Advanced Methods for Real-time Metagenomic Analysis of Nanopore Sequencing Data

Dissertation zur Erlangung des Grades
eines Doktors der Naturwissenschaften (Dr. rer. nat.)
am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

vorgelegt von

JENS-UWE ULRICH

Berlin 2023

Betreuer:	Prof. Dr. Bernhard Renard
Erstgutachter:	Prof. Dr. Bernhard Renard
Zweitgutachter:	Prof. Dr. Sven Rahmann
Tag der Disputation:	19.01.2024

Abstract

Whole shotgun metagenomics sequencing allows researchers to retrieve information about all organisms in a complex sample. This method enables microbiologists to detect pathogens in clinical samples, study the microbial diversity in various environments, and detect abundance differences of certain microbes under different living conditions. The emergence of nanopore sequencing has offered many new possibilities for clinical and environmental microbiologists. In particular, the portability of the small nanopore sequencing devices and the ability to selectively sequence only DNA from interesting organisms are expected to make a significant contribution to the field. However, both options require memory-efficient methods that perform real-time data analysis on commodity hardware like usual laptops.

In this thesis, I present new methods for real-time analysis of nanopore sequencing data in a metagenomic context. These methods are based on optimized algorithmic approaches querying the sequenced data against large sets of reference sequences. The main goal of those contributions is to improve the sequencing and analysis of underrepresented organisms in complex metagenomic samples and enable this analysis in low-resource settings in the field.

First, I introduce ReadBouncer as a new tool for nanopore adaptive sampling, which can reject uninteresting DNA molecules during the sequencing process. ReadBouncer improves read classification compared to other adaptive sampling tools and has fewer memory requirements. These improvements enable a higher enrichment of underrepresented sequences while performing adaptive sampling in the field. I further show that, besides host sequence removal and enrichment of low-abundant microbes, adaptive sampling can enrich underrepresented plasmid sequences in bacterial samples. These plasmids play a crucial role in the dissemination of antibiotic resistance genes. However, their characterization requires expensive and time-consuming lab protocols. I describe how adaptive sampling can be used as a cheap method for the enrichment of plasmids, which can make a significant contribution to the point-of-care sequencing of bacterial pathogens. Finally, I introduce a novel memory- and space-efficient algorithm for real-time taxonomic profiling of nanopore reads that was implemented in Taxor. It improves the taxonomic classification of nanopore reads compared to other taxonomic profiling tools and tremendously reduces the memory footprint. The resulting database index for thousands of microbial species is small enough to fit into the memory of a small laptop, enabling real-time metagenomics analysis of nanopore sequencing data with large reference databases in the field.

Acknowledgements

First and foremost, I would like to thank my supervisor Bernhard Renard for providing the opportunity to work on this exciting project in his research group. Thank you for all the freedoms I had throughout all stages of my projects, but also for giving the best possible support and taking the time for extensive discussions whenever needed. Thank you for giving me all the flexibility a father of three children needed during the Corona pandemic. And last but not least, thank you for being such a great mentor during the Junior Teaching Professionals program. I could not have wished for a better supervisor for my PhD journey.

Second, but by no means less important, I thank my wife, Steffi, for all her support, patience, encouragement, and understanding that made this adventure possible. Thank you for your love and for being there. I also want to thank my three lovely daughters, Sophie, Jasmin & Jette, for listening to all my scientific explanations and sustaining my stupid jokes throughout the years. It's a pleasure to see you grow up.

I am also grateful to all other members of the Renard lab, also known as MF1 and DACS. In particular, Jakub Bartoszewicz for fruitful scientific discussions and providing honest feedback; Tobias Loka and Vitor Piro for discussions on metagenomics, pathogen detection and data structures; Henning Schiebenhöfer, Tom Altenburg, Marta Lemanczyk and Susanne Ibing for being such great office mates; Nina Ihde and Ahmad Lutfi for joining me in exploring many different aspects and applications of this work as students; Elizabeth Yuu, Ferdous Nasri, Katharina Baum, Christoph Schlaffner, Simon Witzke, Pascal Iversen and everyone else in the group for lunches, coffee breaks and lots of cake, with all the inspiring conversations about scientific and non-scientific topics.

Of course, my research was also supported by many collaboration partners. Thanks to Prof. Knut Reinert and the SeqAn team for providing excellent support as well as fantastic workshops and discussions. Thanks also to the Genome Sequencing and Genomic Epidemiology group at the Robert-Koch Institute for providing the DNA sequencing infrastructure. In particular, I want to thank Torsten Semmler, Lennard Epping, Andrea Thürmer, Aleksandar Radonic, Tanja Pilz, Kilian Rutzen, Birgit Walther, and Kerstin Stingl for providing samples and performing all the wet lab work that was crucial for the real-time nanopore sequencing experiments presented in this work. Further, I would like to thank Prof. Sven Rahmann for agreeing to review my thesis.

Finally, I would like to thank my family, in particular my parents, my parents-in-law, and my brother, for their unconditional support. Thank you for your love, sacrifices, and encouragement and for supporting my educational endeavors.

Contents

List of Abbreviations

AS adaptive sampling

AMQ approximate membership query

AMR antimicrobial resistance

ARG antimicrobial resistance gene

API Application Programming Interface

bp base pairs

CPU central processing unit

DNA deoxyribonucleic acid

DP dynamic programming

EM expectation maximization

GB GigaByte

Gbp Giga base pairs

GHz GigaHertz

gRPC Google Remote Procedure Call

GPL-3.0 GNU General Public License 3

GUI Graphical User Interface

GPU graphics processing unit

HIBF hierarchical interleaved Bloom filter

HIXF hierarchical interleaved XOR filter

HGT horizontal gene transfer

IBF Interleaved Bloom Filter

IXF Interleaved XOR Filter

kbp kilobase pairs

MCC Matthews correlation coefficient

MEMs maximal exact matches

MGE mobile genetic element(s)

mL milliliter(s)

mV millivolts

µL microliter(s)

ng nanogram(s)

ONT Oxford Nanopore Technologies

PacBio Pacific Biosciences

PCR Polymerase Chain Reaction

RAM random access memory

RSS resident set size

RNA ribonucleic acid

1 Introduction

Researchers in all domains of life sciences struggle with the ever-increasing amount of data generated from new technologies. This is also true for biology and biotechnology, where, for instance, decreasing costs in deoxyribonucleic acid (DNA) sequencing resulted in tons of new data being produced every year. Analyzing these large data sets requires fast and space-efficient computer programs, particularly if results should be provided in real-time. Presenting advanced methods for solving this issue for microbial identification and metagenomics is the primary goal of this thesis. However, before describing the methods, I will introduce some biological background on microorganisms and how we can use the latest DNA sequencing methods to identify microbial organisms. Finally, in this chapter, I will briefly introduce current state-of-the-art algorithms and data structures used to analyze microbial DNA sequencing data.

1.1 Biological background on microorganisms

Microbes and viruses are everywhere. They inhabit diverse environments like soil (Hermans et al., 2017), oceans (Sogin et al., 2006), and other organisms (Douglas, 2019), and they can survive even under the most extreme conditions, e.g., in hot springs (Ward et al., 1998), on volcanic rock (Staudigel et al., 2008), or in extremely salty water (Oren, 2008). Microbes are defined as small, microscopic organisms, such as bacteria, but there are also other organisms that fall under the characterization of "microbe", like archaea, protozoa, and some fungi (Sanz, 2011). For most scientists, viruses do not count as microbes because viruses are often classified as non-living (Villarreal, 2004). Microbes and viruses have been around for approximately 3.5 billion years, and they are so numerous that they are considered the secret rulers of the Earth (Knoll, 2015).
Microbes, also known as microorganisms, play an integral role in almost every natural process. They break down organic matter from plants and animals (Kirchman, 2018), releasing chemicals like carbon (Gougoulias et al., 2014), nitrogen (Aislabie et al., 2013), and phosphorus (Pingale & Virkar, 2013) that can be used to build new plants and animals. Microorganisms help generate oxygen (Hess, 2004) and carbon dioxide (Smith et al., 2019) and fix atmospheric nitrogen into usable forms for multiple

organisms (Gupta et al., 2017). Both animals and plants are closely associated with microbial communities that make nutrients more available, provide disease protection, make essential vitamins, or combine both (Stark, 2010). For example, some microbes in the human gut produce essential micronutrients like vitamin K, allowing humans to digest and absorb them (Fijan, 2014). Moreover, microbes also play an essential role in the food industry. Many food products, including bread, yogurt, cheese, preserves, preserved meats, and alcoholic beverages, take advantage of microbes and their chemical reactions (Kalsoom et al., 2020).

Although more than 99% of all microbes are harmless or even useful (Ladizinski et al., 2014), numerous diseases in humans, plants, and animals are caused by microorganisms (Gerba & Smith, 2005; Malmstrom et al., 2022; Mir, 2022). Pathogenic microbes and their causing diseases have accompanied humans since the beginning of history, with Leprosis, which is caused by the bacterium *Mycobacterium leprae*, considered one of the oldest infectious diseases in human history (Robbins et al., 2009). The most fatal pandemic - the "black death" - which was caused by the bacterium *Yersinia pestis* killed more than 25 million people in fourteenth-century Europe (Glatter & Finkelman, 2021). Nowadays, pathogenic microbes like the bacterium *Mycobacterium tuberculosis* and viruses like *SARS-CoV-2* kill millions of people yearly, posing a major burden on public health systems worldwide (Chakaya et al., 2021). With the ongoing climate changes and further destruction of natural habitats of wild-living animals, scientists expect an increasing number of zoonotic transmissions for the next decades (Daszak et al., 2001; Estrada-Peña et al., 2014; Naicker, 2011), which increases the need for tools that reliably detect and identify microbes in different kinds of samples. One method that has become very popular for this purpose is high-throughput sequencing, which will be described in more detail in section 1.2.

The development of antimicrobial drugs has revolutionized the treatment of infectious diseases and saved millions of lives since the discovery of penicillin by Alexander Fleming in 1929 (Coates et al., 2002; Fleming, 1941). In particular, antibiotics have proven to be an effective weapon against pathogenic bacteria (Nicolaou & Rigol, 2018). However, adaption processes in the bacteria and the overuse of antibiotics, particularly in farmed animals, have led to the dissemination of drug-resistant bacteria (Ventola, 2015). For example, one widespread antibiotic mechanism found in bacteria is efflux pumps, which can transport antibiotics from the inside to the outside of bacterial cells (Nishino et al., 2021). These acquired resistances are coded as antimicrobial resistance genes (ARGs) in the bacterial genome and can be transmitted from one bacterial species to another (Manaia, 2017; Martínez et al., 2015). An important consideration for human health and the evolution of antibiotic-resistant pathogens is ARGs moving by horizontal

gene transfer (HGT) from nonpathogens to pathogens (Ellabaan et al., 2021; Groussin et al., 2021). One of the main drivers of HGT are plasmids, which are circular epichromosomal DNA elements unique to bacteria. These mobile genetic elements (MGEs) can be transferred within and between bacterial species via a process called conjugation (Norman et al., 2009). The contribution of plasmids to the dissemination of ARG makes their identification and characterization an important aspect of clinical metagenomics (Brolund & Sandegren, 2016). High-throughput sequencing has already helped us to understand the distribution of ARGs and their hosts in specific habitats (Danko et al., 2021; Hendriksen et al., 2019). However, with the ongoing antibiotics crisis, advanced methods are needed that help to cut the costs of plasmid sequencing and improve the characterization of ARG-harboring plasmids.

In addition to symbiotic and pathogenic microorganisms, the human body is also associated with opportunistic pathogens that do not cause diseases under normal circumstances (Brown et al., 2012). These commensals only infect the human host if the host's immunity is impaired, for example, with infection by another pathogen. Here, the community of beneficial microbes serves as a barrier and protects against the colonization of opportunistic and non-opportunistic pathogens (Frost et al., 2020; Plesniarski et al., 2021). However, recent studies have found that changes in the composition of microbial communities (microbiome) in the human body can lead to dysbiosis and affect human health tremendously (Kumamoto et al., 2020; Yu, 2018). Furthermore, investigations of the relationship between the environmental and human microbiome showed effects on immunoregulatory pathways, influencing, for example, the risk of asthma in infants (Kelly et al., 2022; Lowry et al., 2016; Riiser, 2015). These findings reinforce the need for further research in the field of metagenomics to broaden our understanding of the interaction between microbes and microbial communities.

In contrast to studying the genome content of a single culturable organism, metagenomics enables the study of genomes of all microorganisms present in a specific environment at the same time (Hugenholtz & Tyson, 2008; Wooley et al., 2010). This approach comprises DNA extraction and sequencing without prior cultivation of clonal cultures in the laboratory, which allows for an unbiased characterization of the microbial community in that sample (Li, 2015; Yang et al., 2011). Recent studies have proven metagenomics valuable for many applications in clinical microbiology, like pathogen detection (Gu et al., 2019), outbreak investigation (Buytaers et al., 2021; Loman et al., 2013), molecular surveillance (Ko et al., 2022), and ecology studies (Coutinho et al., 2018; J. Gilbert et al., 2011), among others. With the emergence of nanopore-based DNA sequencing and the development of the small hand-held MinION sequencing device by Oxford Nanopore Technologies (ONT), it is now also possible to perform metagenomics se-

quencing and data analysis directly at the place where the sample was taken (Gardy & Loman, 2017; Johnson et al., 2017). Although this method offers faster library preparation than Illumina sequencing (Greninger et al., 2015), the data analysis and computational requirements in the field are still challenging and need further bioinformatics expertise (Boykin Okalebo et al., 2019). In particular, if the metagenomic sample is taken from a human host, the amount of sequenced host DNA can account for up to 99.9%, making microbial identification difficult, if not impossible (Andrusch et al., 2018; Greninger et al., 2015). In this thesis, I will focus on developing and applying computational methods that facilitate real-time analysis of metagenomics sequencing data and help overcome issues with overrepresented sequences in the underlying sample. I will also stress the importance of fast and space-efficient data structures needed for the microbial identification and metagenomics analysis of samples directly in the field when high-performance computing hardware is not accessible.

1.2 Microbial identification using high-throughput sequencing

1.2.1 From First to Third Generation DNA Sequencing: a brief overview

All genetic information of a cellular organism is encoded in a polymer that consists of only four types of nucleotides: *adenine*, *cytosine*, *guanine* and *thymine*. This molecule, known as deoxyribonucleic acid (DNA), is composed of two bonded strains of chained nucleotides that coil around each other and form a double-helix structure (Watson & Crick, 1953). This double helix is connected at each position by forming hydrogen bonds of two complementary nucleotides, i.e., A with T and C with G, called base pairs (bp). In contrast to living organisms, many viruses encode their genomic information using ribonucleic acid (RNA) (Holmes, 2009), which is a different type of nucleic acid that contains the nucleic base *uracil* instead of *thymine*. While DNA in nature only forms a double-stranded structure, RNA can also occur in a single-stranded form (W. Gilbert, 1986). In cellular organisms, the different types of RNA play an essential role in processes like protein synthesis (Mattick, 2011).

The genome consists of genes, which are the blueprint for proteins, and non-coding regions that are not translated into proteins but can have regulatory functionality (Clamp et al., 2007; Rogozin et al., 2002). This information is encoded just by the order of the four different nucleotides. The genetic information is inheritable, implying that closely related organisms have very similar genomes. This assumption is used in many applica-

tions that are based on genetic analysis, for example, genealogy analysis (Rosenberg & Nordborg, 2002), taxonomic classification of microorganisms (Hugenholtz et al., 2016), and reconstruction of pathogen transmission networks (de Bernardi Schneider et al., 2017).

Nowadays, deciphering the order of nucleotides of a DNA molecule is fundamental to many applications in genomics. This method, which is known as DNA sequencing, was first described by two independent groups in the 1970s. The chemical sequencing approach of Maxam and Gilbert was prevalent in the early days of DNA sequencing but lost its relevance because of the extensive use of hazardous chemicals and difficulties in automating the process (Maxam & Gilbert, 1977). In contrast, the dideoxy sequencing method proposed by Sanger et al. has evolved by exchanging radioactive labeling of DNA fragments with fluorescent dyes (Sanger et al., 1978). The success in automation of Sanger sequencing also made the first sequencing of the human genome possible, which was finally published in 2001 (Lander et al., 2001; Venter et al., 2001). Today, the method is known as First Generation Sequencing and is mainly used in low-throughput targeted re-sequencing projects. Compared to recent sequencing technologies, Sanger sequencing is costly for large genome projects (Patel et al., 2016), but it serves as a gold standard for confirming the results of the new technologies (Totomoch-Serra et al., 2017).

After the first draft of the human genome was published, companies started to develop more sophisticated sequencing instruments. The beginning of the new wave of sequencing technologies marked the introduction of a paralleled version of pyrosequencing, which reduced sequencing costs dramatically compared to automated Sanger sequencing (Egholm et al., 2005). The Roche 454 pyrosequencing device was the first commercially successful second-generation sequencing instrument, which produced higher amounts of sequencing data in a shorter time frame with Sanger-like read lengths of up to 1,000 bp (Goodwin et al., 2016). The second new technology that came to the market was Solexa's sequencing-by-synthesis method, which was based on reversible dye terminators and engineered polymerases (Bentley et al., 2008). Later acquired by Illumina, this sequencing platform can produce hundreds of millions of highly accurate reads in less than two days. Illumina currently offers the most cost-efficient and scalable sequencing machines, thus dominating the sequencing market with about 80% market share (Cimino, 2022). Although other second-generation technologies, like the ABI SOLiD system (Valouev et al., 2008) or Ion Torrent Ion semiconductor system (Rothberg et al., 2011), have been developed over the years, none could compete with Illumina's technology. With the expiration of some critical patents, new platforms like MGI's nanoball sequencing (Drmanac et al., 2010) and Element Bioscience's sequencing-by-binding technology

(Arslan et al., 2023) are expected to have a bigger impact on the sequencing market within the next few years. The low error rates have helped Illumina sequencing become a widely adopted platform for metagenomics sequencing and microbial identification (Diao et al., 2022; Patro et al., 2016). However, the short read lengths of up to 300 bp are the major disadvantage of second-generation sequencing and have pushed the development of long-read third-generation sequencing technologies (Pearman et al., 2020).

Sequencing a complex metagenomics sample or a repetitive genome, like that of a human, can be challenging using second-generation sequencing technologies. In many cases, it is unclear from which part of a genome or from which organism a short read of just 300 bp length originates, which can lead to biased data analysis and false conclusions of an experiment (Breitwieser et al., 2019; Portik et al., 2022). Inspired by this problem, a new generation of sequencing technologies has been developed, which can provide read lengths up to millions of base pairs. The first commercial product was released by Pacific Biosciences (PacBio) in 2011, using a technique called single-molecule real-time sequencing (Eid et al., 2009; Levene et al., 2003). Like Illumina sequencing, this technology follows a sequencing-by-synthesis approach but can produce read lengths of 10-25 kilobase pairs (kbp). While initial error rates of the technology ranged between 11-15%, the recently developed multiple pass circular consensus sequencing of long individual molecules produces long sequencing reads with 99.9% accuracy (Wenger et al., 2019). Using this technology, even the highly repetitive telomeres and centromeres of the human chromosomes could be sequenced, resulting in the first gapless human reference genome (Nurk et al., 2022; Rhie et al., 2023).

In 2015, a new sequencing technology came to the market, completely different from the sequencing-by-synthesis approaches of second-generation sequencing and Pacific Biosciences (PacBio). With its nanopore sequencing devices, ONT offers single-molecule long-read sequencing based on moving a single-stranded DNA molecule through a tiny membrane protein called nanopore (Clarke et al., 2009; Kasianowicz et al., 1996; Olasagasti et al., 2010; Stoddart et al., 2009). Besides producing read lengths of up to 2.3 million bp (Payne et al., 2019), ONT also provides the possibility to sequence a sample directly at its origin by offering the small MinION sequencer, which is not much larger than a USB stick (Mikheyev & Tin, 2014). This has already been demonstrated for molecular surveillance during the Ebola outbreak in 2015 (Quick et al., 2016), for metagenomics analyses of clinical and environmental samples (Gowers et al., 2019; Greninger et al., 2015; Urban et al., 2021), and even for sequencing runs onboard the International Space Station (Castro-Wallace et al., 2017). Although these features are clear advantages, the lower throughput compared to other sequencing technologies and

high error rates of 5-15% impeded wider adoption of the technology until recently (Delahaye & Nicolas, 2021; Rang et al., 2018). However, in this work, I will mainly focus on nanopore sequencing because the long read lengths combined with the portability of the platform and the supported real-time analysis of single molecules during the sequencing runs make it an ideal tool for microbial identification and metagenomics profiling.

1.2.2 Nanopore DNA Sequencing

Starting in 1996, a radically different sequencing approach was initiated with the discovery that single strands of nucleic acids can be electrophoretically driven through a nanoscale channel in a lipid bilayer (Kasianowicz et al., 1996). In the first experiments, this lipid bilayer separated two chambers (*cis* and *trans*) filled with a potassium chloride solution, and a voltage of 120-180 millivolts (mV) was applied across the membrane by electrodes placed in each chamber's solution (Akeson et al., 1999). Because of the presence of the nanopore in the lipid bilayer, the positively charged potassium ions are drawn to the negatively charged electrode, and the negatively charged chloride ions are drawn to the positively charged electrode, which results in a measurable ion current, also referred to as open channel current. Since a nucleic acid is negatively charged, its addition to the *cis* chamber will electrophorese it through the nanopore to the positively charged electrode. The traversing nucleic acid reduces the number of potassium and chloride ions that can simultaneously traverse through the nanopore and thus reduces the measured current by 80-90% (Muthukumar, 2016). Since the diameter of a nanopore is barely greater than the diameter of a single-stranded DNA, the hydrogen bonds of the double-stranded DNA molecule placed in the *cis*-chamber will break apart when it comes into contact with the aperture of the nanopore. Thus, only a single-stranded DNA moves through the nanopore, and the small variations of ionic current reflect the sequence of nucleotides traversing through the pore.
With the described setup, only two major obstacles remained to be overcome. First, the rate of nucleotide traversal through the nanopore needed to be controlled to measure and correctly distinguish the small variations in ionic current. Since the DNA moves through the nanopore at a rate of 1,000,000 bases per second, a braking device was needed to slow the translocation (Branton & Deamer, 2019). Introducing polymerases and helicases as motor proteins bound to the single-stranded DNA solved this issue (Byrd et al., 2012). After getting in touch with the aperture of the nanopore, the motor protein steps along the DNA in a direction away from the nanopore, which leads the DNA to move through the pore in a rachet-like motion and slows down the rate to approximately 420-450 bases per second (Cherf et al., 2012; Stoddart et al., 2009). The

second obstacle is that several nucleobases, and not only one, produce the current level changes observed as the DNA moves through the pore. The number of nucleotides that simultaneously affect the measured current depends on the length of the narrowest part of the nanopore, also known as the sensing region (Meller et al., 2001). Currently used nanopores have sensing regions that reflect approximately 3-4 nucleotides (Shi et al., 2016). With this knowledge, base-calling algorithms could be developed that use the ionic current of known sequences and assign them to nucleotide motifs of length three or four. Recent advancements in base-calling algorithms using deep-learning techniques showed tremendous improvements in base-calling accuracy, reaching up to 99% per read accuracy (Boza et al., 2021; Delahaye & Nicolas, 2021).

When the first MinION sequencers became available to the research community in 2014, the portability of the small device and the long reads it produces got the most attention. However, the technology supports another exciting feature, which makes it attractive for many applications in microbiology. The MinION flowcell has 512 sequencing channels, which results in sequencing a maximum of 512 DNA molecules simultaneously (Branton & Deamer, 2019). The device also has the ability to unblock any of the channels that are clogged with tangled DNA or some contaminants. By reversing the potential, the DNA or contaminants can be pulled out of the pore on a per-channel basis, which resets the channel to an "open pore" state and allows sequencing of the subsequent DNA molecule. This ability also offers to program the MinION to respond to partially sequenced DNA molecules, choosing to either completely sequence the captured DNA or reject it by pulling the DNA molecule back into the *cis* chamber. This feature, unique to nanopore sequencing, enables the sequencing of only those DNA molecules that are of a predetermined interest and also saves sequencing time for these higher-priority fragments. In practice, the first few hundred bases of a DNA molecule are sequenced and analyzed in real time while the DNA strand passes the nanopore. If the fragment is found to be of interest, the sequencing continues. Otherwise, the strand can be rejected from the pore, freeing that pore to sequence the next DNA molecule.

The described "Read Until" feature, also known as *adaptive sampling*, is provided by ONT via an API and was first mentioned in the literature by Loose et al. (2016). Their idea was to use a dynamic time-warping algorithm to align the first few hundred bases of each sequenced fragment against the reference sequence of the lambda phage. They enriched two small 5-kbp long genome regions by rejecting all reads that did not align to these regions. Other approaches have been developed during the last years that work on the raw electrical signals (Bao et al., 2021; Kovaka et al., 2021) or use a base-calling and mapping approach (Payne et al., 2021). In this thesis, I will describe an advanced method for nanopore *adaptive sampling* that relies on real-time base-calling but uses

data structures for approximate membership queries instead of read-mapping algorithms. We further applied this method to the *in-silico* enrichment of bacterial plasmids, showing its potential for clinical metagenomics applications.

1.2.3 Taxonomic classification and profiling

In microbial sequencing, the main goal is to study the sequenced genetic material of an environmental or clinical sample. The first step in this workflow is usually the taxonomic classification of all identified organisms in a sequenced sample. This can either be done by sequencing single or multiple loci of genomic regions that are conserved across the microbial kingdom under investigation or by sequencing the entire genetic content of the studied sample (Marchesi & Ravel, 2015). Typical examples of the first approach are community profiling based on the highly conserved 16S ribosomal RNA gene for bacteria (Huse et al., 2008; Lane et al., 1985) and fungi's internal transcribed spacer (ITS) regions (O'Brien et al., 2005). Although these methods can provide fast and cost-effective identification of bacteria and eukaryotes, they only cover the content of specific genomic regions, suffer from amplification bias (Campanaro et al., 2018; Fouhy et al., 2016), and can not be applied to viruses. In contrast, whole shotgun metagenomics sequencing avoids the amplification bias by directly sequencing the complete genome content of the underlying sample. It also achieves a higher resolution and coverage of the studied community and is not limited to certain kingdoms of life (Durazzi et al., 2021; Roux et al., 2019).

The advent of second-generation sequencing technologies has made whole shotgun metagenomics sequencing a cheap alternative to amplicon-based methods. In particular, Illumina's short-read technology is widely used, which generates millions to billions of reads in less than 48 hours of sequencing (Goodwin et al., 2016). Those small fragments of sequences obtained by high-throughput sequencing machines are the primary source of input for many computational methods that try to assign them to a specific taxonomic rank, like species, genus, or family (McIntyre et al., 2017). For this taxonomic classification, state-of-the-art computational methods rely on reference sequence databases of previously cataloged organisms (Sczyrba et al., 2017). Because of the much lower error rates, most taxonomic classification tools were explicitly designed for Illumina's short reads, and only a few have been developed for long-read technologies (Portik et al., 2022).

Some computational tools extend the taxonomic classification by providing the relative abundances of a list of organisms or taxonomic groups for the studied microbial sample. During the last few years, three different approaches to taxonomic profiling have become

popular. First, marker gene-based methods like MetaPhlAn4 (Blanco-Míguez et al., 2023) profile communities based on databases of clade-specific or single-copy genes. They enable fast profiling but potentially miss important low-abundant species when their genome sequences are not completely covered (Breitwieser et al., 2019). Secondly, whole genome methods like Kraken2 (Wood et al., 2019) map the reads to the whole genome references of an underlying database. These methods suffer from reads with matches to multiple reference genomes, which can decrease the specificity of the method on lower taxonomic ranks (Simon et al., 2019). Especially the short read lengths result in a high risk that reads are classified among several similar sequences. Here, using long-read technologies can improve the results of taxonomic profiling despite their higher error rates (Portik et al., 2022). The third set of methods consists of DNA-to-protein tools, like Kaiju (Menzel et al., 2016), that classify sequencing reads by mapping them to a database of protein sequences. These tools have higher computational requirements than the other two approaches because they need to analyze all six frames of potential DNA to amino acid translation. However, they can be more sensitive to novel and highly variable sequences due to the lower mutation rates of amino acid sequences compared to nucleotide sequences (Altschul et al., 1990). The most relevant drawback of these methods is their limitation to target only coding sequences of the genomes, which results in large amounts of unclassified non-coding reads (Simon et al., 2019).

Taxonomic profiling and relative abundance estimations should always be interpreted with caution. Many challenges have been reported in the literature when benchmarking different taxonomic profilers. First, abundance estimations are only based on the underlying reference databases, which are far from complete and can yield overestimates for the abundances of known and more studied species (Nasko et al., 2018). Secondly, the organisms in a community have different genome sizes, resulting in more sequenced reads (and bases) from organisms with larger genome sizes. This can result in two different abundance estimation values: sequence abundance, which counts the number of sequenced bases for each organism, and taxonomic abundance, which normalizes the number of sequenced bases of each organism by its genome length (Z. Sun et al., 2021). Third, some biases can arise from organisms with different susceptibilities to DNA fragmentation or the microbiome within sequencing kits, which is one of the main reasons negative controls should always be the norm in metagenomics sequencing studies (Nearing et al., 2021; Paniagua Voirol et al., 2021).

