaches, the best mean accuracy achieved was 98.61%, which outperformed the 96% accuracy offered by Nanopore itself with super high accuracy (SUP) basecalling. In human cell samples, the best mean accuracy achieved was 98.24%. Notably, the reads in the human samples appeared to be noisier compared to those from the mouse samples. Despite the differences between experiments, UMIclusterer stands as the best-performing tool among the baseline model, UMI-tools and UMIclusterer itself, as shown in tables 5 and 6.

When comparing error rates, it was observed that clustering with UMIclusterer yielded better results than deduplication using UMI-tools. While neither method improved mismatch rates (which remain similar to those of the baseline model), both demonstrated improvements

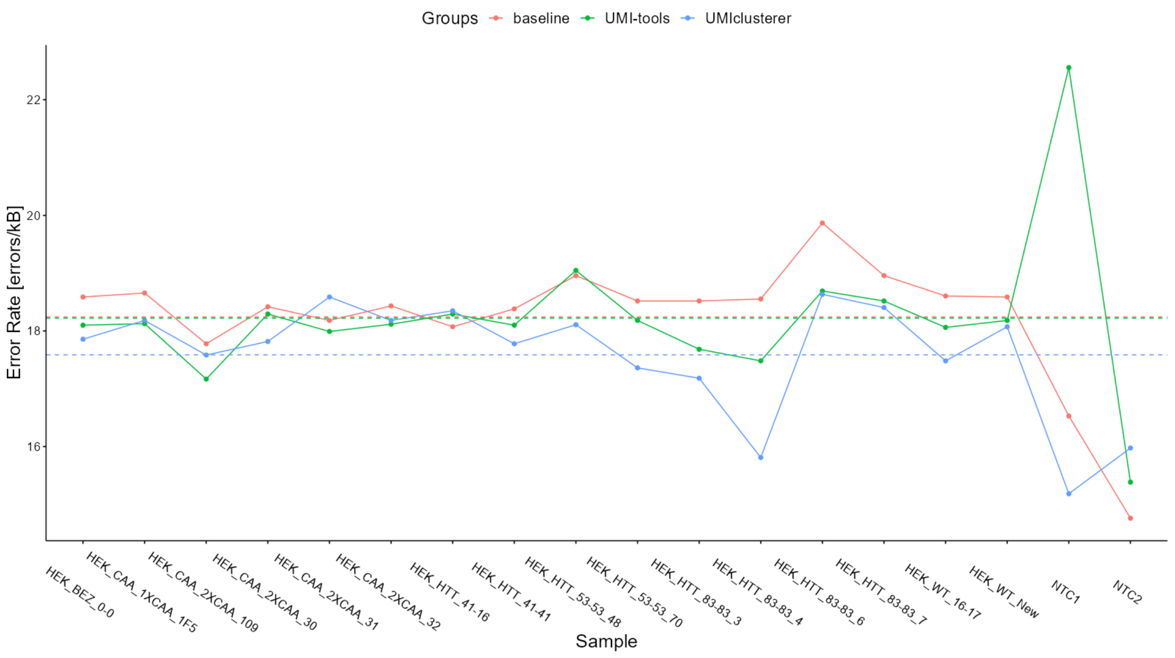

Figure 3: Error rate distribution of the 3 groups across human samples. Dashed lines represent the mean value of each group.

in the *indel* rate. Particularly, the clustering approach showed to be good in reducing *indel* rates compared to the other methods employed. A more close-up image of these aspects can be achieved by looking at figures 3 to 8. It can be observed that, while the total error rate is similar between methods, UMIclusterer is consistently better at tackling the insertion/deletion problem (specially evident in figure 6). However, the remarkable improvement when it comes to *indels* is masked by the not-so-good performance when treating mismatches.

Finally, the effectiveness of UMIclusterer against UMI-tools was also statistically tested using *t-test* contrasts. As shown by table 7, the effect of UMIclusterer is statistically significant both for error-rate reduction and *indel* reduction, reducing up to 1.5 *indels* per kilobase. It is important to note that, however, the error-rate is the sum of the insertion/deletion rate and the mismatch rate and, as the latter was by no means significant, the results are masked by it.