Regardless of the sequencing technology used, all whole-genome-based methods s from the ever-increasing size of reference genome databases. These large databases lead to high computational requirements when building an indexed database or querying the index. In particular, if the reference databases need to contain as many known

references as possible to optimize the sensitivity of the taxonomic classification, the memory requirements of the tools can only be satisfied by high-performance computing (HPC) clusters (Meyer et al., 2022). This makes real-time taxonomic profiling in the field impossible when researchers are usually only equipped with a small laptop. To overcome this issue, new space-efficient data structures are needed that provide fast querying of millions of reads for real-time pathogen detection and profiling of microbial communities for portable labs.

1.3 Pseudo-Mapping and Data Structures for Approximate Membership Queries

1.3.1 K-mers, minimizers and syncmers

With the emergence of DNA sequencing technologies, a new research field was established, focusing on the computational analysis of nucleic acid sequencing data. First, algorithmic approaches adopted methods from computer linguistics, like the edit distance (Levenshtein et al., 1966), to perform pairwise comparisons of DNA sequences (Needleman & Wunsch, 1970; Sellers, 1974). Further advancements of this sequence alignment approach resulted in the first biological sequence database search tool, `BLAST`, which is probably the most used bioinformatics tool in history (Altschul et al., 1990). However, the appearance of second-generation sequencing technologies led to an exponential growth of sequence databases and massive sequencing datasets, which made the application of sequence alignment algorithms computationally infeasible. Heuristic approaches like the seed-and-extend methods that use fixed-length seeds (Kent, 2002), maximal exact matches (MEMs) (Liu & Schmidt, 2012), or maximal unique matches (MUMs) (Delcher et al., 1999; Marçais et al., 2018) between pairs of sequences were developed to cope with the millions to billions of short reads produced by the new sequencing devices. The simplest fixed-length seed is the exact k-mer match, while MEMs are exact matches that cannot be extended in either direction without allowing a mismatch. Maximal unique matches are inherently MEMs but require uniqueness in addition. Many alignment or mapping tools have been developed based on the seed-and-extend approach, including BWA (Li & Durbin, 2010), Bowtie2 (Langmead & Salzberg, 2012), and minimap2 (Li, 2018). These seed-based heuristic methods have been used in many pipelines for microbial identification (Hong et al., 2014; Piro et al., 2016), but also became popular in other applications for biological sequence comparison. In particular, for the challenging task of short-read *de novo* genome assembly, a method for constructing whole genomes from a large number of short DNA fragments, De Bruijn graphs became the method of

choice (Compeau et al., 2011). This approach finds all unique substrings of size k, known as k-mers, in the set of short reads and connects k-mers that share $k-1$ nucleotides. After building the graph, the genome is reconstructed by finding an Eulerian path, or in other words, by finding a path through the graph that traverses each edge exactly once.
During the last 15 years, many k-mer-based approaches have been developed with applications to DNA sequence comparison. In metagenomics, for example, it is often sufficient to know whether a sequenced read belongs to a particular reference genome in a given database, regardless of the correct position in that genome. This so-called pseudo-mapping was initially used in RNA-seq quantification (Bray et al., 2016) because it is computationally cheaper than regular alignment. Later, it was realized that the same method also applies to metagenomic read assignments (Reppell & Novembre, 2018; Schaeffer et al., 2017). Pseudo-mapping-based approaches often use k-mers to represent a reference genome or sequencing read. By determining the cardinalities of these k-mer sets (e.g., the size of the intersection of the k-mer sets of two sequences), metrics like the Jaccard index (Jaccard, 1912; Tanimoto, 1958) or containment score (A. Z. Broder, 1997; Koslicki & Zabeti, 2019) can be applied to determine the similarity between two DNA sequences. However, the number of k-mers in a set is often very large, which motivated the development of methods for selecting subsets of k-mers in order to optimize the time and space requirements. The canonical example of a method designed to select a common subset of k-mers from similar sequences is *minimizers* (Roberts et al., 2004). Here, for two strings that share long enough exact substrings, the *minimizer* selects the same k-mers in the identical substrings, making it suitable to quickly estimate the similarity of two strings. The *minimizer* scheme is defined by three parameters: the k-mer length k, the window size w, and a total order of all k-mers ϕ. For each window of w consecutive k-mers, the smallest k-mer with respect to the ordering ϕ is selected as a *minimizer*, where the set of *minimizers* for a sequence is constructed by taking the union of *minimizers* over all windows. `Kraken` (Wood & Salzberg, 2014) was the first software tool utilizing *minimizers* for metagenomic classification and profiling, which was almost 1,000 times faster in classifying short reads than the fastest alignment-based approach. In the following years, different *minimizer*-based tools have been developed for read mapping (Li, 2018), taxonomic classification (Wood et al., 2019), and other sequence analysis tasks (Sommer et al., 2007; Ye et al., 2012), all trying to reduce the computational requirements by applying advanced underlying data structures.
Although k-mer-based approaches are superior to alignment-based approaches regarding the computational requirements, they tend to have more issues with sequencing errors (Marchet et al., 2021). Detecting sequence similarity using k-mers requires that all letters of a k-length substring are exactly conserved between the sequences. If one of

the letters is mutated (through biological mutation or sequencing errors), the k-mer is removed from the set of matching k-mers. This is also true for *minimizers* because they are a selected subset of k-mers. Beyond that, a *minimizer* can also be removed from the set of matching *minimizers* if one of the positions inside the window but outside the selected k-mer is mutated. In this case, some other k-mer in the window can be the smallest one with respect to the ordering ϕ. For this reason, *minimizers* are classified as a context-dependent k-mer selection scheme (Shaw & Yu, 2022).
When working with short-read data sets, which have relatively small sequencing error rates, context-dependent k-mer selection schemes offer sufficient conservation. Here, conservation is defined as the fraction of bases in a sequence that can be 'recovered' by k-mer matching after the sequence undergoes a random mutation process. For applications using long-read sequencing methods, the conservation of context-dependent k-mer selection schemes is decreased due to higher error rates. For this reason, many context-free k-mer selection schemes have been proposed in recent years, with *syncmers* shown to improve conservation and compression factor compared to *minimizers* (Dutta et al., 2022; Edgar, 2021). In the syncmer scheme, k-mers are selected based on the position of the smallest-valued substring of length $s < k$ within the k-mer. A closed syncmer is only selected if its smallest s-mer is at the start or end of the k-mer, while an open syncmer is selected if the smallest s-mer is the first s-mer within the k-mer. For open syncmers, an offset parameter t can be defined such that the syncmer is only selected if the smallest s-mer starts at position t of the k-mer. Edgar (2021) has shown that syncmers are more robust against mutations and sequencing errors than minimizers, which predestines them for applications utilizing k-mer selection schemes.

1.3.2 Filter-based data structures

The question of determining the membership of a key in a database commonly shows up in many bioinformatics applications. Regarding pseudo-mapping, we are frequently concerned with answering questions of the form "Is k-mer x present in the set of k-mers S built from genome G"? Querying a database for the presence of a key (or k-mer) may be expensive in terms of the database size or query time. Therefore, we sometimes use probabilistic filters that quickly indicate whether a given key is present in a set while using less memory than the set itself (Singh et al., 2020). However, the small size and high speed come at the cost of reporting false positives. That means, with a small probability, a key is reported as being present in the set while, in fact, it is not. On the other hand, if a key is present in the set, probabilistic filters never fail to correctly report that the key belongs to the set. There are two variants of probabilistic filters: the static

variant, where the size of the set S is specified ahead of time, and the dynamic variant, where the elements of the set are specified one by one in an online fashion.

The Bloom filter (Bloom, 1970) is the classical probabilistic filter-based data structure for approximate membership queries. As a dynamic filter structure, it allows for progressive construction without requiring that all keys are known at construction time. A Bloom filter consists of a bit array and a collection of m hash functions $h_1, h_2, ..., h_m$ that map keys (or k-mers) to indexes in the bit array. During the construction of the Bloom filter, keys are added by computing m hash values that point to m positions in the bit array and setting the bits of the corresponding positions to 1. When testing for the membership of a key, we calculate m hash values using the same hash functions as for the construction and check whether all bits at the corresponding positions are set to 1. The standard Bloom filter does not allow removing keys but supports adding keys irrespective of the bit array size or the number of hash functions, which increases the false positive probability as more entries are added, and thus, more bits are set to 1. It is also possible to control the false positive probability if the maximum number of keys to be stored is known. This can be done by choosing an appropriate size of the bit array and finding an optimal number of hash functions (A. Broder & Mitzenmacher, 2004). Many variations of the standard Bloom filter have been proposed, including blocked Bloom filters (Putze et al., 2010), counting Bloom filters (L. Fan et al., 2000), compressed Bloom filters (Mitzenmacher, 2001), and many others (Almeida et al., 2007; Calderoni et al., 2015).

For nucleic acid sequence comparisons, different versions of Bloom-filter-based data structures were developed for counting the number of k-mers of a given sequence in a given database. One such example is the sequence Bloom tree (SBT) (Solomon & Kingsford, 2016), which was designed for collections with high k-mer redundancy, such as human RNA-seq sequencing data. However, for metagenomics applications with large collections of heterogenous k-mer sets, this approach is not suitable, which led to the development of Bloom filter matrix-based methods, including the BItsliced Genomic Signature Index (BIGSI) (Bradley et al., 2019) and the Compact Bit-Sliced Signature Index (COBS) (Bingmann et al., 2019). One special case of the latter approach is the Interleaved Bloom Filter (IBF), where the k-mer set of each reference sequence in a given database is stored in a single Bloom filter, and the single bits of the many Bloom filters are combined in an interleaved fashion (Dadi et al., 2018). This enables fast querying of all single Bloom filters simultaneously, making it a perfect data structure for real-time metagenomics applications (Piro et al., 2020).

There are also many alternatives to the Bloom filter approach, all relying on the concept of a *fingerprint*. These methods store for each key the result of a dedicated hash function as a *fingerprint*, which is typically a word of a fixed number of bits. For a given set of

keys, the data structure is built such that the retrieved key of the data structure maps the *fingerprint*. When querying a key, we consider it present in the initial set of keys if the stored *fingerprint* equals the *fingerprint* retrieved from the data structure. Unlike Bloom filters, which can have a lower false positive probability if fewer keys are stored than the maximum set capacity, *fingerprint* approaches have a fixed false positive probability, which depends on the *fingerprint* size. Several *fingerprint*-based probabilistic filters have been proposed recently, including Golomb-compressed sequences (Putze et al., 2010), Cuckoo filters (B. Fan et al., 2014), Quotient filters (Pandey et al., 2017), and Morton filters (Breslow & Jayasena, 2018), among others (Chazelle et al., 2004; Graf & Lemire, 2022).

The recently proposed *XOR filter* is a particular case of a *fingerprint*-based probabilistic filter data structure (Graf & Lemire, 2020). It consists of a collection of m hash functions and an array of L-bit words. When querying a key, the hash values give us the positions in the array, and we retrieve the fingerprint of the data structure for that key by computing a bitwise XOR of the corresponding L-bit words. If this retrieved L-bit value equals the stored fingerprint value, we consider the key present in the set of keys for which we have built the XOR filter. Building the XOR filter requires that all keys are known a priori because the order in which keys are added to the filter has to be determined before the construction. Although XOR filters use L-bit valued arrays instead of single-bit arrays, they are significantly smaller than Bloom filters. Graf and Lemire (2020) have shown that XOR filter arrays need much fewer entries than Bloom filter arrays, which results in about 15% space savings when both approaches have the same false positive probability. Furthermore, XOR filters show faster query times than Bloom filters. These features generally show that XOR filters are an excellent alternative data structure to Bloom filters for applications based on pseudo-mapping. Since the exponential growth of reference sequence databases poses a challenge to real-time metagenomics analysis, I will dedicate one chapter of this thesis to a new data structure that is based on the concept of XOR filters and Interleaved Bloom Filters (IBFs). I will further show how this data structure improves the computational requirements of taxonomic classification and profiling, enabling real-time long-read metagenomics analysis on commodity hardware in the field.

1.4 Thesis outline

The overall topic of this work is the development and application of pseudo-mapping and probabilistic filters in the context of real-time metagenomics analysis of nanopore sequencing data. The final goal is to apply existing AMQ data structures to nanopore

adaptive sampling for removing unwanted DNA sequences from metagenomics data and improve these data structures to enable real-time taxonomic classification with large reference databases. Thereby, the enhancements in performance and scalability are achieved by algorithmic developments and technical optimizations. In the following chapters, I describe three parts of the project, each corresponding to a specific task in real-time analysis of nanopore sequencing data. While chapters 2 and 3 focus on the application of Interleaved Bloom Filters (IBFs) to nanopore adaptive sampling, chapter 4 presents a more efficient probabilistic filter for pseudo-mapping as a basis for real-time metagenomics analysis.

Chapter 2 introduces `ReadBouncer`, a new tool for nanopore adaptive sampling. It demonstrates superior read classification performance for adaptive sampling compared to other state-of-the-art tools. This is achieved by combining k-mer matching statistics with Interleaved Bloom Filters for improved pseudo-mapping and reduced memory usage. `ReadBouncer` is available as a command-line tool that directly interacts with ONT sequencing devices and comes with easy-to-install binary files for Linux and Windows operating systems. It supports both graphics processing unit (GPU) and central processing unit (CPU) base-calling, enabling adaptive sampling even on commodity hardware. I conceptualized the project with Bernhard Renard, who also offered supervision and support at all stages of the project. I implemented `ReadBouncer` and the supplementary scripts for data analysis with some support from Ahmad Lutfi, who wrote automated tests and developed a Graphical User Interface (GUI) for `ReadBouncer`. Finally, I collected the data and performed all experiments presented in the paper, with assistance from Kilian Rutzen, who prepared samples and performed the sequencing in the laboratory, and I wrote the manuscript with feedback from all authors. The chapter is based on the following article:

> Ulrich, J.-U., Lutfi, A., Rutzen, K., & Renard, B. Y. (2022). ReadBouncer: precise and scalable adaptive sampling for nanopore sequencing. *Bioinformatics*, *38*(Supplement_1), i153–i160. https://doi.org/10.1093/bioinformatics/btac223
>
>

I also presented the results in a proceedings talk at the conference for Intelligent Systems in Molecular Biology (ISMB) 2022.

Chapter 3 presents an application of nanopore adaptive sampling in general and `ReadBouncer` in particular. Instead of removing unwanted DNA sequences of a

specific organism in a metagenomics sample, we remove overrepresented chromosomal sequencing reads from bacterial isolate samples. We show that utilizing adaptive sampling to remove those chromosomal sequences enriches the low-abundant plasmid DNA *in silico* without applying any laboratory enrichment method. This also results in significantly improved quality of *de novo* assemblies of plasmids and reduced sequencing time needed to achieve this quality. We discovered that enrichment could also be achieved with expired flow cells, which, combined with adaptive sampling, shows the potential for cost savings in clinical and laboratory sequencing of bacterial pathogens. The study was jointly conceptualized by Lennard Epping, Torsten Semmler, Bernhard Renard and me. The sequencing experiments were performed at the sequencing core facility of the Robert Koch Institute (RKI) with support from Lennard Epping and Tanja Pilz, who performed the wet lab work. The sequenced bacterial samples were provided by Birgit Walther and Kathrin Stingl. I wrote all supplementary Python and R scripts and performed data analysis and interpretation, supported by Lennard Epping. Finally, I wrote the manuscript with feedback from all authors. Bernhard Renard supervised and supported the project with helpful comments and advice at all times. The chapter is based on the following article:

> Ulrich, J.-U., Epping, L., Pilz, T., Walther, B., Stingl, K., Semmler, T., & Renard, B. Y. (2023). Nanopore adaptive sampling effectively enriches bacterial plasmids. *bioRxiv*. https://doi.org/10.1101/2022.10.03.510741
>
> Manuscript submitted for peer review

I presented the preliminary results of this work as a poster at the Oxford Nanopore Community Meeting 2023.

Finally, Chapter 4 introduces the hierarchical interleaved XOR filter (HIXF) as a new data structure for approximate membership queries. I implemented a pseudo-mapping approach based on this new data structure and open canonical syncmers in a tool for real-time taxonomic classification and profiling of long reads. For the taxonomic profiling step, I implemented a standard expectation maximization (EM) algorithm that utilizes taxonomic abundance estimations to refine the classification results. The presented software `Taxor` is evaluated on simulated and real mock community data sets of ONT and PacBio long reads. I also show results of a read classification benchmarking comparing `Taxor` to state-of-the-art metagenomic profiling tools. Finally, all tools are compared with regard to their computational requirements, showing the superiority of the HIXF approach in terms of memory and disk space usage. Besides designing the HIXF data structure and implementing the pseudo-mapping approach and the EM algorithm in `Taxor`, I also conceptualized the study and wrote the data analysis scripts

for the evaluation and benchmarking of `Taxor`'s results. Finally, I collected the data, performed all experiments presented in the paper and wrote the manuscript with feedback from Bernhard Renard, who also offered supervision and support at all stages of the project. The chapter is based on the following article:

> Ulrich, J.-U., & Renard, B. Y. (2023). Taxor: Fast and space-efficient taxonomic classification of long reads with hierarchical interleaved xor filters. *bioRxiv*, 2023–07. https://doi.org/10.1101/2023.07.20.549822
>
> Submission in preparation.

I presented the preliminary results in a conference talk with an accompanying poster at Intelligent Systems in Molecular Biology (ISMB) 2023.
Chapter 5 summarizes the thesis and provides an outlook for potential future developments in the field of real-time analysis of nanopore sequencing data.

2 Precise and scalable nanopore adaptive sampling with Interleaved Bloom Filters

Summary

Nanopore sequencers allow targeted sequencing of interesting nucleotide sequences by rejecting other sequences from individual pores. This feature facilitates the enrichment of low-abundant sequences by depleting overrepresented ones in-silico. Existing tools for adaptive sampling either apply signal alignment, which cannot handle human-sized reference sequences, or apply read mapping in sequence space relying on fast GPU base callers for real-time read rejection. Using nanopore long-read mapping tools is also not optimal when mapping shorter reads as usually analyzed in adaptive sampling applications. We present a new approach for nanopore adaptive sampling that combines fast CPU and GPU base calling with read classification based on Interleaved Bloom Filters (IBFs). ReadBouncer improves the potential enrichment of low abundance sequences by its high read classification sensitivity and specificity, outperforming existing tools in the field. It robustly removes even reads belonging to large reference sequences while running on commodity hardware without GPUs, making adaptive sampling accessible for in-field researchers. ReadBouncer also provides a user-friendly interface and installer files for end-users without a bioinformatics background.

This chapter is based on Ulrich et al. (2022), which is a joint work with Ahmad Lutfi, Kilian Rutzen and Bernhard Y. Renard. A detailed description of the authors' contributions can be found in section Thesis outline.

2.1 Background

During the last decade, the invention of nanopore sequencing instruments has democratized DNA sequencing in various aspects (Leggett & Clark, 2017; Mikheyev & Tin, 2014). For example, the small MinION devices of ONT provide the possibility to

sequence a sample at the place of its origin without the need to ship the sample to a laboratory (Runtuwene et al., 2019; Sim & Chapman, 2019). This point-of-care sequencing ability makes nanopore sequencing attractive for applications such as pathogen detection in a clinical setting and in the field (Mongan et al., 2020; Quick et al., 2016). It also can shorten the time to detect pathogens or ARGs when using it for point-of-care testing.
While the size of the device and the easier and faster sample preparation are clear advantages, nanopore sequencing still lacks the base quality of sequencing-by-synthesis instruments. However, recent improvements in base-calling algorithms showed per read accuracy exceeding 90% (Rang et al., 2018; Wick et al., 2019). ONT even claims to boost per-read accuracy up to 99% with their latest R10.4 pore version (https://nanoporetech.com/accuracy). Another exciting feature of ONT's instruments is sequencing DNA molecules in a targeted fashion. ONT provides an Application Programming Interface (API) that enables receiving electrical currents, measured while the molecule transverses the pore (Loose et al., 2016). These signals can be translated into sequence space and analyzed in real time. An uninteresting DNA molecule located in a pore can be ejected by sending an *unblock* message back to the control software. This message leads the sequencer to reverse the voltage across the pore, causing the molecule to exit the pore in the reverse direction. The primary requirement for such a live depletion system is that the software making ejection decisions can keep up with the sequencing speed for up to 512 nanopores that concurrently sequence DNA molecules on a MinION sequencer.
Two recent publications describe the implementation of such systems for specific settings. Payne et al. combined ONT's `Guppy` base caller (Wick et al., 2019) with the `minimap2` read aligner (Li, 2018) in their `Readfish` workflow to make ejection decisions after mapping the reads to a reference genome in real-time. Kovaka et al. skipped the base-calling step and performed ejection decisions directly on nanopore current signals. While the latter is designed to run on a general-purpose CPU, it cannot handle large human-size reference genomes. In contrast, `Readfish` can handle larger references but needs additional software like `DeepNano-blitz` (Boža et al., 2020) or ONT's `Guppy` GPU basecaller for real-time base-calling.
Furthermore, using `minimap2` (Li, 2018) for read classification is not optimal. In their study, Payne et al. showed that only 83% of target reads were correctly classified for rejection after 0.8 seconds of sequencing. Marquet et al. observed the same issue when they tried to deplete all human host reads from vaginal samples with ONT's adaptive sampling option. Using the depletion method supported by `MinKNOW`, 25% of human reads could not accurately be rejected by the software, wasting many resources on sequencing uninteresting reads. Further, missed mappings to repetitive regions of the

reference genome can lead to delayed classifications when longer parts of the DNA molecule must be sequenced to make a rejection decision. Both lower sensitivity and classification delay will cause decreased enrichment of clinically relevant sequences of undetected pathogens or antibiotic resistance markers.
This study introduces `ReadBouncer` as a new tool for nanopore adaptive sampling that combines state-of-the-art base-calling software with the DREAM index (Dadi et al., 2018; Piro et al., 2020). `ReadBouncer` facilitates both GPU base-calling with ONT's `Guppy` as well as CPU base-calling with `DeepNano-blitz` (Boža et al., 2020). Its Interleaved Bloom Filter data structure allows for fast querying of hashed k-mers on large sequence datasets resulting in an improved read classification strategy. Within an integrated workflow, `ReadBouncer` uses IBFs to classify base-called DNA fragments for ejection and finally communicates the decision to the sequencing control software.
We first investigate our read classification approach by comparing it to other software tools used for read classification in a nanopore adaptive sampling context. `ReadBouncer` shows the best accuracy, recall, F1-Score, and Matthews correlation coefficient (MCC) among all tools on a simulated and a real-world dataset, while having almost the same precision and specificity as the best competitor. Furthermore, our tool also has the smallest reference sequence index size and peak memory usage.
We also compare `ReadBouncer` with `Readfish` and ONT's `MinKNOW` software using a playback run of a whole human genome sequencing experiment to evaluate its adaptive sampling performance. In this comparison, we demonstrate that `ReadBouncer` outperforms the other tools in a targeted sequencing experiment. `ReadBouncer`'s results consistently show more sequenced base pairss for target references and significantly shorter mean read length of off-target or rejected nanopore reads. These results indicate that `ReadBouncer` can make faster and more reliable rejection decisions than `Readfish` and `MinKNOW`. `ReadBouncer`'s source code and installer files for Windows and Linux are freely available as a Git repository (https://bit.ly/3j8GKPx) under GNU General Public License 3 (GPL-3.0).

2.2 Methods

2.2.1 Read Classification

With the current nanopore sequencing speed of 450 bp per second, an adaptive sampling approach ideally makes ejection decisions within 2 seconds after sequencing of a DNA molecule has started. This requires fast base calling and rapid and reliable classification of read fragments smaller than 500 base pairs. `Readfish` (Payne et al., 2021) uses

the long-read alignment tool `minimap2` (Li, 2018) for this purpose. Although being fast and accurate for long error-prone nanopore reads, the alignment approach poses some challenges when working with short error-prone fragments of less than 500 base pairss. For optimal enrichment of low-abundance genomic regions, we need to make reliable rejection decisions as fast as possible. Payne et al. (2021) showed in their study that it takes about 360 bp for `minimap2` to align 90% of those reads correctly. That means if we want to get higher enrichment, we need to improve the classification sensitivity for the same read length. Mappings are also hard to use when there is no good quality reference sequence available for an organism that is the depletion target, such as non-model organisms. In such scenarios, one would try to use the reference sequence of a closely related species for read classification. Mapping reads to the reference of a closely related species would fail to find numerous reads one would aim to eject from the pore.

All these findings motivated us to seek a different, fast classification strategy. To our knowledge, the fastest current sequence comparison algorithms use k-mer-based approaches, where a DNA sequence is divided into small overlapping substrings of size k. One approach, known as `MinHash` (A. Z. Broder, 1997; Ondov et al., 2016), computes a hash value for every k-mer of a sequence and stores the smallest hash values within a data structure called a sketch. The same procedure is applied to the second sequence, and the number of hash values present in both sketches gives an accurate approximation of the identity between the two sequences. Although this works well for sequences of similar size, it fails for sequence containment tests, where one sequence is much smaller than the other one, which is the case when we want to check if a nanopore read is part of a reference genome.

A better approach for testing if the set of k-mers of a reference genome contains the k-mers of a read is using Bloom Filters (Bloom, 1970; Koslicki & Zabeti, 2019). A Bloom Filter simply is a bitvector of size n and a set of h independent hash functions. To insert a k-mer into a Bloom Filter, the bit positions that correspond to the h hash values of the k-mer are set to 1, and a k-mer is considered present in the Bloom Filter if all h positions return a 1 during the lookup phase. In our case, we would insert all k-mers of a reference genome into the Bloom Filter and lookup for the k-mers of a nanopore read in that Bloom Filter.

The biggest problem of k-mer-based approaches is choosing the correct parameter value for k, which is always a tradeoff between sensitivity and specificity in the presence of sequencing errors. Larger values for k will result in more specific read classification results but will also fail to find many reads from the reference genome when the number of sequencing errors is high. When trying to classify nanopore reads with error rates of

about 10%, the value for k will hardly become bigger than 13. The number of different k-mers of size 13 is combinatorially defined by $4^{13} = 67,108,864$, which is much too small when working with human-sized genomes that compose about 3 billion k-mers of size 13. To overcome this issue, we divide the reference genome into overlapping fragments of size m and construct a separate Bloom Filter for each fragment. However, querying one read against each of the Bloom Filters separately reduces the performance of the Bloom Filter approach. Thus, we decided to use Interleaved Bloom Filters as proposed by Dadi et al. (2018) to index the reference genomes.

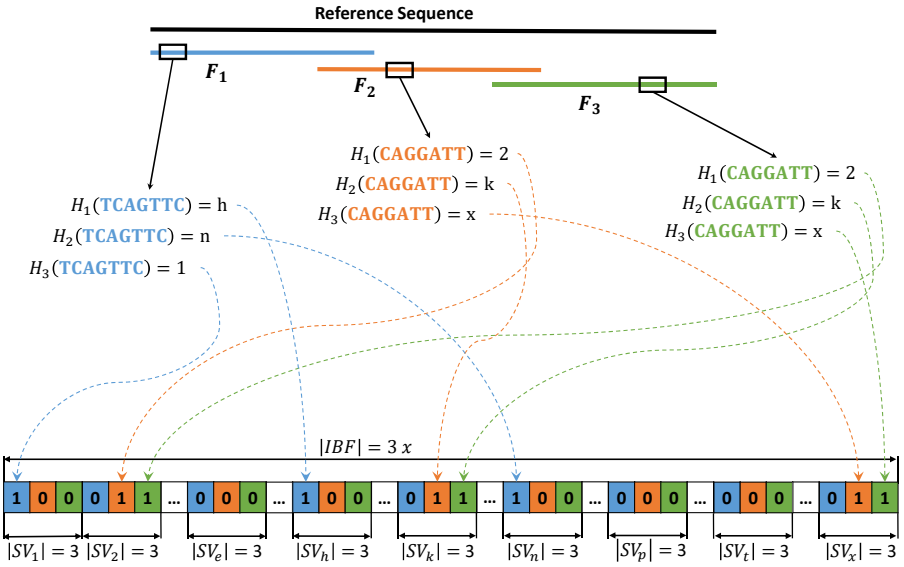

Figure 2.1: Example of an Interleaved Bloom Filter (IBF) construction. In the first step, we subdivide the reference sequence into three overlapping fragments. Then, for each k-mer of the differently colored fragments, all three hash values have to be calculated. The resulting hash values determine the subvector SV_j in which the corresponding bit is set to 1. For example, the second hash function for k-mer $CAGGATT$ from fragment F_3 returns k. Hence, we set the third bit of subvector SV_k to 1. In this way, the three Bloom Filters for the three fragments are combined in an interleaved fashion. Since we have three fragments in our example, the length of every subvector is three, and the length of the IBF is $3x$, where x is the defined length for every Bloom Filter of the three fragments.

An Interleaved Bloom Filter (IBF) combines several Bloom Filters (bins) in one single bitvector. The IBF can be divided into several subvectors, each having the size of the number of bins. Since one bin in the IBF corresponds to one fragment of the reference sequence, the size of each subvector corresponds to the number of fragments. In Figure 2.1, for example, we divided the reference sequence into three overlapping fragments, each corresponding to one bin of the IBF. Thus, each subvector in the IBF consists of 3 bits. The i-th bit of every subvector belongs to the Bloom Filter bin of fragment F_i. When inserting a k-mer from fragment F_i into the IBF, we compute all h hash values,

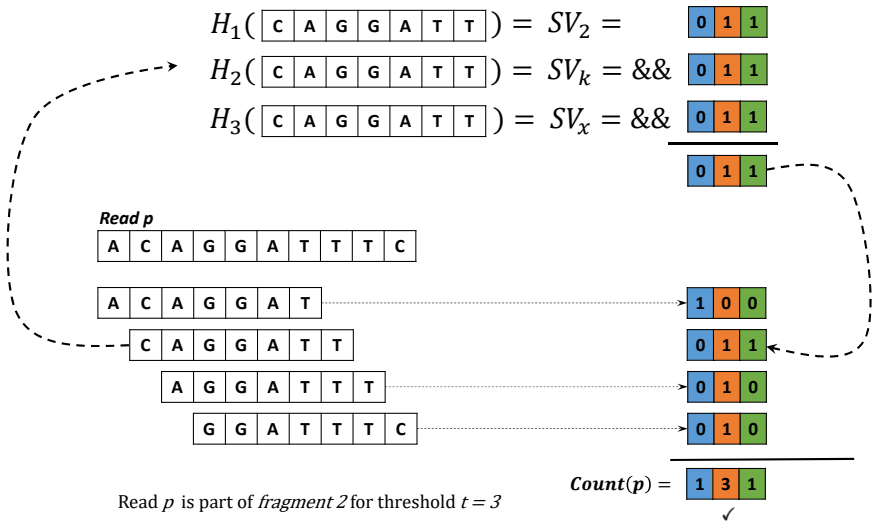

Figure 2.2: Finding the correct fragment for a given read p. For each k-mer of read p, we calculate the three hash values using the same hash functions as for the IBF construction. We use the resulting hash values to find the corresponding subvectors of the IBF. The sub bitvectors are combined with a bitwise AND to a binning bitvector. For all set bits in the binning vectors of the k-mers, we increment the counter of the corresponding bin in a counting vector. Bins whose counter is greater than or equal to a given threshold t are considered to contain the read p. In this example, we show the calculation of the binning bitvector for the 7-mer $CAGGATT$. Using the same three hash functions as for the IBF construction in Figure 2.1, we get the subvectors SV_2, SV_k, and SV_x. We combine these three subvectors via logical AND to get the binning bitvector. The same procedure is applied to the other three 7-mers, and with the resulting four binning bitvectors, we can calculate the number of matching 7-mers of read p with each fragment. If at least three 7-mers match against one fragment, we accept the read as a match with the reference sequence.

which point us to the corresponding subvectors SV_j and then simply set the i-th bit of this subvector to 1.
When querying a read p against the IBF in order to check if it maps to any of the fragments, every k-mer of that read is matched against the IBF. That means we first retrieve the h subvectors SV_j and apply a logical `AND` to them, resulting in the required binning bitvector indicating the membership of the k-mers in the bins. The example in Figure 2.2 visualizes this process. Here, the read consists of four 7-mers, for which we have to calculate the three hash values that point us to the corresponding subvectors SV_j, as can be seen in particular for the 7-mer $CAGGATT$. A logical `AND` of these three subvectors gives us the binning bitvector for that 7-mer. In our example, the binning vector 010 for $CAGGATT$ tells us that this 7-mer only matches fragment F_2. Applying this procedure to every 7-mer of the read gives us four binning bit vectors. Finally, we only need to sum up the 1-bits in the binning vectors for every fragment, which gives us the number of matching 7-mers of the read for every fragment. Thus, instead of computing h hash values for every Bloom Filter separately, we only need to compute the h hash values once, which poses a significant reduction in computing time to investigate the membership of a k-mer in every Bloom Filter. This method lets us quickly count the number of matching k-mers between the reference genome and a specific nanopore read. The challenge is to define a threshold value for the number of matching k-mers required to accept a certain nanopore read as a match against a fragment and, thus, as a match with the reference genome. In our example in Figure 2.2, we consider the read matching fragment F_2 because three of the four 7-mers match that fragment. Generally, the best threshold value depends on the length of the nanopore read and the expected sequencing error rate. We will describe our method for determining this value in the next section.

2.2.2 Optimal Bitvector Size

In the first step, `ReadBouncer` produces overlapping fragments of the given reference sequences, e.g., 100,000 bp long fragments with an overlap of 500 base pairs (bp). Each of those fragments represents a single bin in the Interleaved Bloom Filter (IBF). The constituting k-mers of each fragment are hashed using three different hash functions, and the bits of the corresponding index positions in the Interleaved Bloom Filter (IBF) are set to one (Figure 2.1). Then, `ReadBouncer` automatically calculates the optimal IBF size in bits ($Bits_{IBF}$) based on the following equations.

$$Bits_{IBF} = n_{frag} \times Bits_{SBF} \tag{2.1}$$

where n_{frag} is defined as the number of fragments with maximum size F and $Bits_{SBF}$ as a single Bloom filter size for a single fragment. Let max_{kmer} be the maximum number of k-mers for a fragment of size F, and k-mer size k be defined as

$$max_{kmer} = F - k + 1 \tag{2.2}$$

To calculate the optimal size for the IBF, we use the formula for finding the false positive rate in an IBF as proposed by Dadi et al.

$$p = \left(1 - \left(1 - \frac{1}{Bits_{SBF}}\right)^{h \times max_{kmer}}\right)^{h} \tag{2.3}$$

Then the optimal size of a single Bloom filter can be calculated by resolving the formula for $Bits_{SBF}$:

$$Bits_{SBF} = \left\lceil \frac{-1}{(1-r)^{\frac{1}{h \times max_{kmer}}} - 1} \right\rceil \tag{2.4}$$

where $r = p^{\frac{1}{h}}$, h is the number of used hash functions and p a predefined false positive rate. `ReadBouncer` implicitly uses three hash functions and a maximum false positive rate of 0.01 to minimize the number of false matches between the query sequence and a single bin of the Interleaved Bloom Filter (IBF).

2.2.3 Minimum number of k-mer matches

During the read classification step, the k-mers of every read are hashed with the same three hash functions, and the number of matching k-mers for every bin is calculated as visualized in Figure 2.2. We accept a read as part of the reference sequence if the number of matching k-mers is greater than or equal to a given threshold t for at least one bin. We calculate the threshold using the expected sequencing error rate e and the definition of a $(1-\alpha)$ confidence interval of the number of erroneous k-mers as recently provided by Blanca et al. (2022). They first defined the expected number of erroneous k-mers as follows:

$$E[N_{err}] = L \times q \tag{2.5}$$

For a given read r with length $len(r)$ and k-mer length k, we denote the number of k-mers of read r as $L = len(r) - k + 1$, and q is defined by $(1 - (1-e)^k)$. In the second step, they show that the variance for the number of erroneous k-mers can be

calculated by:

$$\begin{aligned} Var(N_{err}) = L(1-q)\Big(q\Big(2k+\frac{2}{e}-1\Big)-2k\Big) \\ + k(k-1)(1-q)^2 \\ + \frac{2(1-q)}{e^2}((1+(k-1)(1-q))e-q) \end{aligned} \tag{2.6}$$

Finally, they define the $(1-\alpha)$ confidence interval by:

$$E[N_{err}] \pm z_\alpha \sqrt{Var(N_{err})} \tag{2.7}$$

With $z_\alpha = \phi^{-1}(1-\frac{\alpha}{2})$, where we denote ϕ^{-1} as the inverse of the cumulative distribution function of the standard Gaussian distribution. Based on the calculation of the confidence interval for the number of erroneous k-mers, we define our threshold for the minimum number of matching k-mers for read r as:

$$min[N_{match}] = L - (E[N_{err}] + z_\alpha \sqrt{Var(N_{err})}) \tag{2.8}$$

We classify a read as a match if the number of matching k-mers is greater or equal to $min[N_{match}]$ for at least one bin in the IBF. `ReadBouncer`, per default, calculates this threshold for a 95%-confidence-interval, an expected sequencing error rate of 10%, and k-mer length 13. However, these values as well as the fragment size, are adjustable via configuration parameters of the command line or Graphical User Interface (GUI).

2.2.4 Workflow

The workflow of our tool consists of two consecutive parts. First, we build one or more indexes of the given reference sequence data set, which can be used as target or depletion filters. These indexes can be used directly in the second part of the workflow or stored on the computer's hard disk for later usage. The construction of this index, for which we apply Interleaved Bloom Filters (IBFs), is explained in further detail in section 2.2.1. The second part of our tool is the live-depletion or target-enrichment task (Figure 2.3). Here, `ReadBouncer` initially loads the indexes and waits for the nanopore device to start sequencing. Immediately after sequencing has begun, the sequencer streams raw electrical currents for every single molecule from every single sequencing pore of the flow cell to our integrated Read-Until client. ONT provides this functionality via an Application Programming Interface (API) of its `MinKNOW` control

software (https://github.com/nanoporetech/minknow_api), which allows our Read-Until client to receive the raw data while the molecule traverses through the pore. The client is implemented in C++ and communicates with the `MinKNOW` control software via Google Remote Procedure Call (gRPC) (https://github.com/grpc/grpc).
Received raw signal data get pushed onto a base-calling queue, and a separate thread takes raw signals of each read from the queue and sends it to the chosen base-calling algorithm, which translates the electrical currents into a nucleotide string. The user can choose GPU base-calling with ONT's `Guppy` basecaller for which we integrated a `Guppy` client that communicates with a `Guppy` basecall server. Additionally, we integrated `DeepNano-blitz` (Boža et al., 2020) for the base-calling step, which is fast enough to perform the base-calling in real-time, even on CPUs.
Base-called reads get pushed to the classification queue if the read length is greater than or equal to 200 base pairs (bp), and another thread takes each read from that queue and passes it to the classification framework. Otherwise, the thread marks this read as "pending" and waits for the following data chunk to be base called and concatenates the base-called sequences of the read until the minimum read length has been reached. The minimum read length of 200 bp ensures higher confidence in the classification of the reads. In practice, this read length requirement will lead to most reads having about 360 bp length, which corresponds to two data chunks sent by the `MinKNOW` software. The read classification thread then queries the read sequence against the loaded Interleaved Bloom Filter (IBF) indexes as described in more detail in section 2.2.1. Based on the classification, reads can either be marked for a rejection or continue further sequencing. If a read was not classified for rejection on a first try, we mark it as *once_seen* and wait for further sequencing data to try further classification attempts of that read. After the read has reached a maximum read length of 1,500 bp, we stop trying to make ejection decisions and mark the read for continued sequencing as usual. Reads that have been classified for rejection or continued sequencing are finally pushed to the response queue, and no further data chunks of that read are sent by the control software.
The last thread takes the classified reads from the response queue, and our Read-Until client sends response messages back to the `MinKNOW` control software for each read. The client sends an unblock message for reads that could be matched to the Interleaved Bloom Filter (IBF), telling the sequencer to eject the corresponding DNA molecule. A *stop_further_data* message is sent to the control software for reads that were not classified for rejection. This message tells `MinKNOW` to continue sequencing the corresponding DNA molecule and send no additional chunks of data for that read.

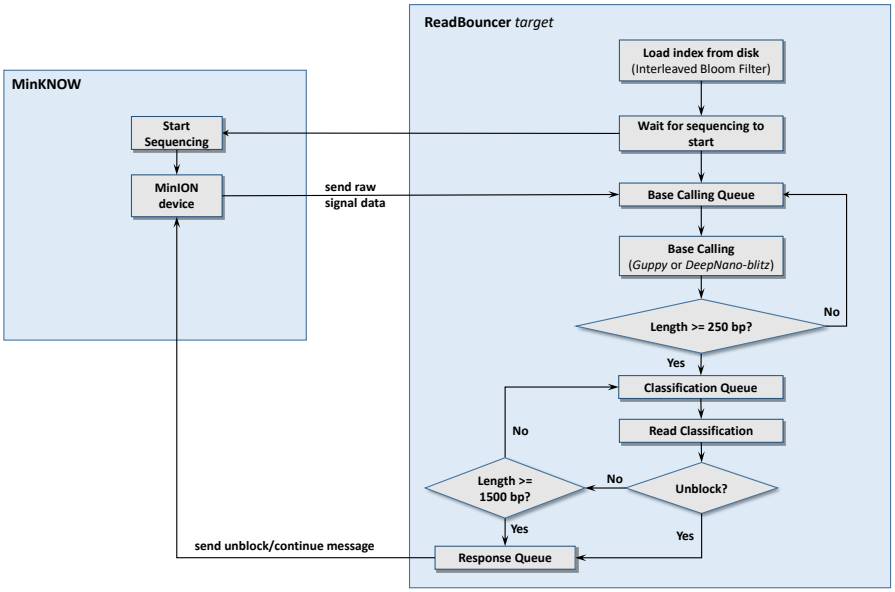

Figure 2.3: Flow diagram of `ReadBouncer`'s adaptive sampling workflow. An Interleaved Bloom Filter (IBF) of reference sequences is loaded from an index file first. `ReadBouncer` then waits until the `MinKNOW` starts the sequencing run. When sequencing has begun, the MinION device sends raw signals for every DNA molecule currently traversing a nanopore to `ReadBouncer`, pushing the signals onto a base-calling queue. The base-calling thread takes signals from the queue and performs base calling via `Guppy` or `DeepNano-blitz`. The base-called sequences get pushed onto the classification queue if the length is equal to or longer than 250 bp or pushed back to the base-calling queue otherwise. The classification thread takes sequences from the classification queue and queries the sequences against the depletion and/or target IBFs. If a sequence is found in the depletion IBF but not in the target IBF, the corresponding read is marked for unblocking and pushed onto the response queue. If the sequence is not found in the depletion IBF or is found in the target IBF and the sequence length is shorter than 1,500 bp, the corresponding read is pushed back to the classification queue. `ReadBouncer` repeats the classification procedure using consecutive chunks of data until the sequence length exceeds 1,500 bp. Reads that were not classified to reject are marked for sequencing as usual. A *stop_further_data* message for those reads is pushed onto the response queue. Finally, the response thread sends back action messages of reads from the response queue to the `MinKNOW` software and MinION device, respectively.

2.3 Results

In this study, we show how adaptive sampling benefits from our improved read classification approach. Therefore, we designed experiments that specifically focus on evaluating this approach when applied to both adaptive sampling strategies, depletion and targeted sequencing. In the first step, we compare `ReadBouncer` to `minimap2` (Li, 2018),

which is used for classification by `Readfish`, and the pan-genomics matching tool `SPUMONI` (Ahmed et al., 2021), which is proposed as an alternative to `minimap2` in targeted nanopore sequencing pipelines. Here, we assess all three tools on simulated and real reads from a recently published microbial mock community (Nicholls et al., 2019). In a second experiment, we compare `ReadBouncer` with `Readfish` in an adaptive sampling setting using the playback feature offered by ONT's `MinKNOW` software to replay an already completed sequencing run. We assess both tools by targeting chromosomes 21 and 22 in a human whole genome sequencing run, looking at their ability to correctly filter out all other human nanopore reads. Here, we do not compare against `SPUMONI` because there exists no adaptive sampling pipeline integrating `SPUMONI` for read classification.
We perform all experiments for classification performance assessment on a laptop with a 2.8 GigaHertz (GHz) Intel Core i7-7700HQ CPU and 16 GigaByte (GB) of memory with an Ubuntu 20.04 OS installed. For the classification evaluation, we run each tool with a single thread for runtime comparisons and record the wall clock time and peak resident set size (RSS) reported by the individual tools or GNU `time` 1.7.

2.3.1 Evaluating Read Classification

Experimental Setup

During a nanopore adaptive sampling experiment with ONT's ReadUntil functionality, the sequencing device transmits electrical current data via the `MinKNOW` control software to `ReadBouncer`. This data is received as chunks, representing a maximum of 0.4 or 0.8 seconds of sequencing, depending on the `MinKNOW` configuration. Since a DNA molecule translocates through the pore at a speed of about 450 bp per second, 0.4 seconds of sequencing represent about 180 bp of data. In the following experiments, we mimic the situation where a chunk represents 0.4 seconds of sequencing data received and base called immediately by an adaptive sampling tool. Since we aim to make rejection decisions as early as possible while still being able to classify most of the reads correctly, we want to assess the classification accuracy of the three tools after two chunks of data, which correspond to 360 bp or 0.8 seconds of sequencing. In this section, we assume that base-calling has already been performed. For a fair comparison, we set up all experiments in such a way that all three tools, `minimap2`, `SPUMONI`, and `ReadBouncer`, attempt to classify reads based on the 360 bp long read prefix. In practice, all reads, both simulated and real reads, were cut to only the first 360 base pairs (bp). `ReadBouncer` then hashes all k-mers of these 360 bp and compares the hash values to a prebuilt Interleaved Bloom Filter (IBF) of the depletion target references

to make classification decisions.
We use the software's default settings for the SPUMONI approach, which means splitting the prefix into substrings of 90 bp each for further read classification. SPUMONI also needs a prebuild index of the references but has to include the reverse complement of the depletion target references. SPUMONI matches the substrings against this positive index and a null index, consisting of the reverse sequences of the positive index. Finally, classification decisions are made by using a Kolmogorov-Smirnov test.
For the minimap2-based approach, we evaluate two different parameter settings. First, we mimic the read classification of Readfish by using the mappy Python interface (https://pypi.org/project/mappy/) for minimap2. Here, we align the read prefixes with the *map-ont* settings, which are the same settings used by Readfish and correspond to a k-mer size of 15. Since the choice of the k-mer size impacts the classification performance, we also aligned the read prefixes using a k-mer size of 13 in a second experiment to ensure a fair comparison with ReadBouncer.
To evaluate the three tools, reads correctly classified as belonging to the depletion target are considered true positives (TP), while reads falsely classified for depletion are called false positives (FP). Consistently, reads that are correctly not classified as depletion targets are considered true negatives (TN), and reads belonging to the depletion target but not classified for depletion are called false negatives (FN). We calculate the classification accuracy, precision, recall, specificity, and F1-score for all three approaches based on those considerations. The recall (or sensitivity) is the relative number of correctly classified reads from the depletion target defined by $\frac{TP}{TP+FN}$. Specificity is defined as the relative number of reads correctly classified as not belonging to the depletion target $\frac{TN}{TN+FP}$. Further, the accuracy is the relative number of all correct classifications, defined as $\frac{TN+TP}{TN+TP+FN+FP}$. The F1-score is the balanced harmonic mean of precision $\frac{TP}{TP+FP}$ and recall, calculated by $2 \times \frac{recall*precision}{recall+precision}$. Since we assume an imbalanced number of sequenced reads between depletion and enrichment targets, we also report the Matthews correlation coefficient (MCC) in every experiment.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{2.9}$$

Simulated Mock Community

In the first dataset, we consider simulated ONT-like reads derived from the identical genomes of the ZymoBIOMICS High Molecular Weight DNA Mock Microbial community (ZymoMC) (Nicholls et al., 2019). This mock community consists of seven bacterial species - *Enterococcus faecalis*, *Listeria monocytogenes*, *Bacillus subtilis*, *Salmonella*

enterica, *Escherichia coli*, *Staphylococcus aureus*, and *Pseudomonas aeruginosa* – as well as *Saccharomyces cerevisiae*. We use `PBSIM2` (Ono et al., 2021) to simulate ONT-like reads (R9.4 pores) from Zymo Mock Community references at varying levels of mean read accuracy: 80%, 85%, 90%, 95%, and 98%. Furthermore, we simulated proportions of reads from each genome in such a way to mimic a scenario where only 2.16% of reads originate from *Saccharomyces cerevisiae* (Figure A.1). The goal here is to enrich *Saccharomyces cerevisiae* sequences by correctly classifying bacterial reads, which we would aim to eject from the pores in a real nanopore sequencing run. This can be considered a depletion-only experiment, where a priori only the depletion references are known, but not the enrichment targets. Therefore, we build an index of the seven bacterial reference genomes and query all bacterial and yeast reads against the index. Consistent with our definition in section 2.3.1, we consider correctly classified bacterial reads true positives (TP), while yeast reads found in the index are considered false positives (FP). In addition, we define bacterial reads that are missed to be found by a tool in the index as false negatives (FN), and yeast reads that are not found in the index are considered true negatives (TN).
On all read accuracy levels, `ReadBouncer` consistently demonstrates the best accuracy, recall, precision, F1-scores, and MCC (Table A.1). Figure 2.4 visualizes recall and specificity for the three tools across various read accuracies. It can be observed that recall improves with increasing read accuracy for all three tools while specificity stays almost unchanged. On all read accuracy levels, `ReadBouncer` demonstrates slightly but consistently better recall (sensitivity) than `SPUMONI`, while both tools outperform `minimap2`. `Minimap2` is the only tool that shows 100% specificity, but `ReadBouncer` comes close to 100% as well. `SPUMONI` lags a bit behind the specificity scores of the other two tools. It can be seen that `ReadBouncer` is the best-performing tool for this read classification task. It combines high recall (sensitivity) with high specificity. The other two tools either have high recall but lower specificity or high specificity but lower recall scores.

Real Mock Community

Next, we applied our method to real nanopore reads from a Zymo Mock Community (NCBI BioProject PRJNA742838). After sample preparation, we sequenced the mock sample on a MinION flowcell (FLO-MIN106) with v.R9.4.1 pores (section A.1). Obtained Fast5 files were base called with `DeepNano-blitz` using a recurrent neural network size of 48. For better comparison with `minimap2`, we first build a separately obtained `minimap2` mapping as a gold standard. Therefore, we filter out all reads

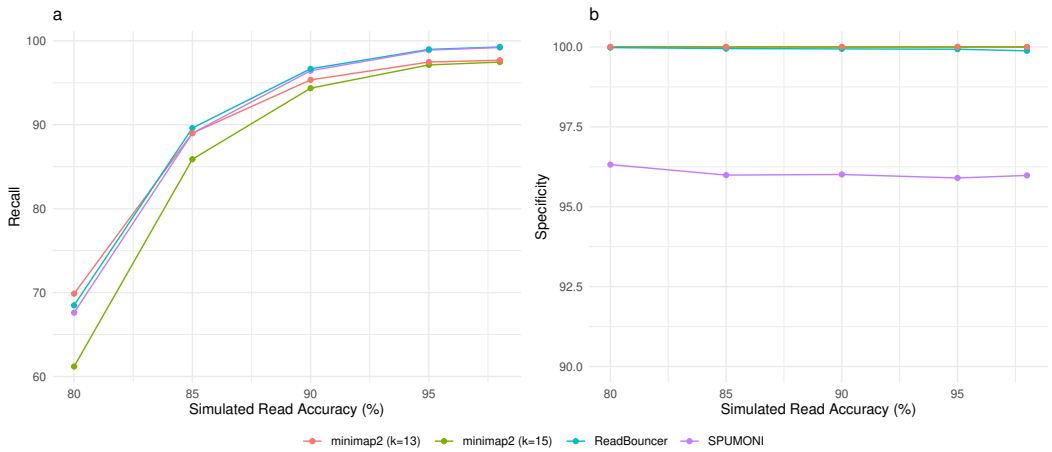

Figure 2.4: Visualization of *(a)* Recall and *(b)* Specificity with varying simulated read accuracies for `ReadBouncer`, `minimap2`, and `SPUMONI`.

shorter than 2,000 base pairs (bp) and trim the first 360 bp from each read since we use these bases for later classification. Then, we mapped the trimmed reads with standard ONT settings to the ZymoMC reference genomes and only reads with a mapping quality score greater or equal to 30 are considered confidently mapped. From these mapped reads, the trimmed 360 bp long prefixes are used for the read classification by the three tools again. Proportions of reads from each genome are similar to the simulated experiment with 2.27% of reads from *Saccharomyces cerevisiae* (Figure A.2). In this experiment, we also measure the peak resident set size (RSS) and index size in GigaByte (GB) and the throughput for each tool in reads classified per second.
Results in Table 2.1 show that `ReadBouncer` achieves better accuracy, recall, and F1-score than `SPUMONI` and `minimap2`, which both have similar results for those three measures. `Minimap2` has slightly better precision and specificity than `ReadBouncer`. While `SPUMONI` has almost the same precision as `ReadBouncer` and `minimap2`, it shows significantly less specificity. These results are consistent with those for the simulated data sets in section *Simulated Mock Community* and show that `ReadBouncer` outperforms the other tools on read classification for short nanopore reads.

Another important aspect is the amount of main memory a tool needs to hold the reference index needed for read classification. Using the seven bacterial reference genomes of the Zymo Mock Community as depletion target (reference index), `ReadBouncer` shows the smallest maximum memory consumption measured as peak resident set size (RSS). It only needs 0.099 GB of main memory, in contrast to 0.251 GB consumed by `minimap2` with k-mer size 13. Furthermore, `ReadBouncer` has the smallest

Table 2.1: Comparing `ReadBouncer`, `SPUMONI`, and `minimap2` across various metrics on a real Zymo Mock Community data set consisting of seven bacterial species and *Saccharomyces cerevisiae*. Reads from a nanopore sequencing run are mapped to the eight organisms to generate ground truth. We use only the first 360 bp for classification from those confidently mapped reads to mimic unblock decision-making after 0.8 seconds of sequencing the individual read. All reads are mapped against the seven bacterial reference sequences to filter out only the bacterial reads. At the same time, we want to keep as much *Saccharomyces cerevisiae* reads, which corresponds to an enrichment of that organism in an enrichment/depletion experiment. Consistent with the simulated data, `ReadBouncer` can classify a higher percentage of bacterial reads with slightly less precision and specificity than `minimap2`. Our approach also is the computationally most effective one, with the lowest memory footprint and highest classification throughput.

Tool	**ReadBouncer (k=13)**	**SPUMONI**	**minimap2 (k=15)**	**minimap2 (k=13)**
Accuracy	**94.50**	90.96	89.33	92.33
Precision	99.99	99.89	**100.00**	99.99
Recall	**94.38**	90.85	89.08	92.15
Specificity	99.73	95.87	**99.95**	**99.95**
F1-Score	**97.10**	95.16	94.23	95.91
MCC	**0.52**	0.41	0.39	0.45
Peak RSS (GB)	**0.099**	0.163	0.272	0.251
Index Size (GB)	**0.047**	0.153	0.097	0.090
Throughput (reads per sec)	**5967**	1102	5632	5306

index file size (0.047 GB) of all three tools. In addition to the smallest memory footprint, `ReadBouncer` also achieves the highest classification throughput. We can classify 5967 reads per second with our approach compared to 5632 reads per second by `minimap2` (k-mer size 15) and 1102 reads per second achieved by `SPUMONI`. These results show that `ReadBouncer` can correctly classify more reads and is computationally more efficient than other state-of-the-art tools used for nanopore adaptive sampling.

2.3.2 Adaptive Sampling Evaluation

In our live experiment, we assess our read classification based on Interleaved Bloom Filter in a targeted adaptive sampling setup. For this purpose, we downloaded a bulk Fast5 file of a human whole-genome sequencing experiment provided via the GitHub page of `Readfish` (https://github.com/LooseLab/readfish). Such a bulk Fast5 file (Payne et al., 2019) allows the playback of the whole sequencing run for testing if the ReadUntil functionality is working correctly. Oxford Nanopore Technologies' `MinKNOW` software

simulates an already finished sequencing run without requiring a physical sequencing device when performing a playback run. Compared to the original sequencing run, read signals are reported at the same time point after starting the run. Unblocking a read does not cause MinKNOW to finish sending signals for that read during a playback run. It just breaks the read when receiving an unblock message for the read and creates a new read identifier but continues to send signals of the same original read. Here, we compare ReadBouncer and Readfish using both real-time GPU base-calling with ONT's Guppy basecaller and real-time CPU base-calling with DeepNano-blitz. While ReadBouncer integrates DeepNano, we had to use a special git branch of Readfish (https://github.com/LooseLab/readfish/tree/caller_refactor) to facilitate CPU base-calling. In all experiments, ReadBouncer and Readfish were run on a separate Ubuntu 18.04 Laptop with 16GB random access memory (RAM) and Intel Core i7 while GPU live base-calling and the playback were performed on an NVIDIA Jetson AGX Xavier (512-core NVIDIA Volta GPU, 32GB LPDDR4X Memory). Additionally, we compared the results with two MinKNOW adaptive sampling experiments, one using MinKNOW's *target* and the other using MinKNOW's *deplete* method. Both experiments were performed on the NVIDIA Jetson AGX Xavier, too.