Table 7: Paired t-test results on sample data (UMIclusterer vs UMI-tools).

Contrast	**Mean difference**	**P-value** $\vert < \vert$
Error rate	-0.8763481	$3.453 \cdot 10^{-05}$
Indel rate	-1.481594	$2.2 \cdot 10^{-16}$
Mismatch rate	0.1032	0.7375

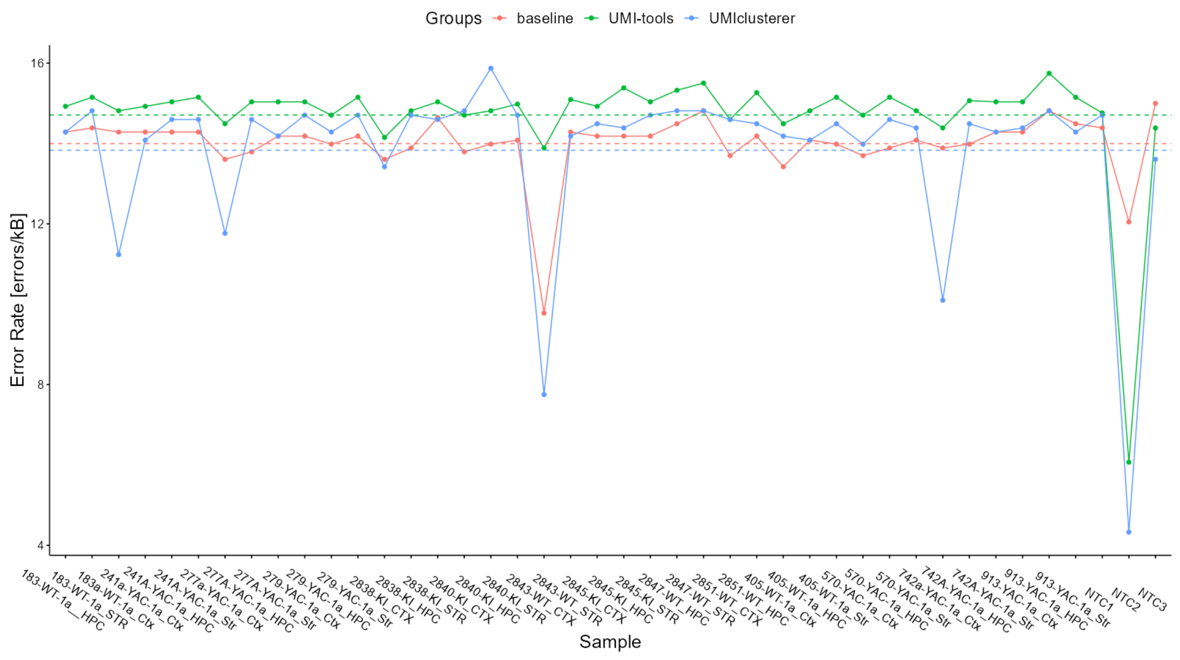

Figure 4: Error rate distribution of the 3 groups across mouse samples. Dashed lines represent the mean value of each group.

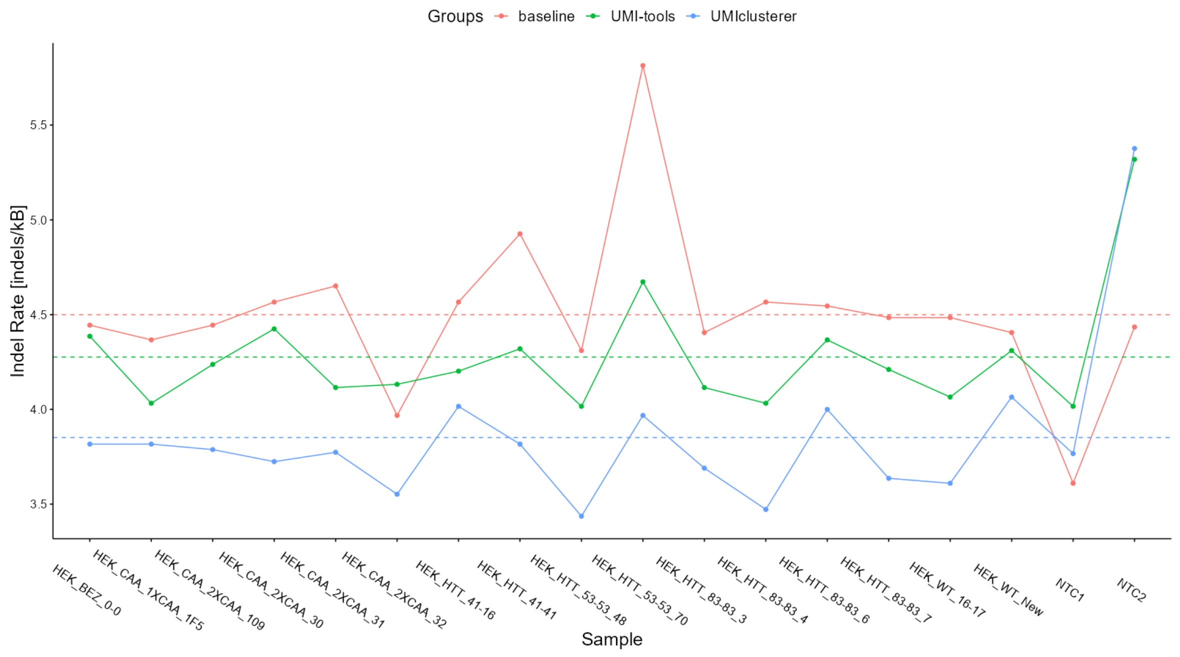

Figure 5: Insertion/deletion rate distribution of the 3 groups across human samples. Dashed lines represent the mean value of each group.

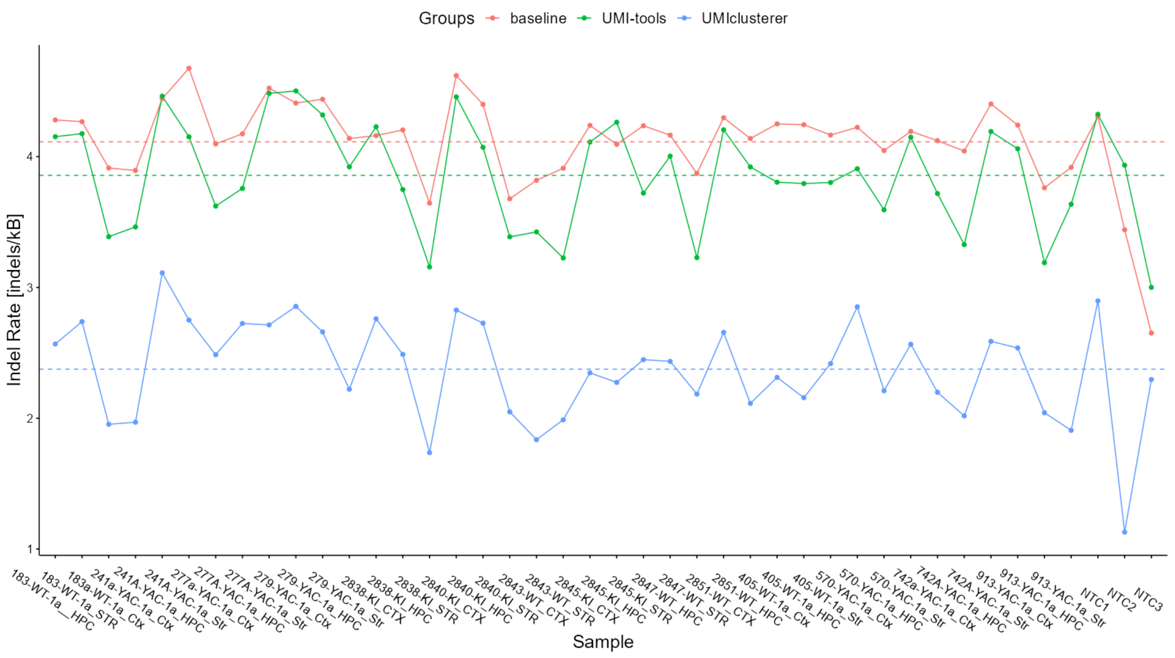

Figure 6: Insertion/deletion rate distribution of the 3 groups across mouse samples. Dashed lines represent the mean value of each group.