In our experiments, we do a playback of a complete human genome sequencing run with the goal of enriching chromosomes 21 and 22 of the human genome and depleting all other human reads from that run. This setup not only mimics a targeted sequencing approach but also corresponds to the application of sequencing a clinical human blood sample where up to 99% of the reads are human reads that we would want to deplete in order to enrich the number of reads from a pathogenic microbe. We perform playback runs for 60 minutes on ONT's MinKNOW control software (version 4.3.3). To ensure that the vast majority of the sequenced reads are of human origin, we first perform a playback run without adaptive sampling. Reads were base called with Guppy (version 5.0.14) and mapped with minimap2 to the human Telomere-to-Telomere Consortium ("T2T") CHM13 v1.1 reference assembly (Nurk et al., 2022). From the resulting reads passing the in-built quality filtering of MinKNOW, 99.66% could be mapped to the human reference genome. To compare the tools in an adaptive sampling setting, we first adjust the *break_reads_after_seconds* parameter within MinKNOW to 0.4 seconds as recommended by the Payne et al. (2021). Since MinKNOW sends data as chunks, this parameter sets the size of one chunk to a maximum of 180 base pairss. Both tools, ReadBouncer and Readfish, can concatenate the data chunks and perform classification after receiving every chunk. For integrated CPU base calling with DeepNano-blitz we used a neural network size of 48 for both tools. For real-time GPU base-calling on the NVIDIA Jetson AGX Xavier, we used the fast base-calling mode of Guppy (version 5.0.14) for

Table 2.2: Comparison of `ReadBouncer`, `Readfish` and `MinKNOW` in a targeted sequencing experiment. Four 60-minute playback runs of a whole human genome sequencing experiment were performed using either `ReadBouncer` or `Readfish` in combination with either `DeepNano` CPU base-calling or `Guppy` GPU base-calling. The same experiment was repeated with `MinKNOW`'s adaptive sampling functionality in *target* and *deplete* mode. The goal of all experiments was to target chromosomes 21 and 22 while rejecting all other human reads. For chromosomes 21 and 22, the highest mean and median read lengths across all experiments are highlighted. For rejected reads, the lowest mean and median read lengths across all experiments are highlighted. `ReadBouncer` and `Readfish` consistently show better results when using GPU base-calling, with `ReadBouncer` having shorter mean and median read lengths for non-target reads regardless of the used basecaller. `ReadBouncer` outperforms `MinKNOW` *target* by having longer read lengths for on-target reads and shorter read lengths for off-target reads caused by a better read classification. `MinKNOW` *deplete* has the worst results of all tools indicated by high numbers of on-target reads with short read lengths caused by lots of false unblock decisions for on-target reads.

Tool	Basecaller	contig	Reads	Bases	Mean	Median
ReadBouncer	DeepNano	chr21	73	2,208,211	**30,249**	9,025
		chr22	92	1,179,449	12,820	6,262
		others	122,745	136,441,510	1,112	503
	Guppy	chr21	77	2,189,976	28,441	**9,442**
		chr22	83	1,210,472	**14,584**	**7,663**
		others	154,684	140,636,076	907	**479**
Readfish	DeepNano	chr21	73	2,118,199	29,016	9,285
		chr22	91	1,177,699	12,942	5,449
		others	92,527	140,303,151	1,516	1310
	Guppy	chr21	71	2,126,553	29,951	9,262
		chr22	88	1,178,629	13,394	6,602
		others	140,267	133,484,295	952	877
MinKNOW *target*		chr21	77	2,099,268	27,263	9,170
		chr22	105	1,061,911	10,113	3,368
		others	38,656	140,284,944	3,629	520
MinKNOW *deplete*		chr21	1425	2,285,420	1,604	845
		chr22	468	1,219,518	2,606	883
		others	177,549	132,949,878	**749**	769

`ReadBouncer` and `Readfish`.

To evaluate both tools, we repeat the same playback run for 60 minutes. In the experiment with CPU base-calling, we ran `ReadBouncer` with default parameters ($fragment\_size = 100,000$ and $kmer\_size = 13$) using three base-calling threads and three read classification threads, respectively. The same setting was applied to `Readfish` with three CPU base-calling threads and `minimap2` using three threads

per default. Since Guppy ensures a higher raw read accuracy, we ran ReadBouncer with $fragment\_size = 200,000$, $kmer\_size = 15$, and $error\_rate = 0.05$ in the GPU base-calling experiment. In all experiments, we used chromosomes 21 and 22 as the target filter and all other chromosomes as the depletion filter in ReadBouncer. Our settings within the Readfish configuration file correspond to the example TOML file in the GitHub repository (https://github.com/LooseLab/readfish/blob/master/examples /human_chr_selection.toml) and aim to target chromosomes 21 and 22 as well while unblocking all reads that do not map to the targets. For MinKNOW *target*, we used chromosomes 21 and 22 as a reference, and for MinKNOW *deplete*, all other chromosomes as the reference sequence. Before starting the adaptive sampling experiments, we had to build index files for all three tools. For Readfish and MinKNOW, we created minimap2 index files, which took 103 seconds on an Intel Core i7 with one thread and peak RSS of 11.68 GB. Building ReadBouncer index files took 478 seconds on the same system but needed only 8.62 GB peak RSS. After finishing the playback run, the resulting Fast5 files were basecalled in high accuracy mode with Guppy (version 5.0.14). All reads in the resulting fastq files were mapped to the human genome reference and mapping statistics were calculated with Readfish's *summary* script. Using a playback run allows a fair comparison of the different approaches since the same sequencing data come from MinKNOW during the same amount of time. Thus, we expect a similar number of on-target reads and on-target bases across all experiments. On the other hand, we expect different numbers of rejected reads while retaining a similar number of bases for those reads due to the different lengths of rejected reads caused by different unblock time points. The reason is that MinKNOW just splits a sequenced read into two segments when receiving an unblock message for that read. Thus, the earlier we reject an off-target read, the shorter the read length, and the more off-target reads are seen.

The results of all six experiments can be seen in Table 2.2. Our first observation is that the results for our target chromosomes 21 and 22 are similar for all experiments but the MinKNOW *deplete* experiment. Here, the number of on-target reads is much higher while showing the smallest mean and median read lengths caused by a high number of false rejection decisions. These results suggest that MinKNOW *deplete* is not suitable for targeting single chromosomes of the human genome in an adaptive sampling experiment. On the other hand, MinKNOW *target* shows similar results for chromosomes 21 and 22 when compared to ReadBouncer and Readfish. However, the mean read length of 3,629 bp measured for unblocked reads is much higher than those in the ReadBouncer and Readfish experiments, which shows that MinKNOW *target* spends too much time sequencing off-target reads. These experiments show that ReadBouncer and Readfish outperform the two MinKNOW adaptive sampling strategies.

Comparing ReadBouncer with Readfish, when both tools use the same basecaller, ReadBouncer shows better results regarding median read lengths and the number of bases sequenced. We also see that the choice of the base-calling tool has a significant impact on the outcome of the adaptive sampling experiment. Using Guppy GPU base-calling for both tools, ReadBouncer and Readfish result in much shorter read lengths for non-target (unblocked) reads. Interestingly, we observe that unblocked reads from the ReadBouncer playback runs have shorter mean and median read lengths than those from the Readfish playback runs. This is also shown in the length distribution plots of unblocked reads for playback runs with Guppy base calling presented in Figure 2.5.

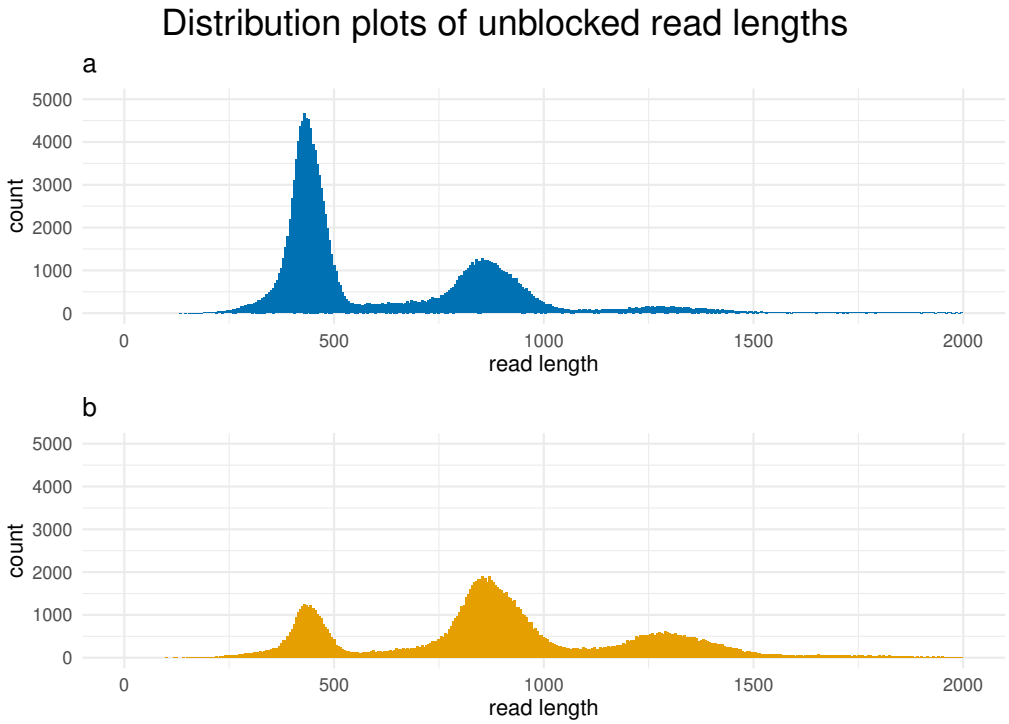

Figure 2.5: Read length distributions of unblocked reads when using *a)* `ReadBouncer` or *b)* `Readfish` on a 60 minutes playback run of a whole human sequencing experiment with real-time `Guppy` GPU base-calling. ReadBouncer makes faster rejection decisions than Readfish, which can be observed by shorter read lengths of unblocked nanopore reads.

2.4 Discussion

The idea of adaptive sampling is to selectively sequence individual DNA molecules on nanopore sequencing devices using in-silico methods. This study presents a new tool for adaptive sampling that improves read classification by combining Interleaved Bloom Filters with k-mer matching statistics. `ReadBouncer` shows a higher read classification sensitivity than other state-of-the-art classification tools for adaptive sampling while retaining a high specificity. Our tool also improves classification performance and memory usage compared to the other tools. We could observe shorter read lengths of non-target reads in different playback experiments when using `ReadBouncer` instead of `Readfish`. In a real experiment, this could mean that `ReadBouncer` investigates more DNA molecules in the same amount of sequencing time. We developed our tool as an easy-to-install software application with a Graphical User Interface on Linux and Windows operating systems. Additionally, `ReadBouncer` supports fast CPU base-calling, providing even small sequencing facilities or in-field researchers that typically only have access to low-cost hardware the possibility to use the adaptive sampling feature of the MinION sequencer.

The key benefit of our new tool is the improved read classification. We neither use signal nor sequence space mapping algorithms for read classification compared to other adaptive sampling tools. Instead, our Interleaved Bloom Filter (IBF) approach uses k-mer counting in Bloom Filters for sequence containment testing, resulting in smaller index files and fewer memory requirements. However, the improved sensitivity comes at the cost of decreased classification speed with increasing reference database size due to our approach of fragmenting the reference genome sequences and using one bin of the Interleaved Bloom Filter per fragment. The fragmentation approach ensures a high classification specificity for nanopore reads with high error rates of approximately 10-15% as observed by the CPU basecaller `DeepNano-blitz` (Boža et al., 2020). This error rate forces us to use small k-mer sizes such as 13, which requires smaller fragment sizes down to 100,000 bp to avoid too many false positive matches. Using real-time GPU base-calling with single raw read accuracies of about 94% allows increasing the k-mer size to 15 and fragment size to 200,000 bp, reducing the number of bins in the Interleaved Bloom Filter by 50%. In the future, we expect to use even fewer fragments per genome and consequentially improve the classification speed for larger genomes as Oxford Nanopore Technologies steadily improves its per-read accuracy. This could also enable using our IBF approach for real-time metagenomics classification of nanopore reads or the construction of pan-genomics indexes that store all different haplotypes of a pathogen in one IBF, with one haplotype per bin. To further increase performance,

combining ReadBouncer and minimap2 could be worthwhile, as the integration of different methods in related fields has demonstrated (Piro et al., 2017).
A second key feature of ReadBouncer is its support for fast and accurate real-time GPU base-calling with ONT's Guppy and real-time CPU base-calling with DeepNanoblitz. This study showed that both approaches show reliable results for a whole human sequencing playback run with the application to target specific chromosomes while rejecting reads belonging to all other chromosomes. Since there are some performance drawbacks of MinKNOW when using a playback run, the measured read lengths of rejected reads can deviate from a real experiment. Other users reported much shorter unblocked read lengths on real experiments performed on NVIDIA Jetson AGX Xavier (https://github.com/sirselim/jetson_nanopore_sequencing). To ensure reproducibility and fair comparison between tools and to reduce the influence of potential artifacts, we evaluated our tool here on a playback of a well-performed experimental run rather than during run-time of the sequencer. Since a playback run is data from a real sequencing experiment, we do not expect any bias from this comparison but can guarantee a fair comparison between tools. We also do not expect any negative impact on ReadBouncer's classification approach's improved sensitivity by using a playback run.
We expect that ReadBouncer can also contribute to the field of pathogen detection in non-model organisms. Metagenomics sequencing of such samples easily consists of up to 99% host reads that can be depleted with adaptive sampling resulting in an in-silico enrichment of pathogenic reads as shown by other research groups (Marquet et al., 2022; Martin et al., 2022). Here, our CPU-based approach also makes access to adaptive sampling much easier for researchers studying wild living animals in the field. With nanopores being successfully applied to peptide sequencing (Brinkerhoff et al., 2021), we also see possible modifications of the approach to be useful for targeted protein sequencing.
Another potential use case for adaptive sampling is the real-time detection of antibiotic resistance and virulence genes. In their recently published study, Zhou et al. (2021) showed that direct nanopore metagenomics sequencing of human blood samples could detect pathogens in real-time but failed to detect antimicrobial resistance gene (ARG). They compared the direct metagenomic sequencing approach to the MinION sequencing of blood culture samples. Using blood cultures, they could deplete human reads to about 65% of all sequenced reads in the corresponding sample, which was sufficient to identify more than 80% of resistance genes after 2 hours of sequencing. We expect that the number of sequenced human host reads can be depleted at a similar rate by using adaptive sampling, which was already shown by Marquet et al. (2022). This could reduce costs and decrease the time to detect pathogens in human blood samples. In the

future, a point-of-care test for ARGs in human patient samples that also avoids shipping the samples to a nearby laboratory could decrease antibiotic drug usage and help restrict the development of antibiotic resistance, which is a burden to many healthcare systems all over the world. Besides further sample preparation and sequencing technology improvements, we encourage scientists to set up proof-of-principle studies investigating the potential application of adaptive sampling for real-time ARG detection.

3 Nanopore adaptive sampling effectively enriches bacterial plasmids

Summary

Bacterial plasmids are key drivers in the spread of antimicrobial resistance. They are usually underrepresented in clinical samples and need to be enriched in the laboratory, which is expensive and prone to bias. Here, we introduce nanopore adaptive sampling as a bias-free in-silico method for enriching low-abundant plasmids in known bacterial isolates. We show that a significant enrichment can be achieved even on expired flow cells. Adaptive sampling improves the quality of *de novo* plasmid assemblies while reducing the sequencing time. Our experiments also highlight issues with adaptive sampling if target and non-target sequences span similar regions.

This chapter is based on Ulrich et al. (2023), which is a joint work with Lennard Epping, Tanja Pilz, Birgit Walther, Kerstin Stingl, Torsten Semmler and Bernhard Y. Renard. A detailed description of the authors' contributions can be found in section Thesis outline.

3.1 Background

Infectious diseases caused by bacterial pathogens have lost their threat to people living in high-income countries due to the discovery of antibiotic drugs within the last 70 years. However, adaptation processes within bacteria cause these drugs to lose their effectiveness in treating infectious diseases. The emergence of such antimicrobial resistance (AMR) already poses a significant threat to public health, with an estimated 4.95 million deaths associated with bacterial AMR in 2019 (Murray et al., 2022), and will even worsen, with around 10 million expected deaths per year by 2050 (O'Neil, 2014; O'Neill, 2016).

Besides vertically passing antimicrobial resistance genes (ARGs) to their offspring, bac-

teria can also transfer ARGs across the bacterial population by horizontal gene transfer. This process is mediated via mobile genetic element (MGE), such as plasmids, which are epichromosomal DNA elements unique to bacteria (Gonçalves et al., 2021; Partridge et al., 2018). Plasmids are a major driver in the spread of ARGs in bacterial populations (Carattoli, 2013) and have recently been found to accelerate bacterial evolution by enhancing the adaptation of the bacterial chromosome (Rodríguez-Beltrán et al., 2021).
Classifying plasmid types is crucial to understanding antibiotic resistance transmission between bacteria. Several recent studies have shown the benefit of whole genome sequencing for classifying plasmid types (Hidalgo et al., 2019; Orlek et al., 2017). In particular, the emergence of long-read sequencing by Oxford Nanopore Technologies (ONT) promises improvements for outbreak investigations due to its lower capital investment and shorter turnaround times (Stohr et al., 2020; Taylor et al., 2019). However, these methods suffer from the small proportion of plasmid DNA within the sequenced samples, primarily if bacteria can not be cultivated in the lab (Kav et al., 2013). Therefore, a large proportion of the plasmids in such samples are probably missed, or the sequencing depth is insufficient to assemble them correctly (Lynch & Neufeld, 2015). Thus, additional sample preparation steps are required to isolate or enrich plasmids before DNA sequencing, but they are too expensive and laborious for applications in clinical diagnostic settings.
While nanopore sequencing has been shown to reconstruct plasmids accurately (Wick et al., 2021), the technology offers a feature called adaptive sampling (AS) that has the potential to improve plasmid classification. First described in 2016 by Loose et al. (2016), nanopore adaptive sampling has been increasingly used for *in-silico* target enrichment within the last two years. Here, DNA molecules can be rejected from individual nanopores if the corresponding sequence is not interesting for downstream analysis. Pulling out unwanted DNA frees the nanopore for the following molecule to be sequenced and reduces the time spent sequencing uninteresting DNA fragments. Different tools implement adaptive sampling (Kovaka et al., 2021; Payne et al., 2021; Ulrich et al., 2022), using dynamic time warping (`UNCALLED`), read mapping (`Readfish`, `MinKNOW`) or k-mer-based (`ReadBouncer`) strategies, all performing rejection decisions by analyzing the first 160 to 450 base pairs (bp) of each read. Recently, deep-learning-based tools like `SquiggleNet` and `DeepSelectNet` have also been developed, addressing host depletion in human microbiome samples (Bao et al., 2021; Senanayake et al., 2023). The potential enrichment reached by using adaptive sampling was already shown, and even mathematical models that predict the enrichment factor were described in previous studies (Martin et al., 2022; Payne et al., 2021; Viehweger et al., 2023). In one study, Marquet et al. (2022) could enrich the microbiome in human vaginal samples by deplet-

ing host DNA. Further, Kipp et al. (2023) used adaptive sampling to enrich bacterial pathogens in tick samples, while Viehweger et al. (2023) even enriched single ARGs in human microbiome samples. However, no study has shown the potential enrichment of plasmid sequences in bacterial isolates by depleting their chromosomal DNA sequences. In the present study, we investigate the efficiency of adaptive sampling to enrich plasmid sequences in five different bacterial isolates. For this purpose, we used two adaptive sampling tools, which were shown to have high read classification performance, namely `MinKNOW` and `ReadBouncer` (Bao et al., 2021; Ulrich et al., 2022). Both tools use a combination of base-calling with ONT's `Guppy` and read classification on sequence level. While `MinKNOW`'s adaptive sampling feature is based on the Readfish (Payne et al., 2021) scripts and uses `minimap2` (Li, 2018) to map read prefixes against a given reference sequence set, `ReadBouncer` utilizes pseudo-mapping based on k-mers and interleaved Bloom Filters for making rejection decisions. We refrained from using `UNCALLED` because Bao et al. (2021) showed that the combination of base-calling and mapping has a higher read classification accuracy than `UNCALLED`. In order to increase sustainability and reduce sequencing costs, we also investigate whether enrichment of plasmids can be achieved with adaptive sampling on expired flow cells with reduced active pores. Finally, we evaluate the effective plasmid enrichment by comparing it to the predicted enrichment calculated by the mathematical model proposed by Martin et al. (2022) and demonstrate the usefulness of adaptive sampling for plasmid assemblies.

3.2 Methods

3.2.1 Culture and DNA extraction

Campylobacter strains were streaked on Columbia Blood agar (Oxoid, Thermofisher Scientific, USA) and incubated at 42 °C under a microaerobic atmosphere. *Enterobacter, Salmonella* and *Klebsiella* strains used in this study were streaked out on Luria Bertani (LB) plate and incubated over night at 37 °C. DNA extraction for *Campylobacter jejuni* (GCF_008386335.1) was done using the MagAttract HMW Genomic Extraction Kit (Qiagen). For *Salmonella enterica* (GCA_025839605.1), *Campylobacter coli* (GCF_025908295.1) (Tegtmeyer et al., 2022), *Klebsiella pneumoniae* (GCF_025837075.1) and *Enterobacter hormaechei* (GCF_001729785.1) DNA was extracted using the QIAamp DNA Mini Kit (Qiagen, Hilden, Germany) according to the manufacturer's instruction. The total amount of DNA was quantified using a Qubit fluorometer (Thermo Fisher Scientific) and frozen at -80ºC until further analysis.

3.2.2 Library preparation and sequencing

Sample preparation was performed according to the manufacturer's instructions without any optional pre-enrichment steps or size selection using the Rapid Barcoding Kit SQK-RBK004. Different barcodes were used for each of the bacterial isolate samples to correctly assign sequenced reads in the data analysis. Since we used expired flow cells with less expected overall sequencing yield, we decided to only sequence two or three bacterial isolates on one flow cell. Finally, the barcoded samples were sequenced on an Oxford Nanopore MinION (Oxford, UK) using FLO-MIN106D(R9.4.1) flow cells. All sequencing experiments were started via ONT's `MinKNOW` control software (version 4.5.0).

3.2.3 *In-silico* enrichment via adaptive sampling

We performed four sequencing runs using `MinKNOW` software v4.5.0 on an Nvidia Jetson AGX Xavier (512-core NVIDIA Volta GPU, 32GB LPDDR4X Memory) for 24 hours. In all experiments, we compared adaptive sampling with standard sequencing by dividing the flow cells into two parts: Adaptive sampling was performed on the first 256 channels, and standard sequencing was performed on channels 257 to 512. We used a new flow cell with 1,153 active pores for the first run (ReadBouncer1) and sequenced two *Campylobacter* isolates using barcodes RBK01 and RBK02. For the second run (ReadBouncer2), we used an expired flow cell with only 636 active pores for sequencing the three barcoded bacterial isolates (*Enterobacter, Salmonella, Klebsiella*) using barcodes RBK03, RBK04 and RBK05. The third run (MinKNOW1) used the same *Campylobacter* samples as the first, but we performed sequencing on an expired flow cell with only 557 active pores. For the fourth run (MinKNOW2), we used the identical three bacterial isolates as for the second run and performed sequencing on an expired flow cell with only 718 active pores after the initial flow cell check.
On the first two flow cells, we performed adaptive sampling with `ReadBouncer` (Ulrich et al., 2022) using the chromosomal references of the bacterial isolates as depletion targets. Here, a k-mer size of 15, a chunk length of 250 bp, a fragment size of 200,000 bp, and an expected error rate of 5% were used as parameters for the read classification. `ReadBouncer`'s k-mer size parameter was chosen accordingly to the default k-mer size used for mapping with `minimap2` (Li, 2018), which is used by `MinKNOW`'s adaptive sampling feature. The expected error rate reflects the current average per-read accuracy by ONT's `Guppy` basecaller. The other two parameters are default parameters. For flow cells three and four, `MinKNOW`'s adaptive sampling feature was used, which is based on the `Readfish` (Payne et al., 2021) scripts and uses `minimap2` (Li, 2018) to map

read prefixes against a given reference sequence set for read classification. We built a `minimap2` index file (parameter `-x map-ont`) for these experiments, including the chromosomal reference sequences, which we used as depletion targets for adaptive sampling. Read prefixes classified as "chromosomal" were rejected from the pore, and decisions were written to log files by both tools, `ReadBouncer` and `MinKNOW`. In all experiments, `ReadBouncer` and `MinKNOW` used `Guppy` GPU basecaller (fast model, v6.0.6. Oxford Nanopore Technologies (ONT)) for real-time base-calling of the raw signal data received from the device after at most 0.4 seconds of sequencing.

3.2.4 Data Analysis

All data analysis scripts were written in Python and R, and are freely available in the GitHub repository https://github.com/JensUweUlrich/PlasmidEnrichmentScripts. All plots were created in R using `ggplot2`.
After the sequencing runs were finished, we basecalled and demultiplexed all raw data with `Guppy` GPU basecaller (super accuracy model, v6.0.6. Oxford Nanopore Technologies (ONT)). `Guppy` trimmed barcodes and adapter sequences from the resulting nanopore reads during that process. Afterward, we computed read length metrics (see Table 3.1 and Figure 3.1) and created contour plots (see Figures 3.4 & 3.3) using the sequencing_summary files provided with the `MinKNOW` and `Guppy` output directories. Next, we mapped all demultiplexed and base-called reads against the reference genomes (including plasmid sequences) of the five bacterial strains using `minimap2 v2.19` (Li, 2018) with parameter `-x map-ont`. Based on the mapping results, we could assign each mapped read to either the bacterial chromosome or plasmid(s) of one of the bacterial isolates to create Figure 3.2. We also used the mapping results to calculate the percentage of sequenced plasmid and chromosome base pairs after 24 hours for each bacterial sample, resulting in Figure 3.6. We further used the sequencing summary file to separate the reads by their species of origin and partitioned them to comprise the cumulative data from the beginning of each experiment up to 24 hours, separated by 30 minutes of sequencing, which resulted in 48 individual timepoint data sets. With this information, we calculated for each experiment time point t the plasmid enrichment by yield for each bacterial strain,

$$Enrichment_{yield}(t) = \frac{yield_{AS}(plasmid, t)}{yield_{CTRL}(plasmid, t)} \tag{3.1}$$

where $yield_{AS}(plasmid, t)$ is the number of sequenced plasmid bases of a strain from the adaptive sampling region at time point t and $yield_{CTRL}(plasmid, t)$ is the number of

sequenced plasmid bases of a strain from the control region (without adaptive sampling) at time point t.
Similarly, we calculated the enrichment by the number of reads,

$$Enrichment_{reads}(t) = \frac{reads_{AS}(plasmid, t)}{reads_{CTRL}(plasmid, t)} \tag{3.2}$$

and the enrichment by the mean depth of coverage of the plasmid reference sequences.

$$Enrichment_{depth}(t) = \frac{depth_{AS}(plasmid, t)}{depth_{CTRL}(plasmid, t)} \tag{3.3}$$

According to the definitions above, $reads_{AS}(plasmid, t)$ and $reads_{CTRL}(plasmid, t)$ represent the number of reads from the adaptive sampling (AS) or control region (CTRL) that map to a plasmid of a given bacterial strain at experiment time point t. Further, $depth_{AS}(plasmid, t)$ denotes the mean sequencing depth of plasmids from a strain using mapping data from the adaptive sampling region at time point t and $depth_{CTRL}(plasmid, t)$ is the mean sequencing depth of plasmids on the control region at time point t. Here, we used `samtools coverage` (Li et al., 2009) to calculate the mean depth of coverage of every species' plasmid reference at each time point for the control and adaptive sampling regions. The different enrichment factor values calculated for each bacterial sample at any of the 48 time points were plotted and shown in Figure 3.9.
For the plots of active channels over time (Figure 3.5), a channel was defined as active from the beginning of the experiment up until the time it sequenced its final molecule (as long as it sequenced at least one molecule). The enrichment by composition shown in Figure 3.8 (a) and (b) was calculated by dividing the relative plasmid abundance from adaptive sampling regions by the relative plasmid abundance from control regions, both shown in Figure 3.6. We compared observed enrichment by composition and yield against predicted enrichment values using the mathematical model from Martin et al. (2022). To calculate predicted enrichment values, we used the recommended sequencing speed of 420 bp per second, capture time of 0.5 seconds, decision time of 1 second, and mean read lengths for each bacterial sample as provided in Table 3.1, and plasmid abundances of control regions for each sample as shown in Figure 3.6.