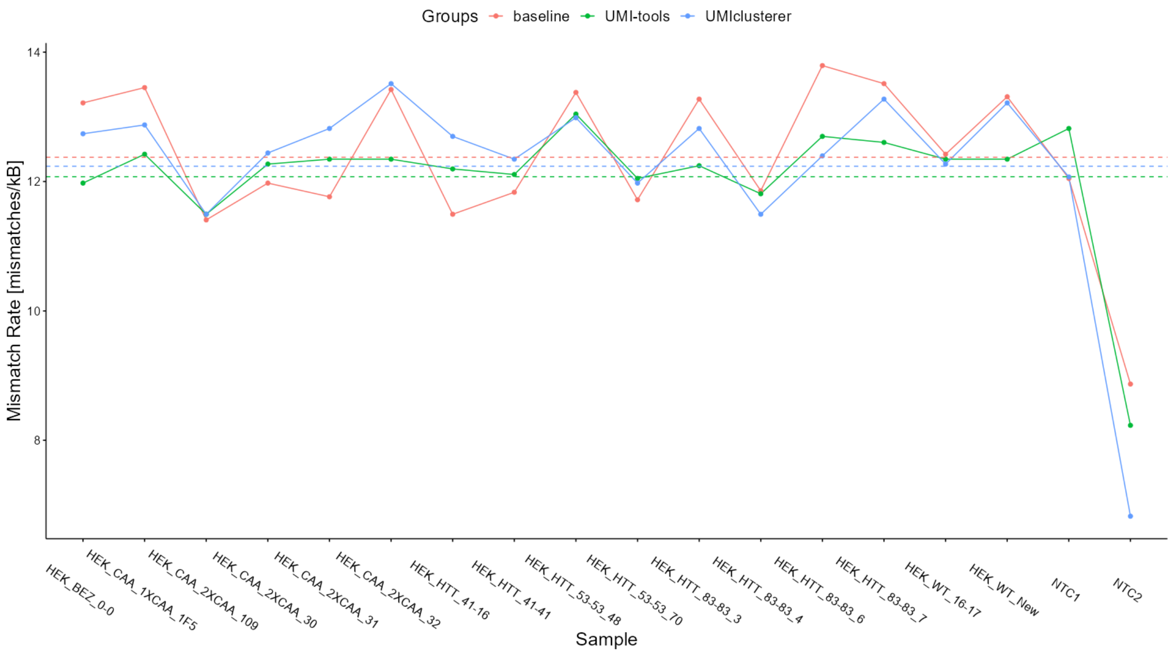

Figure 7: Mismatch rate distribution of the 3 groups across human samples. Dashed lines represent the mean value of each group.