Since we expect plasmid sequences in our use case scenario to be usually unknown, we also did a *de novo* assembly of the demultiplexed fastq files, containing all reads sequenced after one and two hours of sequencing. This helps us to estimate the time required to obtain high-quality plasmid assemblies. Therefore, we assembled all demultiplexed nanopore reads from control and adaptive sampling regions separately using `Flye/metaFlye` assembler (v2.9.2, parameter "–meta") (Kolmogorov et al.,

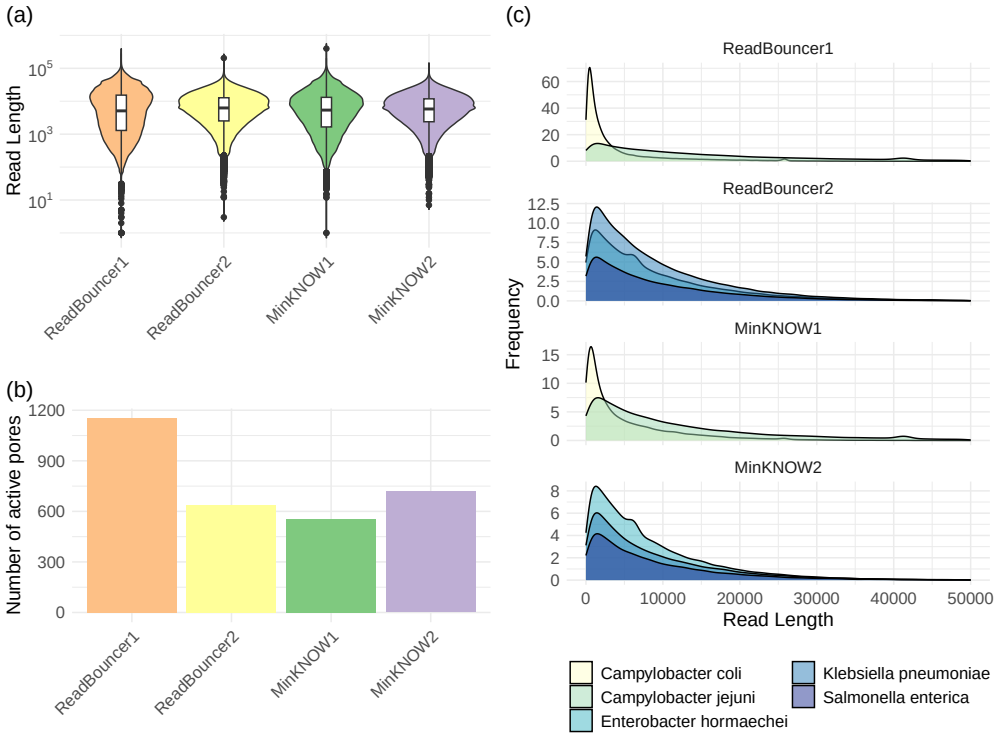

Figure 3.1: Evaluation of control regions from the first four sequencing runs. **(a)** Violin plots (log scale) of read length distributions. Box plots for read lengths are included within the violin plots. **(b)** Active pore count measured at the start of each sequencing run. ReadBouncer1 has the highest number of active pores because it was the only flow cell that was not expired. **(c)** Read length distributions for each species from control regions of the four sequencing runs.

2020; Kolmogorov et al., 2019). Then, we polished the obtained `metaFlye` assemblies with one round of `Medaka consensus` (v.1.8.0, default parameters, model r941_min_sup_g507, Oxford Nanopore Technologies (ONT)) using the same nanopore read set. We assessed the quality of the final assemblies with `Quast` (v5.2.0) (Gurevich et al., 2013) and combined reported metrics like mean depth of coverage for both time points.

3.3 Results

3.3.1 Reduced sequencing yield but same data quality with expired flow cells

In this study, we present the application of nanopore adaptive sampling on the *in-silico* enrichment of plasmids by depleting chromosomal reads during the sequencing of bacterial isolates. For this purpose, we separately sequenced five bacterial strains - *Campylobacter jejuni*, *Campylobacter coli*, *Salmonella enterica*, *Enterobacter hormaechei* and *Klebsiella pneumoniae* - on four different flow cells, each divided into an adaptive sampling and a control region. We refer to the flow cells according to the adaptive sampling tool used. This section provides an overview of the general sequencing run and sample metrics of the control regions to assess the quality of the four sequencing runs.

First, we investigated the number of active pores on each flow cell (Figure 3.1 c) and the sequencing yield from the control regions in terms of the number of base pairs and the number of reads (Figure 3.2 (a,c)). We observe that flow cell ReadBouncer1 has the highest number of active sequencing pores (1,153) at the start of the run, while the other three flow cells have between 557 and 718 active pores. The fewer active pores can be explained using expired flow cells for these three sequencing runs. Consistent with the number of active pores, flow cell ReadBouncer1 has the highest overall sequencing

Table 3.1: Overview of read length metrics of the five bacterial isolates. The metrics were computed from reads sequenced on the control side of the flow cells where no adaptive sampling was applied.

Reference	Flow cell ID	Mean Length	Median Length	Std. Deviation
Campylobacter jejuni	ReadBouncer1	16,525.65	10,950	17,083.76
	MinKNOW1	13,360.65	8,192	15,072.50
Campylobacter coli	ReadBouncer1	4,375.37	1,580	6,952.86
	MinKNOW1	5,536.69	2,736	6,894.11
Salmonella enterica	ReadBouncer2	9,304.79	6,455	9,131.90
	MinKNOW2	8,679.22	5,952	8,629.98
Enterobacter hormaechei	ReadBouncer2	8,909.21	6,239	8,845.49
	MinKNOW2	8,309.64	8,861	8,192.69
Klebsiella pneumoniae	ReadBouncer2	9,379.93	6,512	9,180.22
	MinKNOW2	8,842.23	5,963	8,928.63

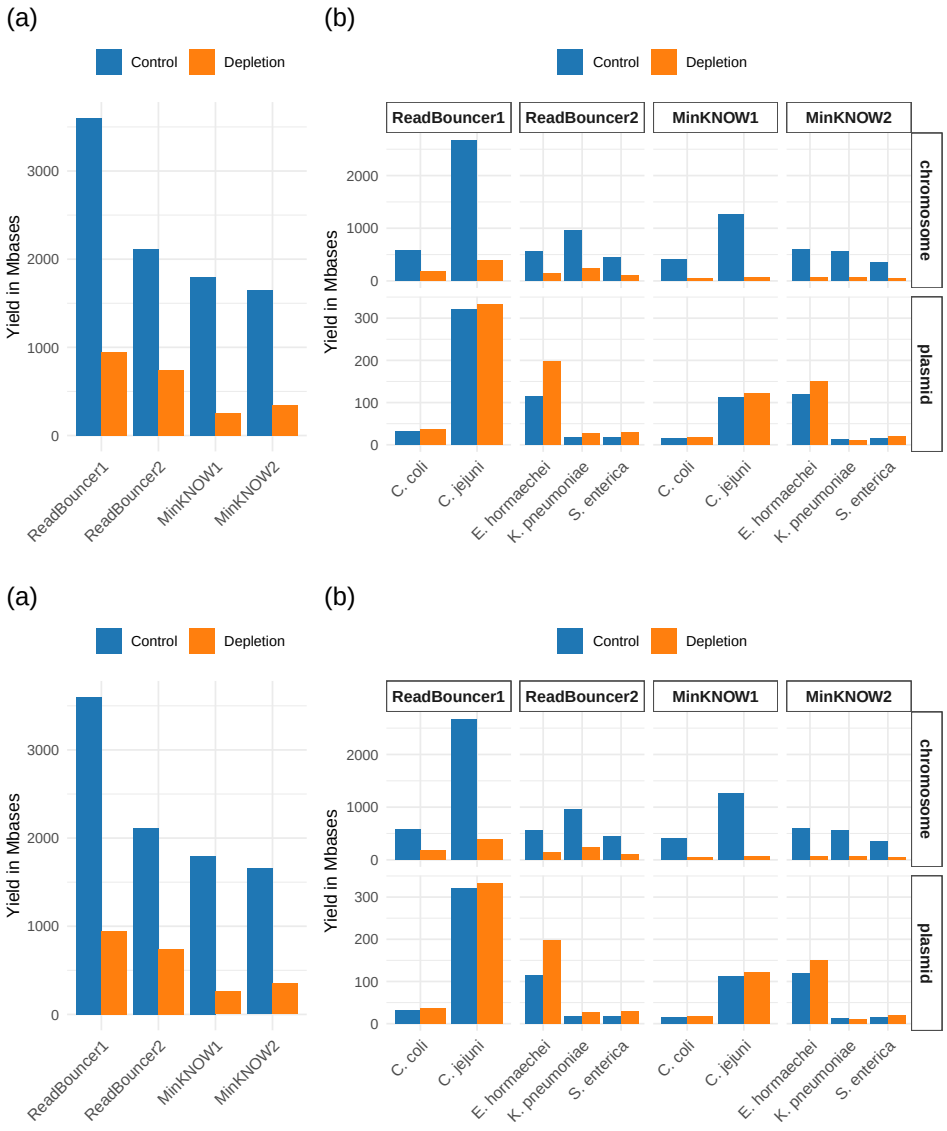

Figure 3.2: Comparison of flow cell yield in terms of sequenced base pairs and reads after 24 hours. **(a)** Yield in Megabases for each flow cell divided by control and adaptive sampling region (Depletion). **(b)** Yield in Megabases for each flow cell region divided by plasmid and chromosome. **(c)** Number of sequenced reads for each flow cell divided by control and adaptive sampling region (Depletion). **(d)** Number of sequenced reads for each flow cell region divided by plasmid and chromosome.

yield (about 4.5 Gigabases). Surprisingly, flow cell MinKNOW2 results in significantly less yield than ReadBouncer2 (2 Gigabases vs. 3 Gigabases), although having a higher number of active pores (718 vs. 636) at the start of the sequencing run. This shows that the number of active pores does not necessarily correlate with flow cell yield for expired

flow cells. The yield on control regions is much higher for all flow cells than on adaptive sampling regions. This observation is consistent with previous studies (Martin et al., 2022; Payne et al., 2021) and can be explained by more overall time spent capturing a new molecule after rejecting one from a pore. In this context, we observe on all four flow cells a higher number of reads sequenced on the adaptive sampling regions than on the control regions (Figure 3.2 (c)). Thus, many reads are classified as chromosomal by the adaptive sampling tools and rejected from the pores, leading to more reads sequenced on the adaptive sampling regions. Here, the flow cell run ReadBouncer2 has a higher number of reads on the adaptive sampling region than ReadBouncer1. This results from a larger number of chromosomal reads that were rejected on the adaptive sampling region of flow cell ReadBouncer2 (approx. 370,000) than on the adaptive sampling region of ReadBouncer1 (approx. 350,000). To further assess and compare the quality of the four sequencing runs, we look at the read length and quality from the control regions of the sequencing runs. The contour plots in Figure 3.3 show that for all four sequencing runs, a large proportion of reads have a mean Phred quality between 12 and 15. Only

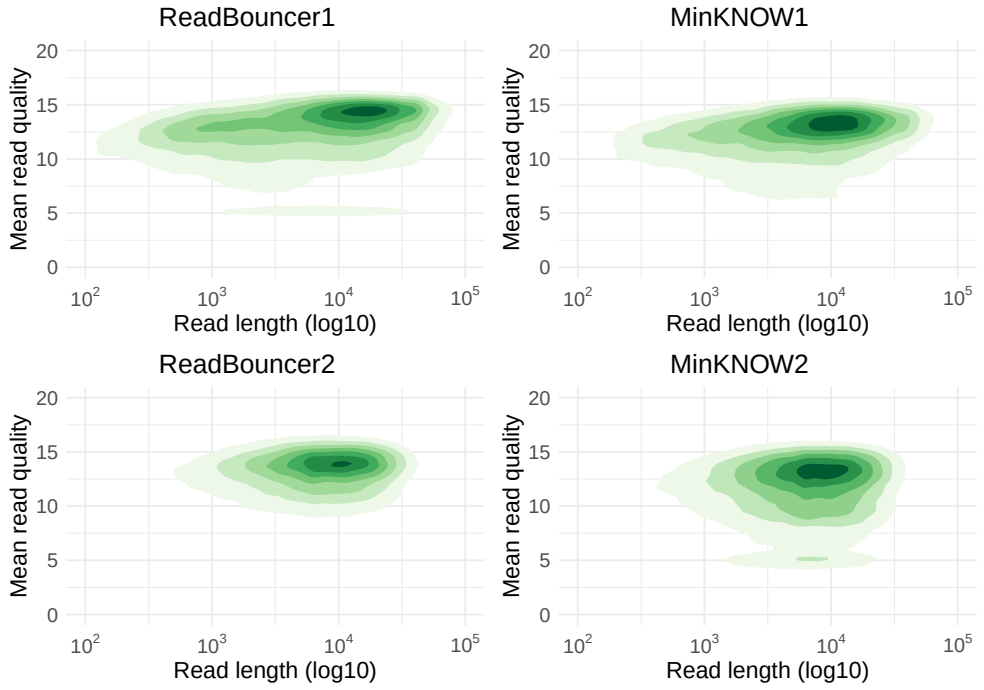

Figure 3.3: Contour plots of read lengths (log scale) against mean read quality for control regions of the four sequencing runs. Darker regions indicate a higher proportion of reads that fall into that slice. For example, ReadBouncer1 and MinKNOW1 have a higher proportion of reads with lengths above 10,000 bp than for ReadBouncer2 and MinKNOW2.

for runs ReadBouncer1 and MinKNOW2, we observe a significant proportion of reads with low mean phred quality between 5 and 7. This suggests no drop in per-read quality when using expired flow cells for normal sequencing. We can also not observe any read length-related quality drop for the expired flow cells. Regarding read lengths, we can only compare ReadBouncer1 with MinKNOW1 and ReadBouncer2 with MinKNOW2. For ReadBouncer1 and MinKNOW1, both having the same *Campylobacter* samples sequenced, we see a larger proportion of longer reads above 10,000 bp for ReadBouncer1. In order to investigate that difference, we looked into per-sample read length metrics, which are provided in Table 3.1.
First, the sequencing data after base calling from the control regions show a large difference in mean and median read lengths and the standard deviation for the two *Campylobacter* samples. Here, the application of the MagAttract HMW Genomic Extraction Kit (Qiagen) results in larger read lengths for the *C.jejuni* samples. We also see a difference between the read lengths of the same sample from the two *Campylobacter* sequencing runs (ReadBouncer1 and MinKNOW1), particularly for *C. coli*. However, there is no trend that read lengths on expired flow cells are generally shorter because the read lengths for *C. coli* are longer on the expired flow cell MinKNOW1 when compared to ReadBouncer1. This suggests that sample handling and preparation, as well as the used barcodes, could have more influence on the read length than flow cell expiration. For the other two flow cells, we observe that the mean read quality is better for ReadBouncer2, and the read lengths are longer for each sample when compared to MinKNOW2. Although both flow cells were expired, we cannot say whether the flow cells' quality could have influenced both metrics. However, the number of active pores for MinKNOW2 at the sequencing start was higher than that for ReadBouncer2, which suggests that this number is not a reliable indicator of the quality of the sequenced data.

3.3.2 Adaptive sampling reduces the number of active channels and sequencing yield, but not read quality

One of the major aspects of our study is the investigation of the impact adaptive sampling has on expired nanopore flow cells. In Figure 3.2, we see that for all four flow cells, the sequencing yield on adaptive sampling regions is significantly reduced in comparison to control regions. This observation aligns with previous studies (Martin et al., 2022; Payne et al., 2021) and originates from a reduced overall time spent for sequencing the DNA and more overall time needed to capture the DNA molecules when adaptive sampling is applied. However, we do not see a higher relative yield reduction for adaptive sampling regions on expired flow cells ReadBouncer1, MinKNOW1, and MinKNOW2.

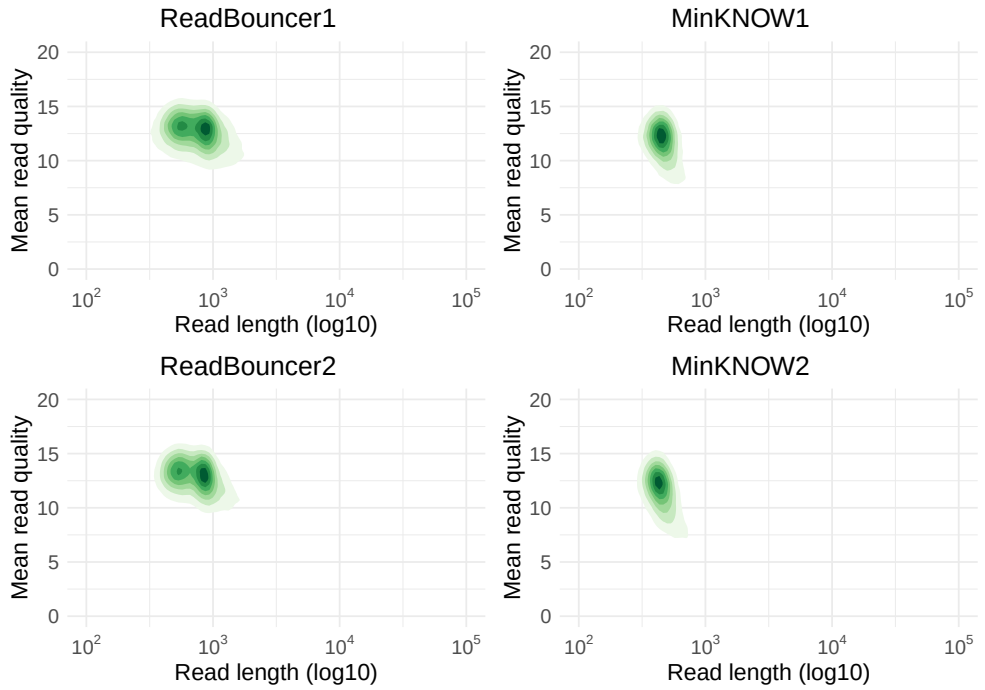

Figure 3.4: Contour plots of read lengths (log scale) against mean read quality for adaptive sampling regions of the four sequencing runs. Darker regions indicate a higher proportion of reads that fall into that slice. For example, most reads from sequencing runs MinKNOW1 and MinKNOW2 have read lengths of about 650 bp and a phred quality value of around 12.

In order to check if adaptive sampling leads to faster pore exhaustion on expired flow cells, we further investigated the effect of adaptive sampling on the number of active sequencing channels and yield in sequenced Mbases per hour (see Figure 3.5). Comparing the four experiments, we consistently observe fewer active sequencing channels on flow cell regions with adaptive sampling than in control regions across all experiments. In summary, we find between 1.4 to 2.6 times more active channels in control regions than in adaptive sampling regions. However, we could not detect bigger systematic differences in active sequencing channels on expired flow cells when compared to the fresh flow cell ReadBouncer1. Finally, we also compared the average read quality scores from reads sequenced on control regions (Figure 3.3) with those sequenced on adaptive sampling regions (Figure 3.4). This comparison shows no significant loss in read quality when applying adaptive sampling to expired flow cells.

3.3.3 Rejecting chromosomal reads increases the relative plasmid abundance

In our four experiments, we investigate the potential enrichment of plasmid sequences in bacterial samples by rejecting the chromosomal reads using adaptive sampling. First, we calculated the percentage of sequenced chromosomal and plasmid base pairs for each sample from the adaptive sampling and control regions. We refer to the percentage of plasmid base pairs as the relative plasmid abundance in a sample. In Figure 3.6, we see that after 24 hours of sequencing, adaptive sampling increases the relative abundance of plasmid base pairs (bp) for all samples on the four flow cells. For instance, we could increase the abundance of *Campylobacter coli* plasmid bases from 3.68% to 24.75% when rejecting chromosomal reads with `MinKNOW`. We further observe that plasmid abundances are much higher when using `MinKNOW` instead of `ReadBouncer` for adaptive sampling. Since `ReadBouncer` has shown a higher read classification

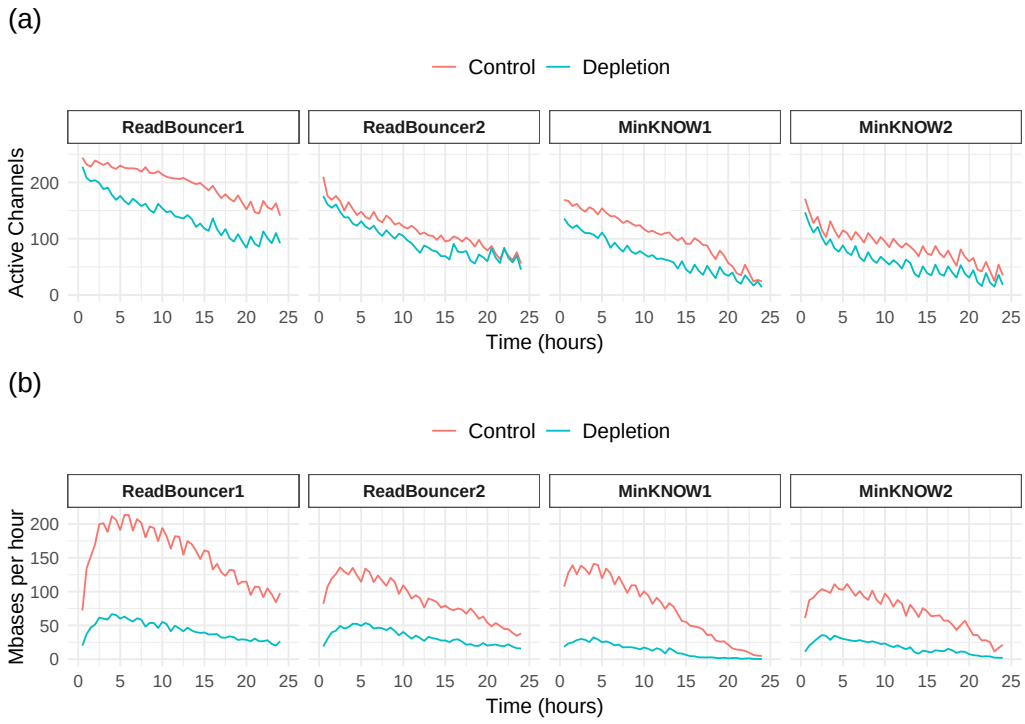

Figure 3.5: Comparison of active channels and yield between control and adaptive sampling (depletion) regions in all four experiments. **(a)** Plots showing how the number of active channels varies with time in adaptive sampling (depletion) and control regions. There are more active sequencing channels in control regions on all four flow cells **(b)** Hourly yields from depleted channels vs control channels. Usage of adaptive sampling results in lower overall sequencing yield compared to normal sequencing.

accuracy in a previous study, we suggest that ReadBouncer needed more time than MinKNOW to reject unwanted chromosomal DNA molecules. Thus, we investigated the lengths of rejected reads in the final output of the experiments by using information from the sequencing summary files (*end_reason* = *data_service_unblock_mux_change*) produced by the MinION control software. The read length histogram in Figure 3.7 compares the lengths of reads rejected by MinKNOW with those rejected by ReadBouncer. In the histogram, we see that reads rejected by ReadBouncer are much longer than those rejected by MinKNOW, with an average length of 848 bp compared to 520 bp. This confirmed our assumption that ReadBouncer rejects reads later during the adaptive sampling process resulting in a higher abundance of unwanted chromosomal base pairs in the final output. To avoid confusion, we have to note that the lengths of rejected reads in the final output are not the same as the read prefix (or chunk) length used by adaptive sampling tools for making rejection decisions. Lengths of rejected reads in the

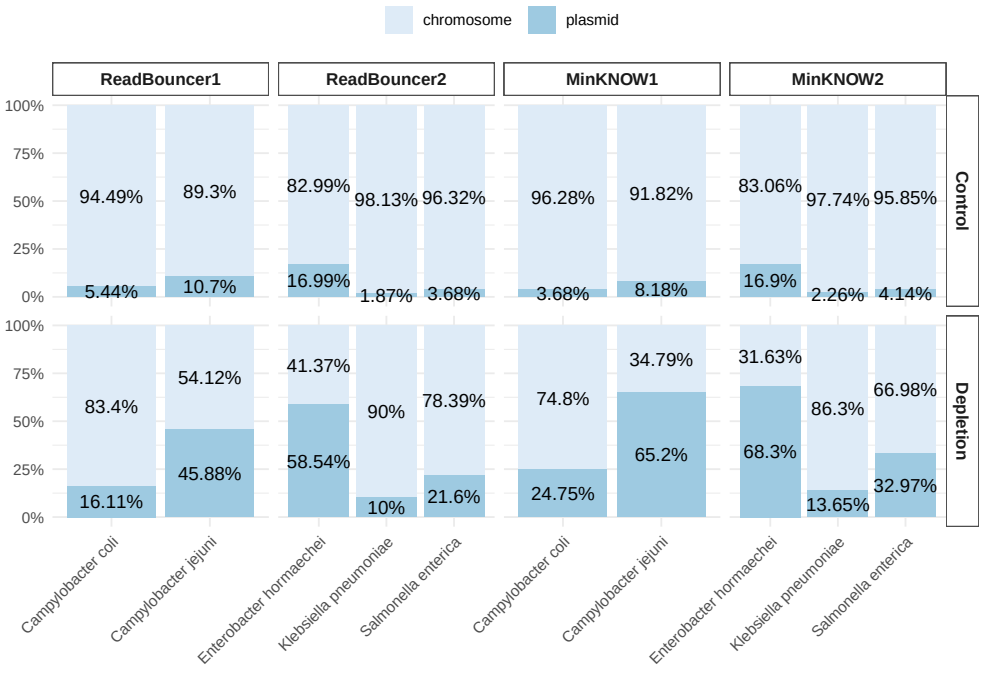

Figure 3.6: Comparison of plasmid abundances in five bacterial samples. Adaptive sampling with MinKNOW was used on flowcells MinKNOW1 and MinKNOW2 and ReadBouncer was used as adaptive sampling tool on flowcells ReadBouncer1 and ReadBouncer2. For all experiments, plasmid abundances for each sample were measured after 24 hours of sequencing for control regions and adaptive sampling regions (Depletion). Plasmid abundances are highest when using MinKNOW for depletion of chromosomal nanopore reads.

final output represent the time needed for the whole decision process, including time for communication with the API and mapping of reads against index data structures.
We further examined whether the plasmid enrichment by composition and yield we observe in our experiments corresponds to the predicted enrichment by the mathematical model proposed by Martin et al. (2022). We calculated the enrichment by composition by dividing the relative plasmid abundance from adaptive sampling regions by the relative plasmid abundance from control regions. Accordingly, we calculated the enrichment by yield using the number of sequenced plasmid base pairs from adaptive sampling and control regions. As predicted by the model, the enrichment factor was higher for samples with less abundant plasmids (Figure 3.8 (a)). The highest levels of enrichment by composition were obtained using `MinKNOW`, which can be explained by faster rejection decisions. The predictions from the mathematical model by Martin et al. (2022) correlated moderately with our observations (Pearson's $r = 0.55$) as shown in Figure 3.8 (b). In contrast, the original plasmid abundance has no impact on the enrichment by yield (Figure 3.8), with enrichment by yield being significantly less than enrichment by

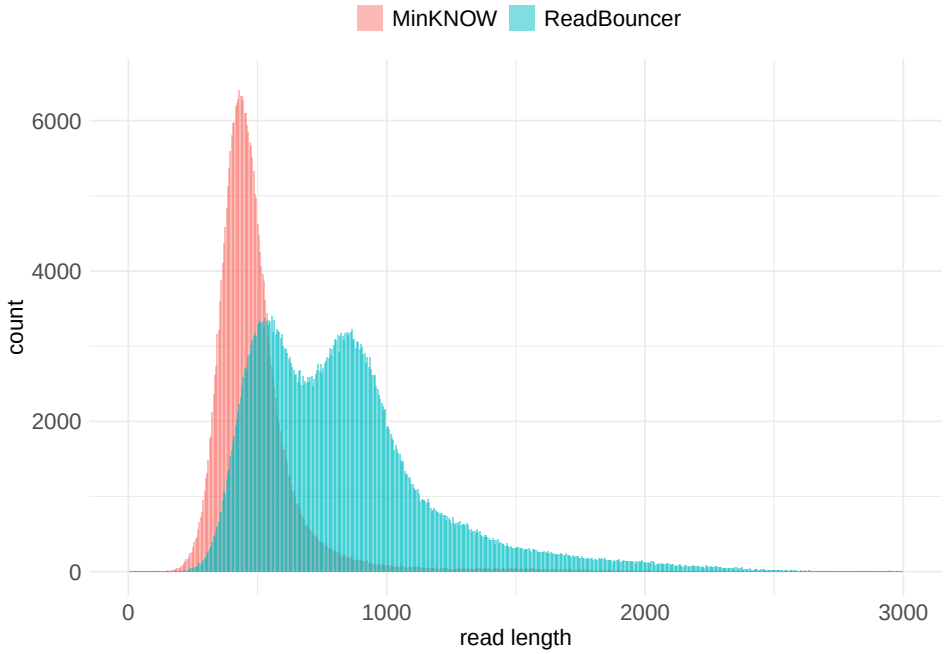

Figure 3.7: Read length histogram of rejected reads by `MinKNOW` and `ReadBouncer`. Adaptive sampling with `MinKNOW` leads to rejected read lengths of about 450 to 500 bp while lots of rejected reads are even longer than 1,000 bp when using `ReadBouncer`

composition. We also noticed that the predicted enrichment values by the model do not correlate with the observed enrichment values by yield (Pearson's $r = -0.07$, Figure 3.8 (d)).

3.3.4 Effective enrichment of plasmids by yield, read number and mean depth of coverage

We examine the effective plasmid enrichment at different time points of sequencing for each experiment by calculating the plasmid enrichment for the five bacterial species in 30-minute intervals. According to equation 3.1, the enrichment by yield is the ratio of

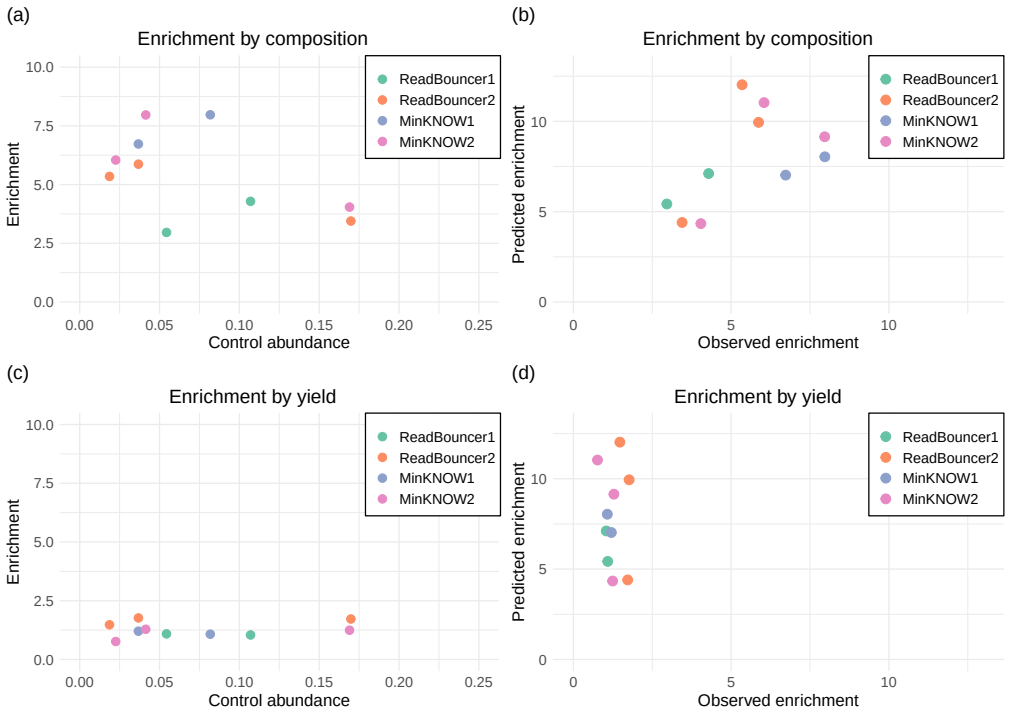

Figure 3.8: Scatterplots for plasmid enrichment by composition and yield. (a) Enrichment factor by composition against relative abundance. Each point represents a bacterial sample, with the position on the x-axis indicating the original relative abundance of plasmids in the sample and the position on the y-axis indicating the enrichment factor obtained. (b) Correlation between observed enrichment values by composition and predicted enrichment values by the mathematical model (Pearson's r of 0.55). (c) Enrichment factor by yield against relative abundance. Each point represents a bacterial sample, with the position on the x-axis indicating the original relative abundance of plasmids in the sample and the position on the y-axis indicating the enrichment factor obtained. (d) Correlation between observed enrichment values by yield and predicted enrichment values by the mathematical model (Pearson's r of -0.07).

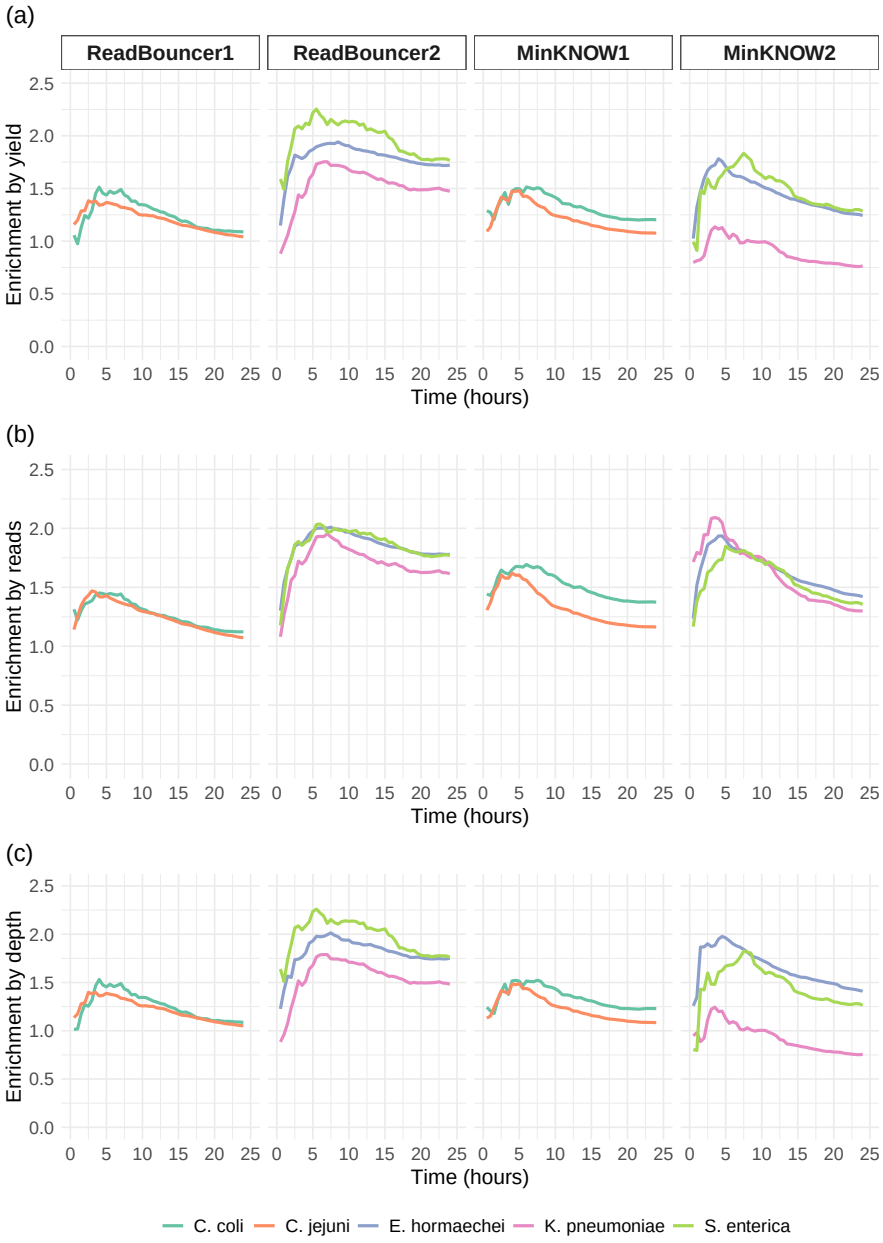

Figure 3.9: Comparison of enrichment in five bacterial samples. **(a)** Enrichment by number of sequenced plasmid bases for all five bacterial strains across the four sequencing runs. **(b)** Enrichment by number of plasmid reads for the five bacterial strains across all four sequencing runs. **(c)** Enrichment by mean depth of coverage of plasmid references for the five bacterial strains across the four sequencing runs. All strains but the *Klebsiella pneumoniae* sample, where MinKNOW was used for adaptive sampling, show a slight enrichment.

cumulative plasmid bases from the adaptive sampling region and the control region at time point t. We calculate the enrichment by the number of plasmid reads and mean depth of coverage in the same manner as proposed by equations 3.2 and 3.3. Figure 3.9 (b) illustrates that we obtain an enrichment of plasmid reads for all samples in all experiments at any given time point. This observation confirms that the number of sequenced plasmid reads can be increased by using adaptive sampling. We can see the same effect for the enrichment by yield (Figure 3.9 (a)) for all but one sample. For the *Klebsiella pneumoniae* sample of flow cell MinKNOW2, we observe that the number of plasmid bases from adaptive sampling is less than that from the control channels. Thus, we failed to obtain an enrichment of *Klebsiella pneumoniae* plasmids in that experiment where we used `MinKNOW` to deplete chromosomal reads. For all other samples, we observe an enrichment of 1.1x to 1.8x after 24 hours, corresponding to $10 - 80\%$ more plasmid data when using adaptive sampling, even when using expired flow cells with reduced active pores.

We further investigated the difference in enrichment between the same samples from experiments ReadBouncer2 and MinKNOW2. First, flow cell MinNKOW2 has fewer active sequencing channels and produces less sequencing yield than the flow cell from experiment ReadBouncer2 (see Figure 3.5). Figure 3.4 also illustrates that the average read quality in the adaptive sampling region of flow cell MinKNOW2 is smaller than for flow cell ReadBouncer2. Both observations suggest a decreased pore quality of flow cell MinKNOW2. Although this might explain the reduced enrichment by yield in this experiment, it does not explain why there is an effective depletion of plasmid bases for *Klebsiella pneumoniae* in experiment MinKNOW2. Thus, we identified all reads from the final output that mapped against the *Klebsiella pneumoniae* plasmids but were rejected by `MinKNOW`. We extracted these falsely rejected plasmid reads of *Klebsiella pneumoniae* and mapped them to the corresponding bacterial chromosome reference sequences with `minimap2` (Li, 2018). Using `samtools depth` (Li et al., 2009), we could identify four regions (between 829 and 2,101 bp long) on the *Klebsiella pneumoniae* chromosome with read depth ≥ 10. These findings reveal regions of high identity between *Klebsiella pneumoniae* plasmid targets and non-target chromosome sequences. Such similar regions between target and non-target sequences pose a challenge for the application of nanopore adaptive sampling and potentially lead to an increased number of falsely rejected target reads. Here, it seems that `ReadBouncer` can avoid a high number of false rejections by using longer read prefixes (see Figures 3.7 and 3.4) for making rejection decisions. Our observations suggest that using more sequence information by increasing the chunk size for adaptive sampling with `MinKNOW` could circumvent such issues.

3.3.5 Adaptive sampling helps improving plasmid assemblies

Table 3.2: Plasmid Assembly statistics of adaptive sampling and control region after one and two hours of sequencing for two different bacterial isolates from sequencing runs ReadBouncer2 and MinKNOW2. All reads from the adaptive sampling and control regions were separately assembled for *Salmonella enterica* and *Enterobacter hormaechei* using `Flye` assembler. Assembly statistics provided by `Quast` show better results for plasmid assemblies from adaptive sampling than control regions.

	Salmonella enterica				***Enterobacter hormaechei***			
	ReadBouncer		**MinKNOW**		**ReadBouncer**		**MinKNOW**	
Adaptive Sampling (channels 1-256)								
Time (hours)	1	2	1	2	1	2	1	2
Ref. plasmids	1	1	1	1	4	4	4	4
Assembled plasmid contigs	1	1	2	1	4	4	6	5
Plasmid reads	108	277	80	194	1,383	3,656	1,469	3,831
Ref. avg. coverage depth	11	29	5	19	14	43	13	37
Ref. coverage $\geq$ 5x(%)	100	100	84.67	100	98.41	99.84	91.99	99.84
Ref. coverage $\geq$ 10x(%)	70.81	100	2.8	100	54.28	99.84	53.12	97.68
Mismatches per 100kb	136	126	358	132	55	26	94	38
Indels per 100kb	653	652	895	656	89	25	136	40
Control (channels 257-512)								
Time (hours)	1	2	1	2	1	2	1	2
Ref. plasmids	1	1	1	1	4	4	4	4
Assembled plasmid contigs	1	1	1	1	6	4	6	5
Plasmid reads	79	160	59	128	945	2,136	971	2,186
Ref. avg. coverage depth	7	15	7	13	11	25	10	24
Ref. coverage $\geq$ 5x(%)	98.41	100	88.38	100	82.66	99.34	73.8	99.84
Ref. coverage $\geq$ 10x(%)	31.21	96.98	21.07	87.24	25.46	89.4	23.06	90.7
Mismatches per 100kb	219	133	397	144	193	31	306	93
Indels per 100kb	744	654	764	652	238	43	488	137

Our experiments demonstrated an effective enrichment of plasmids after 2-5 hours by using adaptive sampling. Since plasmid assemblies are possible after 3-4 hours of sequencing without adaptive sampling (Taylor et al., 2019), we wanted to see if adaptive sampling enables faster plasmid assemblies. In order to evaluate the effect of adaptive sampling on the assembly of low-abundant plasmids, we took the reads available after one hour and two hours of sequencing from the adaptive sampling and control regions two of the bacterial isolates, *Salmonella enterica* and *Enterobacter hormaechei*. We did not include the *Campylobacter* strains in this analysis because the coverage of plasmids from sequencing only two isolates on a fresh flow cell (sequencing run ReadBouncer1) was extraordinarily high and assembly statistics would not be comparable between the sequencing runs ReadBouncer1 and MinKNOW1. We also did not include *Klebsiella pneumoniae* because of the findings mentioned in the last subsection that could bias our

analysis.
We separately assembled all reads for the control and adaptive sampling regions using `metaFlye` assembler (Kolmogorov et al., 2020). After one round of polishing with `Medaka consensus`, we measured quality metrics for the final assemblies using `Quast` (Gurevich et al., 2013). Table 3.2 shows results after one and two hours of sequencing the two bacterial isolates from sequencing runs ReadBouncer2 and MinKNOW2. In all cases, plasmid assemblies were improved by using adaptive sampling, with reduced numbers of mismatches and indels for plasmid assemblies from the adaptive sampling regions. This can be explained by an increased sequencing depth and reference coverage due to the plasmid enrichment by adaptive sampling. However, we recognize some gaps in *Enterobacter hormaechei* plasmids assembled using reads from the adaptive sampling region of run MinKNOW2. This can be caused by similar regions between target and non-target sequences, as we have observed for *Klebsiella pneumoniae*. In general, adaptive sampling can improve plasmid assemblies and enables assemblies even after 2 hours of sequencing on flow cells with fewer active pores.

3.4 Discussion

Recent studies have demonstrated the utility of adaptive sampling for the enrichment of underrepresented sequences in various applications, such as host depletion in human vaginal samples or antibiotic resistance gene enrichment in metagenomics samples. In this study, we examine the potential of adaptive sampling for the enrichment of low-abundant plasmid sequences by rejecting chromosomal sequences in bacterial isolate samples. We demonstrate the possibility of using even older or expired flow cells with fewer active sequencing pores for the *in-silico* enrichment via adaptive sampling. Since we wanted to know if enrichment is independent of the adaptive sampling tool, we evaluated plasmid enrichment for two tools, namely `ReadBouncer` and ONT's `MinKNOW` sequencing control software. Although we observed different levels of plasmid enrichment, the tools consistently enriched for low-abundant plasmid sequences. Our study was by no means designed to benchmark different adaptive sampling tools, which would require the inclusion of more tools and a setup that ensures that all tools use the same amount of sequence information for making rejection decisions.
The enrichment by yield, the most critical value for researchers, lies for all but one sample in our experiments between 1.1x and 1.8x after 24 hours of sequencing on an ONT MinION sequencing device. We also demonstrated that the difference between enrichment by yield, number of reads, and mean depth of coverage is negligible in all our samples. High-quality assemblies of plasmids are possible within two hours of

sequencing with adaptive sampling and show even better results than plasmid assemblies without adaptive sampling. These results reflect the benefit of adaptive sampling in assembling low-abundant plasmid sequences. Since we sequenced three bacterial isolates on only half a reused flow cell, we reason that up to 20 bacterial isolates can be sequenced on a flow cell with adaptive sampling for plasmid enrichment.

Our experiments showed that expired flow cells with decreased number of active pores could be used in combination with adaptive sampling. Previous studies demonstrated that the number of active sequencing pores decreases faster when using adaptive sampling. Although we show the same trend in our study, we do not see a negative impact on the enrichment of target sequences and the average quality of sequenced reads. Thus, we encourage researchers to use flow cells with reduced active pores in adaptive sampling experiments for more sustainable lab experiments and cost savings in core facilities and larger research institutions.

Our results show that rejecting chromosomal sequences with adaptive sampling increases the abundance of plasmid sequences in the final output. Dependent on the plasmid abundance in the original sample, the values for plasmid enrichment by composition are between 2.5x and 8x. These observations moderately correlate with the predictions from the mathematical model proposed by Martin et al. (2022). Furthermore, a consistent enrichment of plasmid sequences with regard to the number of base pairs, number of reads, and depth of coverage was shown by using adaptive sampling. Independent of the size of the sequencing libraries, we could increase the amount of sequenced plasmid base pairs by 10-80% after 24 hours of sequencing. However, in one experiment, we recognized the depletion of plasmid sequences of *Klebsiella pneumoniae* after 24 hours when ONT's `MinKNOW` was used as an adaptive sampling tool. Our investigations reveal that regions with high sequence identity located both on the chromosome and the plasmid lead to false read rejections, which result in a depletion of the targeted plasmid sequences. This highlights potential issues with the usage of nanopore adaptive sampling and sounds a note of caution if target and non-target sequences are similar. We hypothesize from our findings that using larger read chunks for making rejection decisions could circumvent this issue. However, such an examination is beyond the scope of this study and needs systematic investigations to find the optimal read chunk length that minimizes false rejections decisions while still rejecting unwanted reads fast enough to obtain sufficient enrichment.

Both adaptive sampling tools used in this study need known reference sequences to reject the chromosomal reads. If the bacterial species in the given sample are unknown, a more extensive reference database of all potential bacterial chromosome references must be used to enrich plasmids successfully. Alternatively, researchers could also do a targeted

enrichment of the plasmids by using plasmid databases such as PLSDB (Schmartz et al., 2022) and reject all reads that do not match the database. However, this approach risks missing unknown plasmids not covered by the database. Using specific plasmid markers, like the origin of replication, for correctly classifying unknown plasmids is also tricky in an adaptive sampling experiment. Here, the specific markers would need to be located on the first 1,000 bp of the read to prevent false rejection of plasmid reads. These limitations reinforce the need for improved classification algorithms that can even classify reads from unknown plasmids based on the raw nanopore signals.

We envisage several applications for the *in-silico* enrichment of plasmids in the near future. One possibility is the surveillance of plasmid outbreaks in hospital settings. Here, clinicians are interested in studying the transmission of specific antimicrobial resistance genes (ARGs) harboring plasmids from one bacterial species to another. Such community transmissions can indicate the selection pressure on bacteria caused by antibiotic pharmaceuticals and help decide on the corresponding drugs' future usage.

Another possible application of adaptive sampling is the improvement of known bacterial assemblies. In this study, we demonstrated the improved time-to-assembly of plasmids by depleting the known bacterial chromosomes. We plan to develop a pipeline for the real-time *de novo* assembly of bacterial isolates in the future. Using adaptive sampling, we could reject reads that cover assembled regions with a minimum depth of coverage, enriching for unseen or assembled regions with low sequencing depth. In such a way, we could complement the dynamic re-sequencing framework BOSS-RUNS (Weilguny et al., 2023) with a dynamic *de novo* adaptive sampling framework. We believe this could improve both the quality of bacterial and plasmid assemblies as well as metagenomics assemblies of unknown bacterial species.

4 Fast and space-efficient taxonomic classification of long reads with hierarchical interleaved XOR filters

Summary

Metagenomic long-read sequencing is gaining popularity for various applications, including pathogen detection and microbiome studies. To analyze the large data created in those studies, software tools need to taxonomically classify the sequenced molecules and estimate the relative abundances of organisms in the sequenced sample. Due to the exponential growth of reference genome databases, the current taxonomic classification methods have large computational requirements. This issue motivated us to develop a new data structure for fast and memory-efficient querying of long reads. Here, we present `Taxor` as a new tool for long-read metagenomic classification using a hierarchical interleaved XOR filter (HIXF) data structure for indexing and querying large reference genome sets. `Taxor` implements several k-mer-based approaches, such as syncmers for pseudo-alignment to classify reads and an expectation maximization (EM) algorithm for metagenomic profiling. Our results show that `Taxor` outperforms competing short- and long-read tools regarding precision while having a similar recall. Most notably, `Taxor` reduces the memory requirements and index size by more than 50% and is among the fastest tools regarding query times. This enables real-time metagenomics analysis with large reference databases on a small laptop in the field.

This chapter is based on (Ulrich & Renard, 2023), which is a joint work with Bernhard Y. Renard. A detailed description of the authors' contributions can be found in section Thesis outline.

4.1 Background

Identifying organisms in an environmental or clinical sample is a fundamental task in many metagenomic sequencing projects. This includes the detection of pathogens in

samples with a large host background (Andrusch et al., 2018), as well as studying the composition of microbial communities composed of bacteria, archaea, viruses, and fungi (Doytchinov & Dimov, 2022). Over the last years, many tools have been developed that classify short and long sequencing reads by comparing their nucleotide sequences with a predefined set of references (Dilthey et al., 2019; Kim et al., 2016; Wood et al., 2019). While each tool uses a different classification strategy, they all try to resolve the species present in the sample and determine their relative abundances (Fischer et al., 2017; Lindner & Renard, 2013).

Among the different read classification strategies, alignment-based approaches were the first used for taxonomic profiling. Tools like `SLIMM` (Dadi et al., 2017), `DUDes` (Piro et al., 2016) or `PathoScope` (Hong et al., 2014) use the results of common read mappers like `Bowtie2` (Langmead & Salzberg, 2012) and bin the sequencing reads across the different reference genomes. Although these methods have high accuracy, their computational performance decreases tremendously when using entire public databases such as NCBI RefSeq or GTDB as reference datasets. Thus, high-performance computing clusters are needed to run these tools in a reasonable amount of time and fulfill their memory requirements. In contrast, marker-based approaches such as `MetaPhlAn2` (Truong et al., 2015) and `mOTUs2` (Milanese et al., 2019) identify bacterial and archaeal species by their 18S or 16S rRNA genes. However, this approach is infeasible for viruses since they have no universally conserved genes. More recent taxonomic classification strategies rely on machine-learning approaches. Tools such as `DeepMicrobes` (Liang et al., 2020) and `BERTax` (Mock et al., 2022) show promising results for classifying reads on higher taxonomic levels but perform poorly at genus and species levels. Most state-of-the-art taxonomic profilers, like `Kraken2` (Wood et al., 2019), `Ganon` (Piro et al., 2020) and `KMCP` (Shen et al., 2023), use k-mer-based methods for read classification. In the first step, these methods count the exact matches of substrings of length k among the different reference sequences in the database and use further statistical analysis to assign reads to references. These profiling tools mainly differ in the indexing of the reference set and/or the k-mer selection method used to calculate the similarity between read and reference sequences.

What all taxonomic classifiers have in common is that they struggle with the ever-increasing amount of reference genomes. Databases such as NCBI RefSeq (O'Leary et al., 2016) and GTDB (Parks et al., 2022) already comprise hundreds of thousands of microbial reference assemblies belonging to 62,000 bacterial species (GTDB Release 207) and 12,000 viral species (RefSeq Release 211) and are constantly increasing. This poses a major computational challenge to the profilers in terms of memory usage, index construction, and query time.

Several approaches for efficient indexing and querying of large collections of reference sequence sets have been developed over the last few years to overcome these issues. The popular `Kraken2` (Wood et al., 2019) classifier uses minimizers and introduces a probabilistic, compact hash table to reduce the size of the index. On the other hand, color-aggregative methods like `Bifrost` (Holley & Melsted, 2020) or `Mantis` (Pandey et al., 2018) use compacted de Bruijn graphs or counting quotient filters for indexing and querying k-mers. Those methods have the disadvantage that they index each reference separately using data structures for approximate membership queries, e.g. Bloom filters (Bloom, 1970). In contrast, Sequence Bloom Tree (SBT) (Harris & Medvedev, 2020; Solomon & Kingsford, 2018; C. Sun et al., 2018) approaches exploit the k-mer redundancy of homogeneous datasets such as those from RNA-Seq experiments to compare large sequence datasets. However, these approaches are unsuitable for heterogeneous k-mer sets, such as microbial genomes. Tools like `BIGSI` (BItsliced Genomic Signature Index) (Bradley et al., 2019), `COBS` (Compact Bit-Sliced Signature Index) (Bingmann et al., 2019) and `KMCP` (Shen et al., 2023), which are based on Bloom filter matrices, are promising much better results for taxonomic profiling of large sequencing data sets. Interleaved Bloom Filters (IBFs), which belong to the latter approaches, improve the indexing data structures by combining several Bloom filters (one per reference) in an interleaved fashion while allowing to query all Bloom filters at once (Dadi et al., 2018). The IBF data structure has been used by the taxonomic classifier `Ganon` (Piro et al., 2020) and was recently enhanced by introducing the hierarchical interleaved Bloom filter (HIBF) in a tool called `Raptor` (Mehringer et al., 2023).

Many k-mer-based classifiers use Bloom filters for approximate membership queries of k-mers in large reference data sets. Their popularity is based on their flexibility, low memory requirements, and fast query times. However, there is a small probability that a k-mer is incorrectly reported as being present in a reference sequence, called a false positive. Some tools let the user define the false positive rate and adapt the Bloom filter size and/or the number of hash functions to that value. Although Bloom filters have a low memory footprint, they use 44% more memory than the theoretical lower bound, even when applied in an optimal manner (Graf & Lemire, 2020). Therefore, several advanced probabilistic filters like cuckoo filters (B. Fan et al., 2014; Mitzenmacher et al., 2020) have been developed over the last few years. In particular, XOR filters have been proposed as an alternative to Bloom Filters, using only 23% more memory than the theoretical lower bound (Graf & Lemire, 2022).

Based on the work of Graf and Lemire (2020) and Dadi et al. (2018), we first developed an Interleaved XOR Filter (IXF), which can be used in the same manner as the IBF, and implemented it as part of the Seqan C++ library (Reinert et al., 2017). We then

extended our new data structure to a hierarchical interleaved XOR filter (HIXF) to avoid using more space than necessary when references in the database are highly divergent in size. The HIXF data structure is implemented as part of the taxonomic classification tool `Taxor`, which allows the user to choose different k-mer-based strategies, such as k-mers and minimizers. Since our new tool is specifically designed for long-read metagenomics experiments, we also implemented open canonical syncmers (OCS) as a k-mer selection approach, which has been shown to be superior to minimizers for error-prone long reads (Dutta et al., 2022; Edgar, 2021). In the final step, taxonomic profiling of the query results is performed by utilizing an expectation maximization (EM) algorithm for abundance estimation and re-assignment of classified reads. We compare `Taxor` to five state-of-the-art short- and long-read taxonomic classification tools on simulated and real mock communities. Our results show that `Taxor` can tremendously reduce the index size and memory requirements for queries while still being on par with the evaluated tools regarding precision and recall.

4.2 Methods

We have designed and implemented our novel taxonomic profiling tool, `Taxor`, as a modular workflow that consists of three mandatory steps. First, `Taxor` computes the k-mer content of the input reference genomes and creates an index for each set of reference genomes. The index is a hierarchical interleaved XOR filter (HIXF), a novel space-efficient data structure for approximate membership query (AMQ) that we will describe in the following subsections. In the second step, sequencing reads are queried against one or several HIXF index files, resulting in one intermediate file for each index. These intermediate files contain all matches of the reads against the different reference genomes and must be merged before the final profiling step. Three-step filtering of spurious matches is performed before an EM algorithm computes taxonomic abundances and reassigns reads based on the taxonomic profile. We finally provide three output files containing information about the sequence abundances (based solely on nucleotide abundance), taxonomic binning (read to reference assignments), and taxonomic abundances (normalized by genome size).