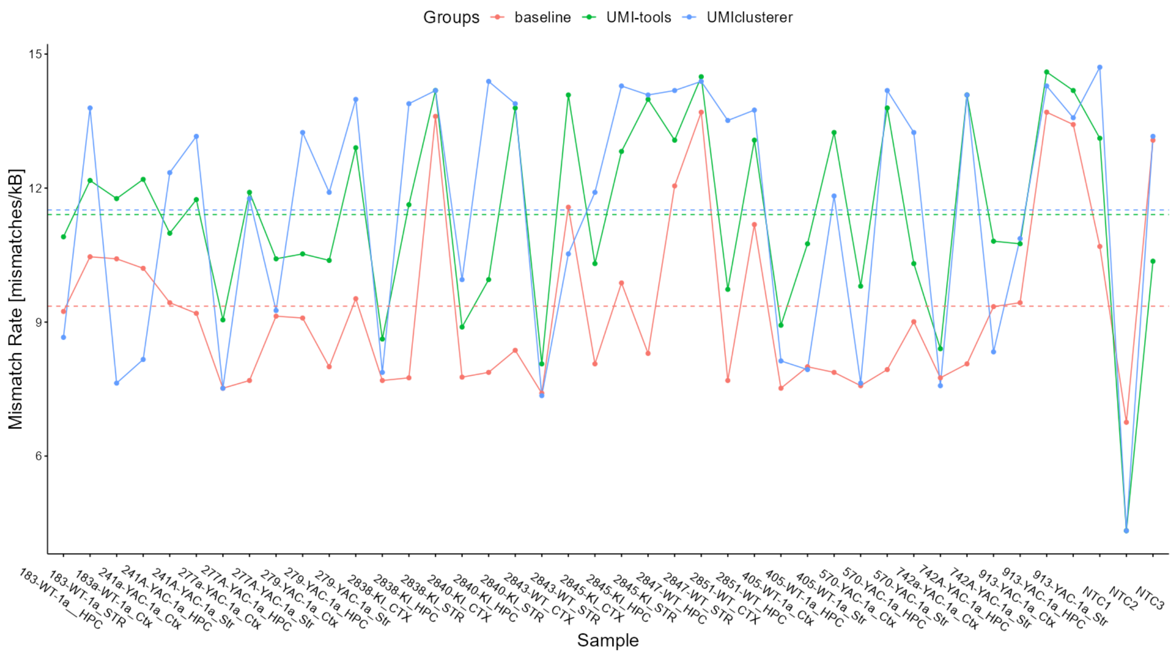

Figure 8: Mismatch rate distribution of the 3 groups across mouse samples. Dashed lines represent the mean value of each group.

7 DISCUSSION

Overall, it is important to note that both experiments (human and mouse) exhibited low coverage. Consequently, it is expected that both UMIclusterer and UMI-tools would demonstrate even better performance with higher-depth sequencing. These findings highlight the potential of UMIclusterer as a valuable tool for improving mapping accuracy and overall quality in targeted sequencing experiments, particularly when addressing *indel* rate reduction.

The strict criteria applied during the clustering process, including the elimination of reads that do not meet the specified thresholds, resulted in a significant reduction in the number of available reads. While this strict filtering ensures higher confidence in the resulting consensus reads, it is essential to acknowledge the trade-off between stringency and the total number of reads obtained. Researchers must carefully consider their specific experimental objectives and sequencing depth requirements when selecting the appropriate level of stringency for preprocessing and demultiplexing.

Furthermore, it was observed that the quality of basecalling plays a crucial role in the number of reads retained during the analysis. With improved basecalling quality, a larger number of reads pass the minimum quality cutoff of the basecalling system and can be preserved for downstream analysis. This highlights the importance of the continuous advancements in basecalling technology.

It is also relevant to consider that the quality of sequencing of the barcode region may be higher in reverse oriented strands, where it is placed before the polyadenylation site, and thus not affecting the quality of the barcode. This could lead to some biases and should be taken into account when interpreting the results.

By acknowledging the influence of basecalling quality, striking a balance between stringency and read retention, and staying updated with advancements in basecalling technology, researchers can optimize the utilization of this pipeline, as well as UMIclusterer and other deduplication methods to achieve more accurate and comprehensive insights into transcriptomic data through Nanopore sequencers.

On a different note, results showed that the main strength of UMIclusterer lies on its ability to tackle artificial insertions and deletions, while it poorly performs when it comes to mismatches. With this in mind, modifications in the algorithm, such as balancing the weights, could lead to further improvements. Moreover, consensus reads could also be realigned to the original reads with modified weights to try and reduce the mismatches without losing the *indel* reduction. This idea stems from the fact that, as observed in figure 8, the original non-deduplicated reads had better mismatch scores and so, there is still room for improvement.

8 CONCLUSION

In this project, a comprehensive pipeline for the analysis of polyadenylation events using targeted sequencing was developed. The pipeline integrated state-of-the-art bioinformatics tools and methodologies to improve mapping accuracy, reduce *indel* rates, and obtain high-quality consensus reads.

Through the evaluation of different deduplication approaches, the effectiveness of the UMIclusterer tool in achieving accurate mapping and reducing *indel* rates was demonstrated. The clustering approach offered a valuable alternative to existing deduplication tools by allowing duplicates with different lengths and generating consensus reads for each cluster. This innovation significantly enhanced the accuracy of the sequences, surpassing the standard quality offered by Nanopore technology.