4.2.1 Interleaved XOR Filter

Many tools for taxonomic classification of sequencing reads facilitate approximate membership queries of k-mers of reads against k-mer sets of the reference sequences. A common approach to implement approximate membership query (AMQ) is using

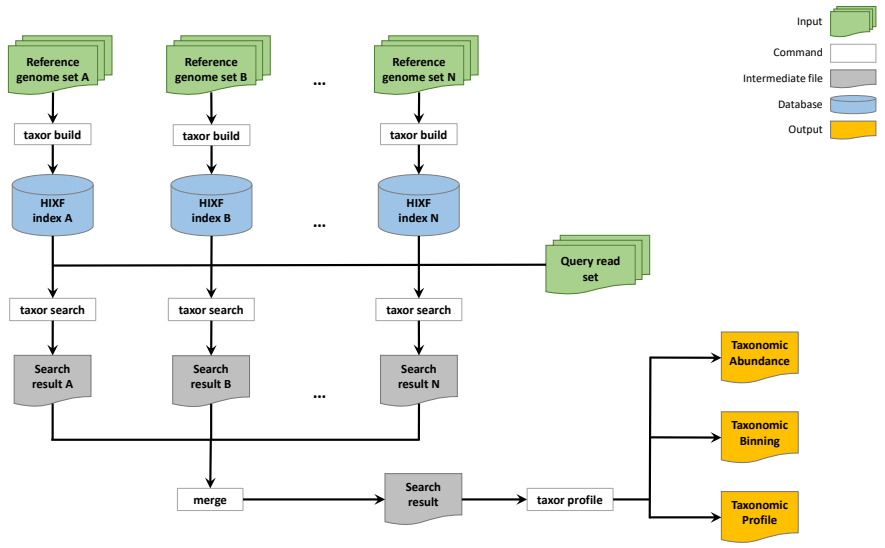

Figure 4.1: Workflow for taxonomic profiling with `Taxor`. A typical workflow for using `Taxor` starts with creating a HIXF index for each reference genome set. Next, a set of sequencing reads is queried against each of the different HIXF index files using the `taxor search` subcommand, which results in an intermediate search result file for each index file. The search result files must be merged before the subcommand `taxor profile` calculates the taxonomic profiling result files.

Bloom filters (Bloom, 1970). Dadi et al. (2018) improved this approach by developing an Interleaved Bloom Filter (IBF) that stores several Bloom filters in one single-bit array that allows querying all Bloom filters simultaneously. Inspired by their approach, we developed an Interleaved XOR Filter (IXF) that combines several XOR filters into one data structure, enabling simultaneous querying of all XOR filters. As described by Graf and Lemire (2020), and similar to Bloom filters, the XOR filter uses three independent hash functions that return a corresponding position in the filter for each key (or k-mer). In a Bloom filter, each bit is considered its own array slot, and bits at the positions to which the hash functions point are set to one. In contrast, in an XOR filter, the bits are grouped together into L-bit sequences, as shown in Figure 4.2. These L-bit sequences in the XOR filter are set in such a way that a bitwise XOR of the three L-bit sequences, corresponding to positions returned by the hash functions, equals the result of the *fingerprint* hash function. While building an XOR filter is almost always successful for larger sets of more than $|S| = 10^7$ elements (Botelho et al., 2007), it can fail for smaller sets, which requires rebuilding with other hash functions. Graf and Lemire (2020) have experimentally shown that the estimated probability for the successful

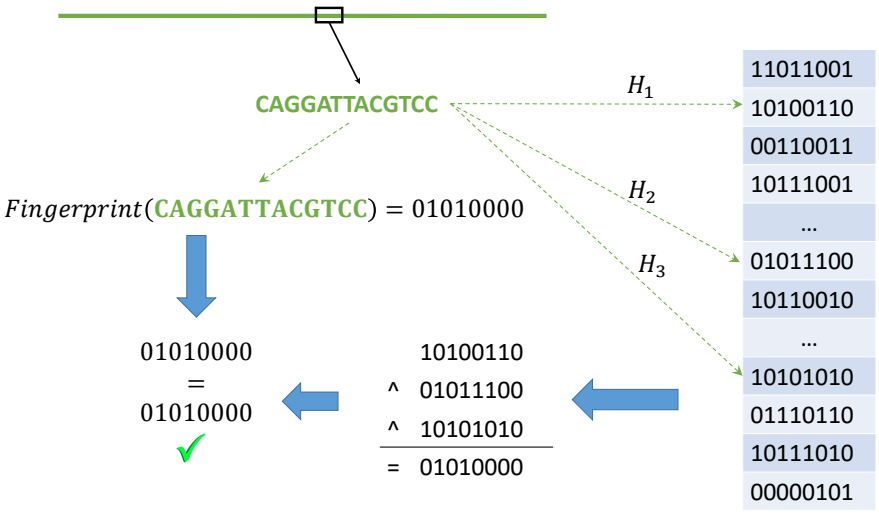

Figure 4.2: Creating an XOR Filter for a given reference sequence. For each k-mer of the reference sequence an L-bit fingerprint and three hash values are calculated. The hash values point to positions in an array consisting of L-bit sequences. The L-bit sequences are set such that a bit-wise XOR of the three L-bit sequences equals the result of the fingerprint function for that k-mer. For example, the fingerprint for the 13-mer $CAGGATTACGTCC$ equals 01010000, and thus a bitwise XOR of the three 8-bit sequences at positions given by the results of the three hash functions H_1 to H_3 will also result in the 8-bit sequence 01010000

building is always greater than 0.8 if the XOR filter size is set to $\lfloor 1.23 \times |S| \rfloor + 32$, which makes this size constraint an optimal compromise between space requirement and build time.

Our IXF implementation combines several XOR Filters (bins) in one single bitvector, using 8-bit sequences for each XOR filter. Therefore, we first need to initially calculate the size of each XOR filter by $\lfloor 1.23 \times |S| \rfloor + 32$, where S is the set of reference k-mers. As for the IBF, the largest XOR filter (or largest reference sequence) determines the size of all bins and, thus, also the size of the entire IXF. If R is the set of reference sequences to be stored in the IXF and S_r the set of k-mers computed for reference r, the size of the IXF can be calculated as follows:

$$Bits_{IXF} = |R| \times \max_{r \in R} \lfloor 1.23 \times |S_r| \rfloor + 32 \tag{4.1}$$

The IXF can be divided into several subvectors, each having the size of the number of bins. Since one bin in the IXF corresponds to exactly one reference sequence, the size of

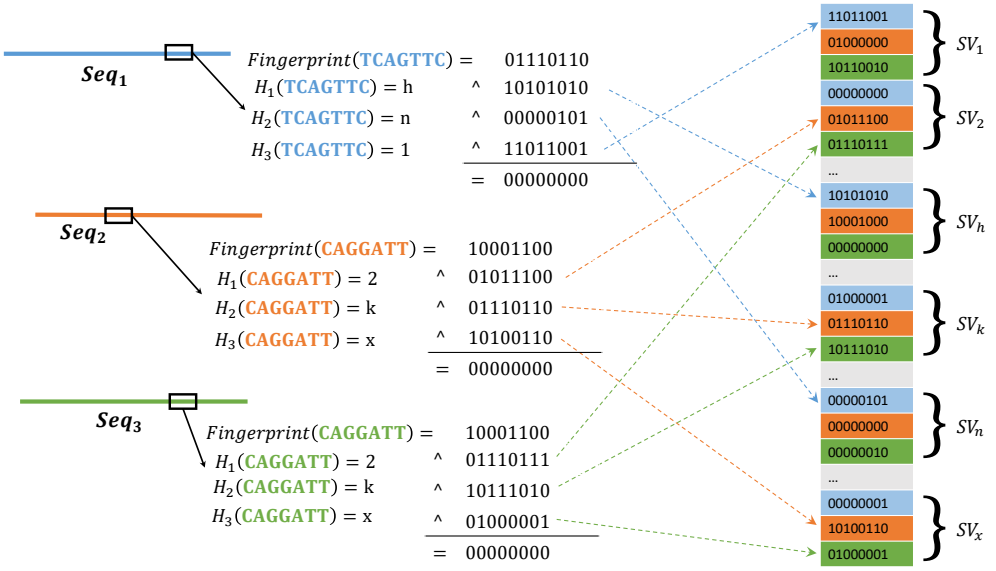

Figure 4.3: Creating an Interleaved XOR Filter for three reference sequences. For each k-mer of the three differently colored reference sequences, an L-bit fingerprint and three hash values are calculated. The hash values determine the three subvectors SV_j in which the L-bit sequence of the corresponding position of the reference sequence is set. L-bit sequences at those positions are set such that a bit-wise XOR of the three L-bit sequences and the fingerprint equal to an $L - bit$ sequence of zeros. For example, the fingerprint for the 7-mer $TCAGTCC$ from Seq_1 equals 01110110, and the three hash functions point to the subvectors SV_1, SV_h and SV_n. Now we set the 8-bit sequences at the first position in each subvector such that a bit-wise XOR of the 8-bit sequences and the fingerprint equals 00000000.

each subvector corresponds to the number of references. When building the IXF for a set of reference sequences, we compute the k-mer sets for each reference sequence and construct the single XOR filters for each reference according to the algorithm proposed by Graf and Lemire (2020). We used the same hash and fingerprint functions for each XOR filter and combined them in an interleaved fashion, as shown in Figure 4.3.

When querying a read against the references stored in the IXF, every k-mer of that read is matched against each XOR filter simultaneously. For each k-mer, we first retrieve the three subvectors SV_j by calculating the three hash values for that k-mer. Next, the resulting L-bit sequence calculated by the fingerprint function is concatenated with itself to the length of the subvectors. Applying a logical bitwise XOR to the three subvectors and the fingerprint vector results in a final L-bit sequence for each reference. If this sequence equals zero, a bit in a binning bitvector is set to one, indicating the presence of the k-mer in the corresponding reference. Combining the binning bit-vectors

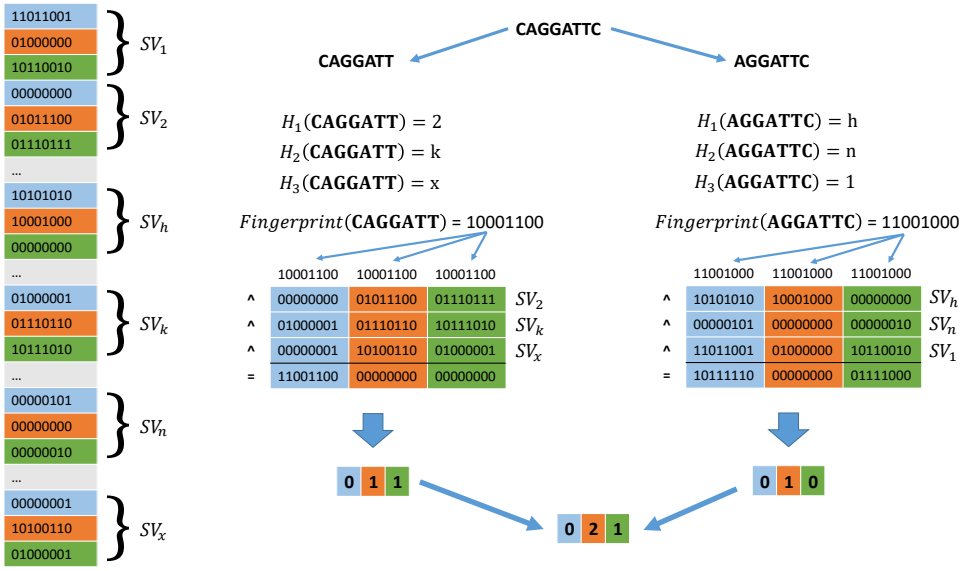

Figure 4.4: Querying an Interleaved XOR Filter consisting of three reference sequences. The read sequence CAGGATTC is divided into two overlapping 7-mers, CAGGATT and AGGATTC. For both k-mers, we separately calculate three hash values by applying the same hash functions used for building the IXF. For example, the hash values of k-mer CAGGATT point us to the subvectors SV_2, SV_k, and SV_x. We further calculate the fingerprint for CAGGATT and concatenate the resulting 8-bit sequence three times because the IXF stores three reference sequences, which results in each subvector having three bins. Applying a bitwise XOR to the three subvectors and the fingerprint vector results in an 8-bit sequence for each reference. For k-mer CAGGATT, the second and the third bins are equal to 00000000, which indicates that the k-mer is present in the second and third reference. This information is stored in a binning bitvector, e.g., 011 for k-mer CAGGATT. Finally, the binning bit-vectors of both k-mers are combined into a counting vector that stores the number of k-mer matches between the read and each reference sequence. Here, the vector 021 indicates that both k-mers are present in the second reference, and thus, the read could match that reference.

of the k-mers to a counting vector finally results in the number of matching k-mers between the read and each reference in the IXF. The example in Figure 4.4 visualizes this process. Here, the read consists of two 7-mers, for which we have to calculate the three hash values that point us to the corresponding subvectors SV_j, e.g., SV_2, SV_k and SV_x for k-mer CAGGATT. A logical bitwise XOR of the three subvectors and the k-mer's fingerprint vector results in an 8-bit sequence for each bin. In Figure 4.4, the resulting 8-bit sequences of references two (orange) and three (green) equal zero, which yields in setting the bit in the binning bitvector for the references to one. Finally, the counters of the corresponding references are incremented in the counting vector of the read. The

resulting counting vector stores the number of matching k-mers between the read and each reference sequence stored in the IXF. Thus, instead of computing three hash values for every XOR Filter separately, we only need to calculate the three hash values once, which poses a significant reduction in computing time to investigate the membership of a k-mer in every XOR Filter. This method lets us quickly count the matching k-mers between the reference genome and a specific sequencing read.

4.2.2 Hierarchical Interleaved XOR Filter

The interleaved nature of the IXF has two important limitations. First, the largest XOR filter determines the overall size of the IXF because all single bins (XOR filters) of the IXF must have the same size. This means that the largest reference sequence dictates the size of the XOR filters storing the k-mer contents of the other reference sequences. Consequently, we would waste a substantial amount of space for smaller references if the reference sequences in the IXF have highly divergent sizes. Second, the query speed slows down with an increasing number of XOR filters stored in the IXF. This is, in practice, not a problem for a few hundred to a few thousand references, but it becomes inefficient when storing many thousands of reference genomes.

To overcome this issue, we adapted the approach by Mehringer et al. (2023) to create a hierarchically structured interleaved XOR filter (HIXF). Here, the idea is to split the k-mer content of larger reference sequences into several smaller k-mer sets while merging the k-mer sets of very small reference sequences into one big set of k-mers. The resulting k-mer sets are stored in a high-level IXF, and for each merged k-mer set, a low-level IXF is stored, holding the k-mer sets of the smaller reference sequences in individual bins (XOR filters). While splitting the k-mer content of large reference sequences and merging the k-mer content of smaller references avoids wasting space, recursively adding an IXF for each merged k-mer set enables querying the individual k-mer sets of the small reference sequences. Depending on the number and size of the reference sequences and the maximum number of bins allowed for each IXF, we can have many levels of the hierarchical interleaved XOR filter (HIXF).

To compute the layout of the HIXF, we utilize the dynamic programming (DP) approach from Mehringer et al. (2023) that finds the optimal balance between space consumption and query speed. Here, we first calculate HyperLogLog sketches (Flajolet et al., 2007) to determine the Jaccard distance between each pair of reference sequences (Baker & Langmead, 2019). Based on this information, the list of reference sequences is rearranged to specify which reference k-mer sets are more similar and, thus, ideal targets for merging on higher levels of the HIXF. Then, the dynamic programming algorithm

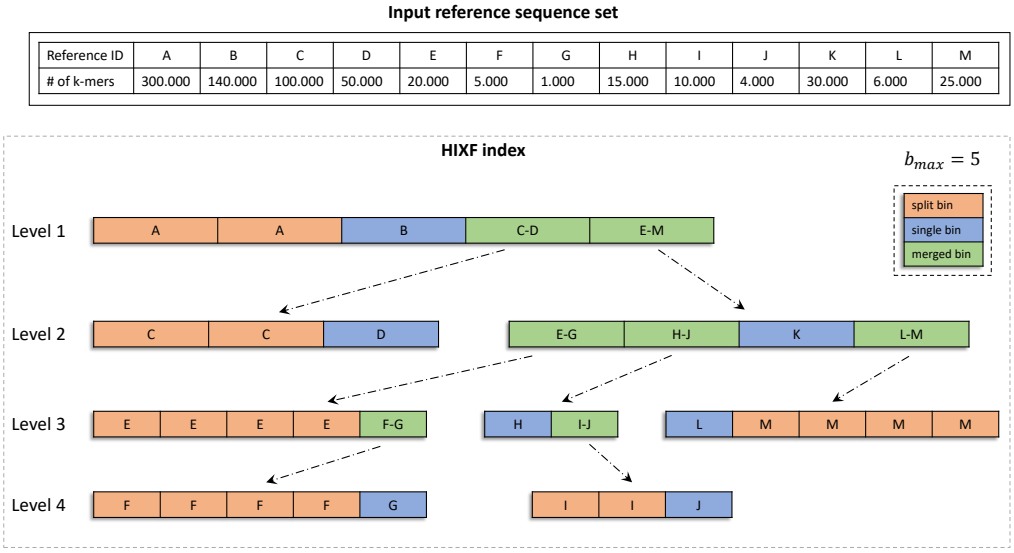

Figure 4.5: Building an HIXF index structure from 13 reference sequences. Exemplary building of a hierarchical interleaved XOR filter from thirteen differently sized reference genomes. Depending on the size of the input sequence's k-mer sets, the HIXF layout is computed. Here, the maximum number of bins per IXF $b_{max} = 5$, which results in 4 levels consisting of 8 interleaved XOR filters. The first level is always a single IXF with exactly B_{max} bins. In the figure, Level 1 consists of two split bins for reference A, one single bin for reference B, one merged bin that stores all k-mers from references C and D, and one that contains the k-mer content of references E to M. On the second level, there is one additional IXF for each merged bin on the first level, resulting in two IXFs in the example above, where the first IXF consists of two split bins for reference C and one single bin for reference D. The second IXF on Level 2 consists of one merged bin containing all k-mers of references E, F and G, one additional merged bin with the k-mer content of references H, I and J, one single bin for reference K and another merged bin with the combined k-mer content of references L and M. Merged bins on each level of the HIXF result in an associated IXF on the next lower level forming a tree-like layout, where the leaves only have split and single bins.

uses a scoring function accounting for the space consumption of each IXF and the number of levels in the HIXF, which finds the optimal splitting and merging operations for the given set of references. Finally, a score is calculated for different values of the maximum number of XOR filters b_{max} in the IXFs, accounting for the expected query time and memory consumption. In general, b_{max} is a multiple of 64, and we calculate the score until the product of query time and space requirement increases. The minimum score determines the maximum number of XOR filters in each IXF, and the backtracing in the corresponding DP matrix returns the layout of the HIXF.

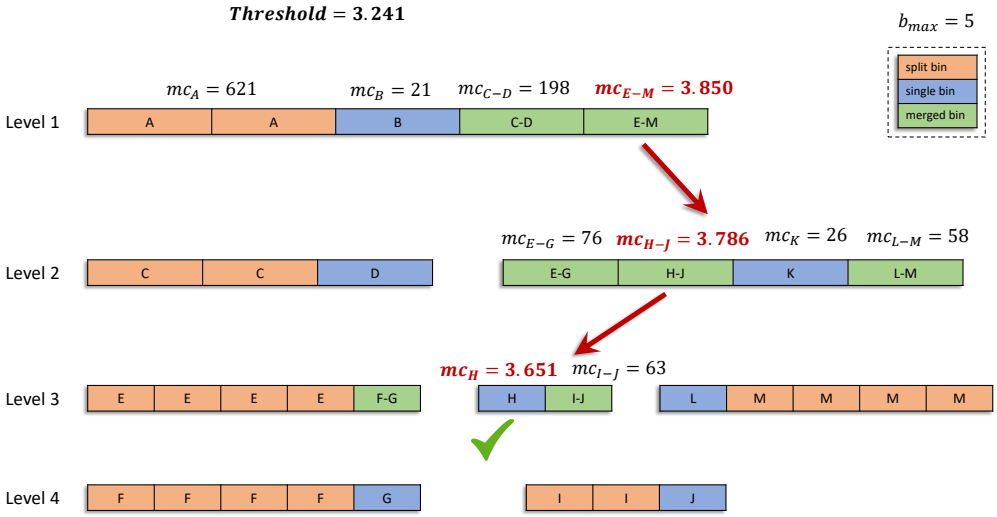

Figure 4.6: Querying a HIXF index structure of 13 reference sequences. When finding matching reference sequences for a given sequencing read, the k-mer content of the read is calculated first. Based on the expected sequencing error rate, a threshold for the minimum number of found k-mers is determined to consider a reference sequence as a hit. Then, the number of matching k-mers with each bin is computed for the first level IXF, considering summed match counts of split bins and match counts of single bins that exceed the threshold as a hit. If the match count for a merged bin exceeds the threshold, we query the associated IXF on the second level. All levels of the HIXF are queried recursively until all reference sequences with k-mer match counts exceeding the threshold have been found. In this example, only the match count for the merged E-M bin exceeds the threshold on the first HIXF level, and we continue querying the associated IXF on the second level. Here, only the merged H-J bin has a k-mer match count above the threshold, which requires querying the associated IXF on the third level, where we find the single bin of reference sequence H to exceed the match count threshold only. Thus, we report reference H as a hit with the queried sequencing read.

When querying the HIXF, we first calculate the k-mer content of the given query read and determine the minimum number of matching k-mers with each reference sequence to be considered a hit. We calculate this threshold based on the k-mer selection scheme described in the following subsection. Then, for each k-mer of the read, the membership in all filters of the top-level IXF is determined, and we further count the total number of k-mers that match each bin in the top-level IXF. If the counter for a bin exceeds the calculated threshold, the read is considered a match with the corresponding reference sequences. For the split bins, the k-mer counts have to be accumulated before thresholding, while we can directly answer the query for single bins that are neither

merged nor split. For merged bins that exceed the threshold, we apply the same procedure recursively on the associated child IXF on the next lower level. This approach allows us to skip querying all lower-level IXFs whose upper-level merged bins do not exceed the given threshold. As a final result, we obtain a list of all reference sequences that exceed the minimum number of matching k-mers with the read under investigation.

4.2.3 K-mer selection & thresholding

In the previous subsections, we introduced our new HIXF data structure describing the usage of membership queries for all k-mers of a given read against all k-mers of a given reference sequence set. We consider a read a hit with a reference sequence if the number of matching k-mers is greater than or equal to a given threshold t. For this k-mer model, we calculate the threshold as described in the Appendix A.3.

Since using all k-mers of the reference sequences can result in huge index sizes, k-mer selection approaches gained much attraction during the last decade, with minimizers being the most popular down-sampling approach for metagenomic classification of short reads (Piro et al., 2020; Wood et al., 2019). However, Edgar (2021) recently showed that syncmers are more sensitive for selecting conserved k-mers in biological sequences, and Dutta et al. (2022) could also improve long-read mapping by using open canonical syncmers instead of minimizers. Therefore, we implemented open canonical syncmers as a down-sampling strategy for large sets of k-mers to decrease the size of the HIXF index. In our implementation, open syncmers are sampled based on three parameters (k, s, t) where k, s, and t are positive integers and $s \leq k$. The method then compares the $k - s + 1$ consecutive s-mers within a k-mer and selects the k-mer as a syncmer if the smallest s-mer occurs at position $t \in [0, k - s + 1]$ within the k-mer. The smallest s-mer is defined by the hash value computed for each s-mer. We use the canonical representation of syncmers, meaning that the lexicographically smallest syncmer out of its forward and reverse-complement sequence is always selected.

Analogous to the k-mer-based approach, we need to determine a threshold for a read's minimum number of matching syncmers to consider it a hit with a reference sequence. In contrast to k-mers, there is no theoretical derivation for a $(1 - \alpha)$ confidence interval of the number of erroneous syncmers. Thus, we decided to derive the threshold empirically by simulating error-prone nanopore reads from a random sample of 1,000 bacterial reference genomes from the GTDB (Parks et al., 2022). We used the Rust implementation of the read simulator `Badread` (Wick, 2019) to simulate nanopore reads with five different read lengths between 1,000 and 5,000 bp and repeated the simulations for 20 different read accuracy rates (80%, 81%, 82%, ..., 99%). Next, we build separate

HIXF index files of the 1,000 genomes, one for each even k-mer value between 16 and 30. We only allow for even-numbered values for the k-mer size because we use canonical syncmers, with 16 being the practically smallest value to distinguish k-mers from different reference sequences and 30 the maximum value that allows finding k-mer matches between error-prone reads and a reference sequence. Finally, we separately queried the simulated reads of different read accuracies against the created HIXF index files and calculated the minimum fraction of found syncmers for each read and every combination of read accuracy and k-mer size. This minimum matching ratio for the different read accuracy and k-mer size combinations is used by `Taxor` to calculate the threshold for the minimum number of syncmer matches between a given read and the reference sequences stored in the HIXF index.

4.2.4 Taxonomic profiling

The existence of homologous regions of genome sequences across multiple microbial species can lead to high false positive rates if taxonomic profiling methods exclusively rely on sequence similarity information. However, setting a low sequence similarity threshold is essential for detecting all species in a sample if the sequencing accuracy hardly reaches values of 98%. Therefore, we apply a three-step filtering approach of potential hits between reads and reference sequences before refining the results using an expectation maximization (EM) algorithm that re-assigns reads to references based on the number of k-mer matches and taxonomic abundances of matched references.
Before the filtering, reads are assigned to matched reference genomes if the number of matching k-mers exceeds a certain threshold. Thus, a read can be assigned to many reference genomes in the index. We perform the first filter step on the single read level, determining the best matching reference genome based on the maximum number of k-mer matches (max_{kmatch}) with the given read. All reference assignments to that read where the number of matching k-mers is smaller than $0.8 \times max_{kmatch}$ are considered spurious matches and removed from the results. The second filtering step creates a list of all reference genomes with at least one uniquely mapped read (a read assigned to exactly one reference genome). Consequently, we remove all read-to-reference assignments in the results where the reference genome has no uniquely mapped read. Finally, we apply the two-stage taxonomy assignment algorithm used in `MegaPath` (Leung et al., 2020) to reduce suspicious matched references. In short, the algorithm identifies reference genomes with less than 5% of their matches being uniquely matched reads. If such a reference genome S also shares a certain amount (e.g., 95%) of matches with another reference genome T, all matches of reads to S are re-assigned to T.

After filtering, we estimate the relative abundances of all matched references and re-assign reads using a standard EM algorithm. This approach iteratively maximizes the likelihood that a given read r comes from a reference genome g, which also maximizes the likelihood of the relative taxonomic abundances in the whole read set. Let G be the set of genomes in the database and let π_g be the probability that a sequencing read in the sample emanates from database genome $g \in G$. We define the likelihood of the mapped read set R as

$$\mathcal{L}(R, \pi, G) = \prod_{r \in R} \sum_{g \in G} \pi_g \times \mathbb{P}(r|g) \tag{4.2}$$

where $\mathbb{P}(r|g)$ is the probability of read r coming from reference genome g. We use $mc(r,g)$, the number of k-mer (or syncmer) matches between read $r \in R$ and genome $g \in G$, and define $\mathbb{P}(r|g)$ as

$$\mathbb{P}(r|g) = \frac{\frac{mc(r,g)}{|mers(r)|}}{\sum_{j \in G} \frac{mc(r,j)}{|mers(r)|}} \tag{4.3}$$

with $|mers(r)|$ being defined as the number of k-mers (or syncmers) computed from r. After initialization of the taxonomic compositions with $\pi = \frac{1}{|G|}$ for all $g \in G$, we calculate in each iteration step an updated read assignment for each $r \in R$ by

$$hit(r,g) = \arg\max_{g \in G}(\pi_g \times \mathbb{P}(r|g)). \tag{4.4}$$

Based on the reassignment of reads, we update the taxonomic compositions π in each iteration step by accumulating the read lengths of all reads mapping to a certain genome and normalizing this value by the genome length.

$$\pi_g = \frac{\frac{\sum_{hit(r,g)} len(r)}{len(g)}}{\sum_{g \in G}\left(\frac{\sum_{hit(r,g)} len(r)}{len(g)}\right)} \tag{4.5}$$

The nominator in Equation 4.5 can be interpreted as the depth of coverage on genome g in the sample under investigation. We divide the coverage of g by the sum of all genome coverages to get the relative taxonomic abundance of g in the sample. The single steps of the EM algorithm are repeated until convergence of the likelihood $\mathcal{L}(R, \pi, G)$ or after a predefined number of iteration steps (default 10).

4.3 Results

We compare Taxor to five state-of-the-art taxonomic profiling tools, `Centrifuge` (Kim et al., 2016), `MetaMaps` (Dilthey et al., 2019), `Kraken2` (Wood et al., 2019), `KMCP` (Shen et al., 2023) and `Ganon` (Piro et al., 2020). `Centrifuge` is the tool underlying ONT's "What's in my pot" (WIMP) application for real-time species identification (Juul et al., 2015). While `Centrifuge` was initially developed to analyze short-read metagenomic samples, `MetaMaps` specifically addresses the task of strain-level metagenomic assignment of long reads. We decided to use both tools and `Kraken2` with default parameter configurations because this results in the best recall and precision in our experiments on simulated data.
Finally, we evaluated `KMCP` and `Ganon`, both utilizing different Bloom Filter approaches to store sets of selected k-mers from reference genomes for short-read classification. For `Ganon`'s *classify* module, we have to set parameters "–rel-cutoff 0.12" and "-rel-filter 0.9" to account for the higher error rates in nanopore reads. For the same reason, we also set parameters "–min-query-cov 0.12", "–min-hic-ureads-qcov 0.2" and "–min-chunks-fraction 0.2" when running `KMCP`'s *search* and *profile* subcommands. For the evaluation of `Taxor`, we only set the parameter "–error-rate 0.15", trying to assign all nanopore reads with error rates lower than or equal to 15%. The specific commands to run the six tools are provided in Appendix A.3.