The analysis of mouse and human samples provided valuable insights into the performance of the pipeline. While Illumina sequencing exhibited superior accuracy, the Nanopore-based approaches, particularly with the application of UMIclusterer, showed substantial improvements in accuracy and *indel* rate reduction, which will get increasingly more performant as basecalling software evolves. It is worth noting that the coverage in both experiments was relatively low, highlighting the potential for even greater performance with higher-depth sequencing.

To sum up, this project contributes to the field of Nanopore and Illumina sequencing analysis by providing a robust and scalable. The results demonstrate the potential of UMIclusterer as a valuable tool for improving mapping accuracy and reducing *indel* rates in targeted sequencing experiments. Further research and optimization of the pipeline, along with advancements in basecalling technology, are expected to yield even more accurate and comprehensive results.

REFERENCES

Andrews, S., Krueger, F., Segonds-Pichon, A., Biggins, L., Krueger, C., and Wingett, S. (2012). FastQC. Babraham Institute.

Dobin, A., Davis, C. A., Schlesinger, F., Drenkow, J., Zaleski, C., Jha, S., Batut, P., Chaisson, M., and Gingeras, T. R. (2012). STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, 29(1):15–21.

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

Ewing, B., Hillier, L., Wendl, M. C., and Green, P. (1998). Base-calling of automated sequencer traces using phred. i. accuracy assessment. *Genome Research*, 8(3):175–185.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825):357–362.

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

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., Durbin, R., and Subgroup, . G. P. D. P. (2009). The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16):2078–2079.

Liao, Y., Smyth, G. K., and Shi, W. (2013). featurecounts: an efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics*, 30(7):923–930.

Martin, M. (2011). Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.journal*, 17(1):10–12.

Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453.

Plotly Technologies Inc. (2015). Collaborative data science.

Shen, W., Le, S., Li, Y., and Hu, F. (2016). Seqkit: A cross-platform and ultrafast toolkit for fasta/q file manipulation. *PLOS ONE*, 11(10):1–10.

Smith, T., Heger, A., and Sudbery, I. (2017). UMI-tools: modeling sequencing errors in unique molecular identifiers to improve quantification accuracy. *Genome Res*, 27(3):491–499.

The Huntington's Disease Collaborative Research Group (1993). A novel gene containing a trinucleotide repeat that is expanded and unstable on huntington's disease chromosomes. *Cell*, 72(6):971–983.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

A SOFTWARE VERSIONS

Summary of the third-party software used, including its versions.

Tool	Purpose	Version
FastQC	Raw read QC	0.12.1
MultiQC	QC report generation	1.14
SeqKit	FASTQ file operations	2.4.0
cutadapt	Demultiplexing, trimming and filtering	4.4
UMItools	UMI extraction and deduplication	1.1.4
STAR	Short-read mapping	2.7.10b
Minimap2	Long-read mapping	2.26
SAMtools	BAM manipulation	1.16.1
featureCounts	Transcription quantification	2.0.3
pySAM	Python BAM manipulation	0.21.0
SciPy	Python ML and algorithm library	1.10.0
NumPy	Python matrices	1.23.5
Nextflow	Parallel process orchestrator	22.10.6 build 5843
Guppy	Nanopore basecalling	6.5.7

Table 8: Versions of the software used in this project.

B SOFTWARE LINKS

A list of the cloud-hosted resources used in the project. The pipeline and UMIclusterer repos are available through the following links:

Illumina2Nanopore pipeline

UMIclusterer deduplication and consensus calling tool

A list of Docker images containing the dependencies of the project are also available on DockerHub:

Guppy-GPU standalone Nanopore basecaller and CUDA toolkit.

FastQC includes the QC software.

MultiQC includes the reporting software.

UMIclusterer contains the tool and all of its dependencies.

Demultiplex contains all the demultiplexing and deduplication software, including cutadapt, UMItools and SeqKit.

Minimap2 contains the aligner.

STAR contains the aligner.

Subread contains featureCounts and SAMtools.

Plotly contains Plotly and other required tools for it to work, such as NumPy or Pandas.