4.3.1 Reference databases

Most taxonomic profilers offer prebuilt reference databases but also allow building custom reference databases. Since the choice of the reference database directly affects the outcome of taxonomic profiling, a fair comparison between tools also requires using the same database. This ensures that observed differences in the single-read assignments are attributed solely to the profiling methods. Thus, we downloaded all complete genome sequences and chromosomes of archaea, bacteria, viruses, and fungi from the NCBI RefSeq database (Release 217) (O'Leary et al., 2016) using `genome_updater 0.5.2` (https://github.com/pirovc/genome_updater). We used only one reference genome per species, resulting in 21,003 genomes used to build custom databases for each tool. This custom database comprises 11,579 viral genomes, 8,938 bacterial genomes, 403 archaea genomes, and 83 fungi genomes.
We build customized index data structures based on the described reference database for all tools included in our evaluation. For `Centrifuge`, we created the reference index using the default parameters, providing only the taxonomic information from the NCBI

taxonomy and reference sequences in fasta format. For MetaMaps, we first created a custom database from our downloaded taxonomy using the provided scripts and following the instructions on https://github.com/DiltheyLab/MetaMaps. Then we created the MetaMaps index from the custom database using default parameters. Since the other tools all use pseudo-alignment utilizing k-mer-based approaches, we build indexes using a k-mer size of 22, which is a good compromise between high specificity on species-level identification and high sensitivity for error-prone nanopore read classification. For all four approaches, we decided to use k-mer selection schemes that downsample the used k-mer sets to roughly 10% of all reference k-mers to reduce memory usage and index size significantly. Specifically, we used ungapped k-mers of size 22 and a window size of 32 for minimizer-based indexes in Kraken2 and Ganon. For KMCP and Taxor, we used a k-mer size of 22 and a syncmer size of 12. We further set the false positive rate for the Bloom Filter-based approaches, namely Ganon and KMCP, to 0.3% to reflect the same inherent false positive rate of Taxor's XOR filter approach. The specific commands and instructions to build the reference indexes of all tools are listed in the Appendix A.3.

4.3.2 Evaluation datasets

We carried out four experiments to evaluate Taxor, covering multiple metagenome composition scenarios from simulated and real data as well as sample contamination with eukaryotic DNA. The simulated dataset consists of 100 randomly chosen reference genomes included in the reference database and comprises 1,124,128 reads. For read simulation, we used pbsim2 (Ono et al., 2021) with parameters "–accuracy-mean 0.95", "–length-min 1000" and "–hmm_model R103.model". We chose a mean accuracy of 95% because the latest basecallers for ONT data have shown to reach such accuracies (Ferguson et al., 2022). We further decided to simulate reads with a minimum read length of 1,000 bp because shorter nanopore reads are commonly misclassified, and thus, MetaMaps and Taxor do not classify those reads.
For evaluation on real data, we obtained two ONT datasets for the ZymoBIOMICS D6300 microbial community standard (Nicholls et al., 2019) and one PacBio HiFi dataset for the ZymoBIOMICS Gut Microbiome Standard D6331. The Zymo D6300 standard consists of ten evenly abundant species, including 8 bacteria at 12% sequence abundance and two yeasts at 2% sequence abundance. The first ONT dataset comes from a continually updated online resource (https://lomanlab.github.io/mockcommunity/r10.html). We downloaded the R10.3 chemistry data release (February 2020), which was produced from two flowcells on an ONT GridION, resulting in 1.16 million reads (4.64

Giga base pairs (Gbp) data). The second ONT dataset was obtained from the European Nucleotide Archive (PRJEB43406: ERR5396170, released March 2021) and represents the 'Q20 chemistry' release for the Zymo D6300 standard (described at github.com/Kirk3gaard/2020-05-20_ZymoMock_Q20EA). It was generated using a PromethION, resulting in 5.4 million reads (17.95 Gbp data).
The PacBio HiFi dataset for the ZymoBIOMICS Gut Microbiome Standard D6331 (PRJNA680590: SRX9569057, released November 2020) contains 17 species (including 14 bacteria, one archaeon, and two yeasts) in staggered abundances. Five species occur at 14% sequence abundance, four at 6%, four at 1.5%, and one per 0.1%, 0.01%, 0.001%, and 0.0001% abundance level. There are five strains of *E. coli* in this community (each at 2.8% sequence abundance), which we treat here as one species at 14% sequence abundance. The PacBio Zymo D6331 dataset was generated using the Sequel II System and contains 1.9 million HiFi reads with a median length of 8.1 kbp, for a total of 17.99 Gbp of data.
To generate a read-level truth set, we use `minimap2` (Li, 2018) to map the reads against the reference genomes provided by ZymoBiomics. All reads that cannot be mapped with `minimap2` are excluded, and the primary alignment for each read determines the assumed true placement. This results in two ONT Zymo D6330 evaluation datasets referred as "ZymoR10.3" and "ZymoQ20", and one PacBio HiFi Zymo D6331 evaluation dataset referred as "HiFi_D6331".
As a negative control, we use `pbsim2` with the same parameters described above to simulate long-read sequencing data from two eukaryotic genomes not present in the reference database. Specifically, we simulate 685,303 reads from the *Aedes aegypti* (yellow fever mosquito) genome (GCF_002204515.2) and 142,677 reads from the *Toxoplasma gondii* ME49 genome (GCF_000006565.2). The two read sets are analyzed independently with `Taxor` and the other tools.

4.3.3 Evaluation metrics

We evaluated the performance of all six tools using several criteria. We assessed read utilization and classification metrics at the species, genus, and family levels and relative abundance estimates at the species level. First, we evaluated read utilization for each profiling method by calculating the total percent of reads assigned to specific taxonomic levels. We performed this for the following ranks: class, order, family, genus, and species. Here, we expected methods like `Kraken2` and `Ganon` that use an assignment to the lowest common ancestor to display read assignments across multiple taxonomic levels, while methods like `Taxor` only report the species level.

We calculated several metrics to evaluate read classification performance based on the number of true positives, false positives, and false negatives. In this context, we define a true positive as a correct taxon assignment of a read. We define a false positive as an incorrect taxon assignment on the read level. We further define a false negative as the failure to detect a taxon for a specific read. The formulas for precision, recall, and F-scores are as follows:

$$\textbf{\textit{Precision}} = \frac{\text{true positives}}{(\text{true positives} + \text{false positives})}$$

$$\textbf{\textit{Recall}} = \frac{\text{true positives}}{(\text{true positives} + \text{false negatives})}$$

$$\boldsymbol{F_1} = \frac{(2 \times \text{precision} \times \text{recall})}{(\text{precision} + \text{recall})}$$

$$\boldsymbol{F_{0.5}} = \frac{((1 + 0.5^2) \times \text{precision} \times \text{recall})}{((0.5^2 \times \text{precision}) + \text{recall})}$$

The values for the above metrics each range from 0 to 1. For precision, a score of 1 indicates that all reads with a taxon assignment have been assigned to the correct taxon, whereas lower scores indicate a higher number of wrong taxon assignments. For recall, a score of 1 indicates that all reads were assigned to the correct taxon, whereas a lower score indicates that no taxon could be assigned for some reads. The F-scores provide a useful way to summarize the information from precision and recall. The F_1 score is the harmonic mean of precision and recall (both measures are weighted equally), whereas the $F_{0.5}$ score gives more weight to precision (placing more importance on minimizing false positives). A value of 1 for either F-score indicates perfect precision and recall.
For accurate measurement of the read classification metrics for the real mock datasets, we had to control for species synonymies in the taxonomy of the used reference database. To avoid a negative impact on the metrics, we used the sum of cumulative counts for the species and all synonyms as the read count for the taxon. In particular, this included five species in Zymo D6300 (*Limosilactobacillus fermentum* = *Lactobacillus fermentum*; *Bacillus subtilis* = *Bacillus spizizenii*; *Escherichia coli* = *Escherichia sp. TC-EC600-tetX4*; *Listeria monocytogenes* = *Listeria sp. LM90SB2*; *Staphylococcus aureus* = *Staphylococcus sp. T93*) and two species in Zymo D6331 (*Limosilactobacillus fermentum* = *Lactobacillus fermentum*; *Escherichia coli* = *Escherichia sp. TM-G17TGC*), where we treated the 5 strains of *E. coli* contained in this community as one species.
We calculated detection metrics for each dataset. To understand the performance of each method across all datasets, we took an average of precision, recall, F_1, and $F_{0.5}$ at the

species level for the simulated and real datasets.
Finally, we attempted to obtain relative abundances for each method, acknowledging differences in reporting abundances as described by Z. Sun et al. (2021). In particular, there are clear differences in intended outputs among methods. For example, profiling methods (`Ganon` & `KMCP`) provide taxonomic abundances, whereas classifiers (`Centrifuge`, `Kraken2` & `MetaMaps`) provide sequence abundances. Since `Taxor` reports both abundance measurements, we did not transform the reported values of the tools but compared `Taxor`'s output directly to the reported abundance types of the respective tools. We calculated an L1 distance between observed and theoretical abundances for each method as described by Portik et al. (2022). The theoretical abundances were obtained from the manufacturer's specifications based on genomic DNA (sequence abundance) and genome copy (taxonomic abundance). We calculated the L1 distance by summing the absolute error between the theoretical and empirical estimate per species across the three communities. In this calculation, we included the false positives lumped in the "Other" category and compared them against a theoretical abundance of zero for this category.

4.3.4 Read utilization performance

Across the four metagenomic evaluation datasets, `Taxor` shows a high total read assignment between 89% (ZymoR103) and 97% (Hifi_D6331). Since we did not implement a lowest common ancestor algorithm in our new tool, all reads have been directly assigned to the species level. Compared to the other five tools in our benchmarking (see Figure 4.7), `Taxor` generally classifies fewer reads than `Centrifuge`, `Kraken2`, and `MetaMaps` but more than `KMCP` and `Ganon`. However, for the real datasets, `Taxor` assigns more reads than `Kraken2` and `Centrifuge` at the species level. Both tools assign a considerable amount of reads to the genus level, particularly for the ONT datasets. `KMCP` shows by far the lowest total read assignments of all tools on the ONT datasets. For the PacBio HiFi_D6331 dataset, all tools show a comparable high percentage of read assignments on the species level (between 92% for `Kraken2` and 97% for `Taxor` and `Ganon`), except for `KMCP`, which assigns 67% to species level, 12% to genus level and 16% to family level. In general, we observe that the read utilization of some tools is highly dependent on the dataset and especially the underlying sequencing technology, with ONT data having significantly higher error rates than PacBio data.

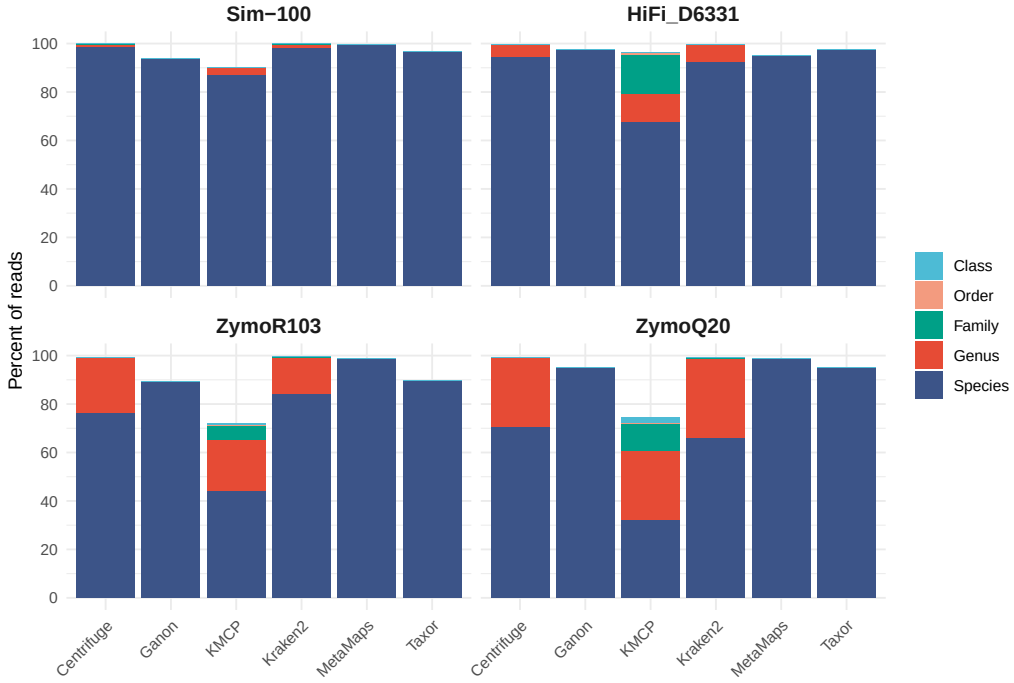

Figure 4.7: Read utilization of simulated and real metagenomic datasets. The stacked bar plots show the total percent of reads that were assigned to different taxonomic ranks, highlighted in different. `Taxor` generally classifies fewer reads than `Centrifuge`, `Kraken2`, and `MetaMaps` but more than `KMCP` and `Ganon`. For the real datasets, `Taxor` assigns more reads than `Kraken2` and `Centrifuge` at the species level.

4.3.5 Classification performance on simulated data

We first evaluate the read classification performance of `Taxor` in a simulation experiment, which represents a medium-complexity metagenomic analysis scenario with 100 randomly chosen species from the used reference database. We report the resulting performance metrics on the species level for all six evaluated tools in Table 4.1. On this dataset, `Taxor` read assignments achieve a recall of 0.96, a precision of 0.99, and F-scores of 0.98 and 0.99. `Taxor` outperforms `KMCP` and `Ganon` in terms of recall by 3-9% while having slightly lower recall than `Centrifuge`, `MetaMaps`, and `Kraken2` (2-3%). All tools show a high read classification precision in this experiment, ranging from 0.98 to 0.99, which means the tools report very few false read assignments. Four of the six tools can also correctly classify more than 95% of the simulated reads. Only `Ganon` and `KMCP` show a lower recall, failing to classify 7-13% of the simulated reads at the species level.

In a second simulation experiment, we assess the effect of contamination with eukaryotic host DNA from larger genomes on read classification. Therefore, we simulated reads from two eukaryotic genomes (*Aedes aegypti* and *Toxoplasma gondii*), neither of which is present in the reference database. `Taxor` has a low false-positive rate for both read sets and correctly leaves the large majority of reads unclassified ($\geq$ 99.99%) on all taxonomic levels. We also note low false positive rates for `KMCP` (0% for both datasets) and `Ganon` (0.24% for *Aedes aegypti* and 0.05% for *Toxoplasma gondii*). The three tools slightly outperform `MetaMaps`, having misclassification rates between 1.71% (*Aedes*) and 2.87% (*Toxoplasma*). In contrast, `Kraken2` and `Centrifuge` show high false-positive rates, with `Kraken2` reporting only 5.67% (*Aedes*) and 8.84% (*Toxoplasma*) of reads as unclassified, and `Centrifuge` reporting 22.90% (*Aedes*) and 26.74% (*Toxoplasma*) of reads as unclassified. Detailed results for these experiments are provided in Appendix A.3.

Tool	Precision	Recall	F_1-Score	$F_{0.5}$-Score
Centrifuge	0.99	0.98	0.98	0.99
MetaMaps	0.99	0.99	0.99	0.99
Kraken2	0.98	0.98	0.98	0.98
KMCP	0.99	0.87	0.93	0.97
Ganon	0.99	0.93	0.96	0.98
Taxor	0.99	0.96	0.98	0.99

Table 4.1: Species-level read classification performance on simulated data. Precision, recall, and F-scores for the species-level analysis of simulated Data. All compared tools show high precision and recall above 0.95, except for `Ganon` and `KMCP`, which have a lower recall of 0.93 and 0.87, respectively.

4.3.6 Classification performance on real data

Since the simulated dataset represents only a medium-complex metagenomic sample, we further evaluate `Taxor` on real data using three sets of metagenomic sequencing data, two ONT datasets for the ZymoBIOMICS D6300 community standard, and one PacBio HiFi dataset for the ZymoBIOMICS Gut Microbiome Standard D6331. The read classification performance evaluation results are shown in Figure 4.8 and Appendix A.3. Consistent with observations on simulated data, `Taxor` shows a high precision on the species level of 0.97 and 0.94 on the ZymoR10.3 and HiFi_D6331 data sets. For both datasets, The precision increases to 0.98 on the genus level and 0.99 on the family level.

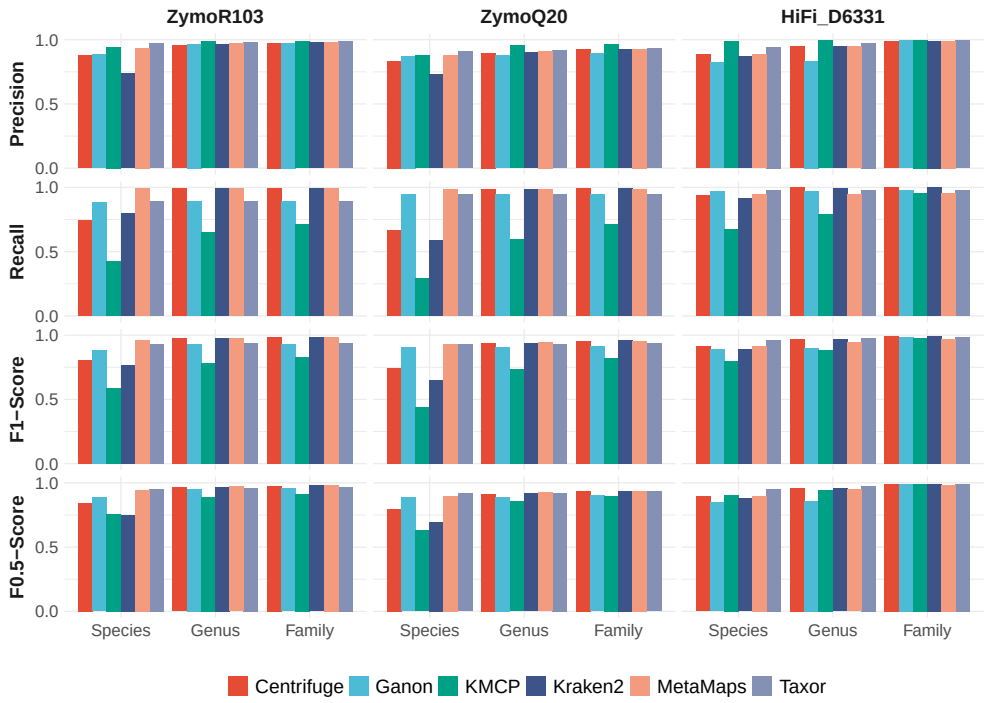

Figure 4.8: Read classification on species, genus, and family level for all three real datasets. Precision, recall, F_1-Score, and $F_{0.5}$-Score are shown for all six compared tools on three taxonomic levels. `Taxor` and `KMCP` show the highest precisions across all three datasets and all levels. `Taxor`'s recall is comparable to `MetaMaps` and `Ganon` on the species level, whereas `KMCP` consistently has the lowest recall of all tools. `Centrifuge` and `Kraken2` have low species-level recall for the two ONT datasets while performing well on the HiFi dataset. `Taxor` and `MetaMaps`, in general, have the highest F-scores, with `Taxor` outperforming `MetaMaps` on the $F_{0.5}$-Score.

For the second ONT dataset (ZymoQ20), `Taxor`'s species-level precision is slightly lower at 0.91 and increases to 0.92 (genus) and 0.93 (family) on higher taxonomic levels. `Taxor` further shows a high recall between 0.95 and 0.97 on all taxonomic levels for the ZymoQ20 and HiFi_D6331 datasets. Only for the ZymoR10.3 data set, `Taxor`'s recall has a value of 0.89 on all taxonomic levels. Our new tool has a F_1-Score between 0.93 and 0.96 and a $F_{0.5}$-Score between 0.92 and 0.95 on the species level across all three datasets.

Consistent patterns emerge when comparing `Taxor` to the other metagenomics read classification tools. Across all taxonomic levels and all three datasets, `Taxor` and `KMCP` show the highest precision. On the species level, `Taxor` shows the highest precision on both ONT data sets (0.97 on ZymoR10.3 and 0.91 ZymoQ20), outperforming

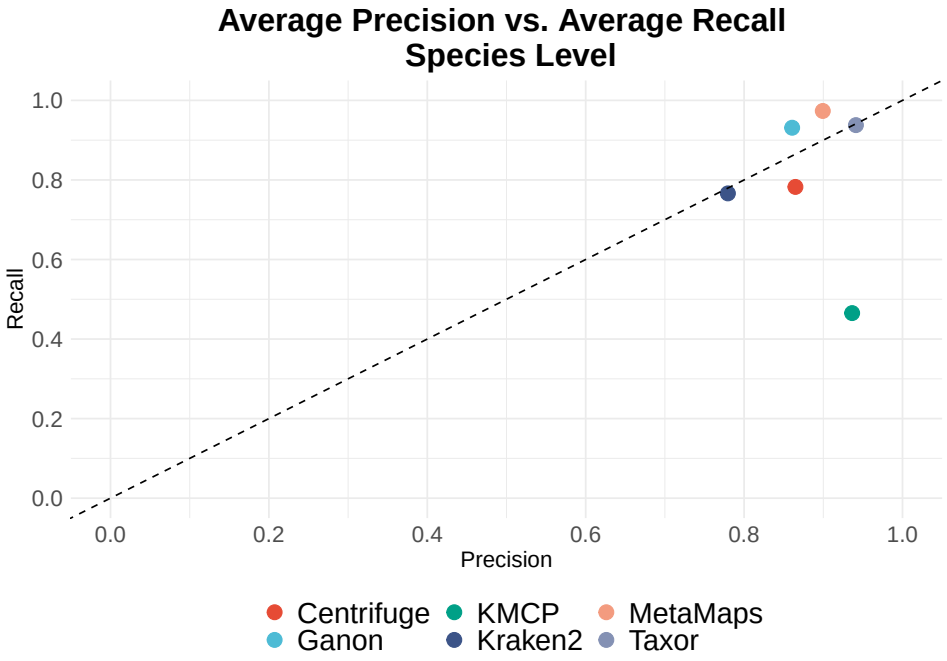

Figure 4.9: Average precision and recall on species level across the three real datasets. Showing average precision and recall of the six compared tools on the species level for the two ONT and the HiFi mock community datasets. `MetaMaps`, `Taxor`, and `Ganon` have the highest average recall, while `Taxor` and `KMCP` outperform the other tools in terms of precision. `KMCP` is the only tool with an average recall below 0.5 across the three datasets.

`MetaMaps` and `KMCP` by 1-3% and `Ganon` by 3-8%. On the HiFi_D6331 dataset, `KMCP` outperforms `Taxor` by 4% precision on the species level. However, `KMCP` shows the worst recall on all datasets across all taxonomic levels, while `Taxor`, on average, has the second-best recall on the species level (see Figure 4.9). We further recognize that `Centrifuge` and `Kraken2` have a low recall between 0.66 and 0.8 on the species level for both ONT datasets, which increases to 0.99 on the genus level. Since `Taxor` does not use a lowest-common-ancestor algorithm, its recall stays relatively constant across all taxonomic levels, which explains why `Centrifuge` and `Kraken2` outperform `Taxor` in terms of recall on higher taxonomic levels. The long-read classification method `MetaMaps` has the highest average recall on the species level across the three datasets.

When looking at F-scores, we see that `Taxor` consistently outperforms the other tools on the species level, except for the F_1-Score on the "ZymoR10.3" dataset, where `MetaMaps` performs best because of its high recall. On average, `Taxor` and `MetaMaps` have

the same species-level F_1-Score across all three datasets, but `Taxor` outperforms `MetaMaps` concerning the average $F_{0.5}$-Score when precision is prioritized over recall (Figure 4.10). On the genus and family level, all six tools are comparable except for `KMCP`, which suffers from low recall.
In summary, the two long-read methods, `Taxor` and `MetaMaps`, perform best on the species level across the three real datasets, making them the best choice for long-read metagenomics classification.

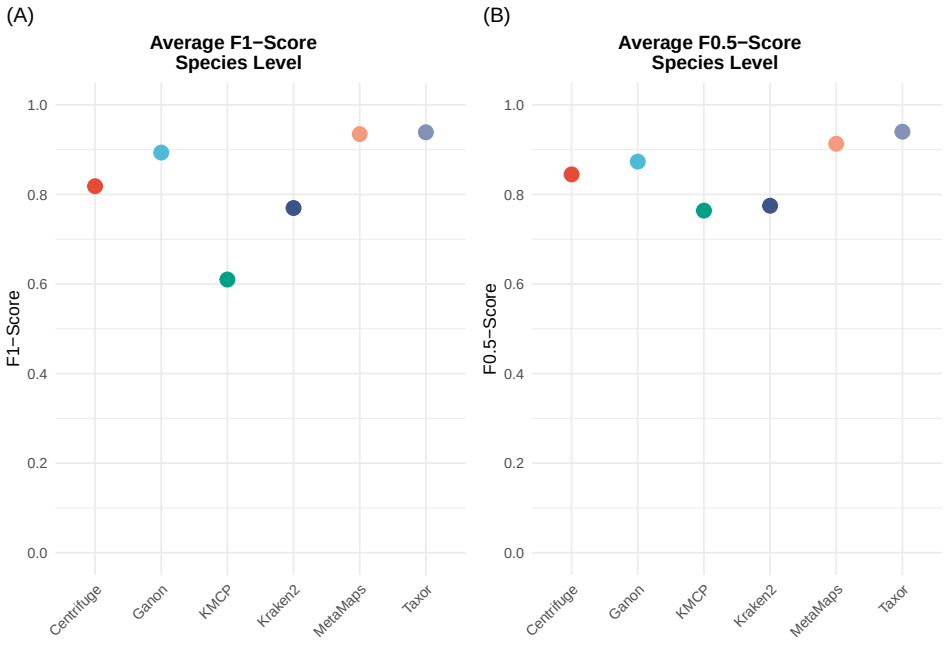

Figure 4.10: Average F-Scores on species level across the three real datasets. **(A)** Average species-level F_1-Score of the six tools across the three real mock communities. `Taxor` and `MetaMaps` show the highest scores, while `KMCP` has the lowest average F_1-Score. **(B)** Average species-level $F_{0.5}$-Score of the six tools across the three real mock communities. Our new tool `Taxor` outperforms the other methods when precision is prioritized over recall.

4.3.7 Relative abundance estimation on real data

All six evaluated tools report relative species abundance estimations after read classification and metagenomic profiling. In Figure 4.11, we compare the theoretical relative abundances of species in the mock communities against the relative species abundances

reported by the different tools. Taxor is the only tool that reports both types of abundance, sequence abundance and taxonomic abundance. When comparing the taxonomic abundance estimates of Taxor to those of Ganon and KMCP, we see that none of the three tools can accurately predict taxonomic abundances for all species in the three mock communities. However, Taxor has the lowest L1 distance of the three tools across all real datasets, demonstrating better relative species abundance estimates than Ganon and KMCP.
Since MetaMaps, Kraken2 and Centrifuge report relative sequence abundances of species instead of relative taxonomic abundances, we compare their profiling results to the theoretical sequence abundances of the mock communities. Consistent with the taxonomic abundance evaluation results, none of the investigated tools can accurately predict sequence abundance for the species comprising the three mock communities. As for the taxonomic abundance, Taxor outperforms the other tools by having the smallest L1 distance between theoretical and predicted species abundances across all datasets.

4.3.8 Computational requirements comparison

With the ever-increasing number of genomes in public databases comes the need for faster and more space-efficient methods to facilitate metagenomic read classification and profiling. Thus, we assessed the computational requirements of Taxor for indexing the reference database used in this study. Then, we also measured the required peak memory usage and runtime for querying the "ZymoR10.3" dataset against the built database index. We further compare Taxor's computational requirements against the other five tools used in this study. For all tools, we used the same reference database and built database indexes using the commands provided in the Appendix A.3. Specific commands for querying the built indexes are also provided in the Appendix A.3. We performed computational benchmarking on an AMD EPYC 7742 high-performance computing cluster (HPC) node using 30 threads. All times and peak memory usage were measured using the Linux "time" command with the parameter "-v".
Results of the computational benchmarking are provided in Table 4.2. KMCP is the fastest tool when building the index and has the lowest peak memory usage (5.84 GB). Ganon is the second fastest tool but has a high memory usage (53 GB) and a large index size on disk (36 GB). Taxor and Kraken2 both need approx. 80 minutes to build an index, but need only 50% (Kraken2) and 25% (Taxor) of Ganon's memory requirements. Although Taxor needs 2.5 times more memory than KMCP, the final HIXF index is 40% smaller than KMCP's index and 65% smaller than the Kraken2 index. Centrifuge's index is comparable to KMCP's index size, but Centrifuge

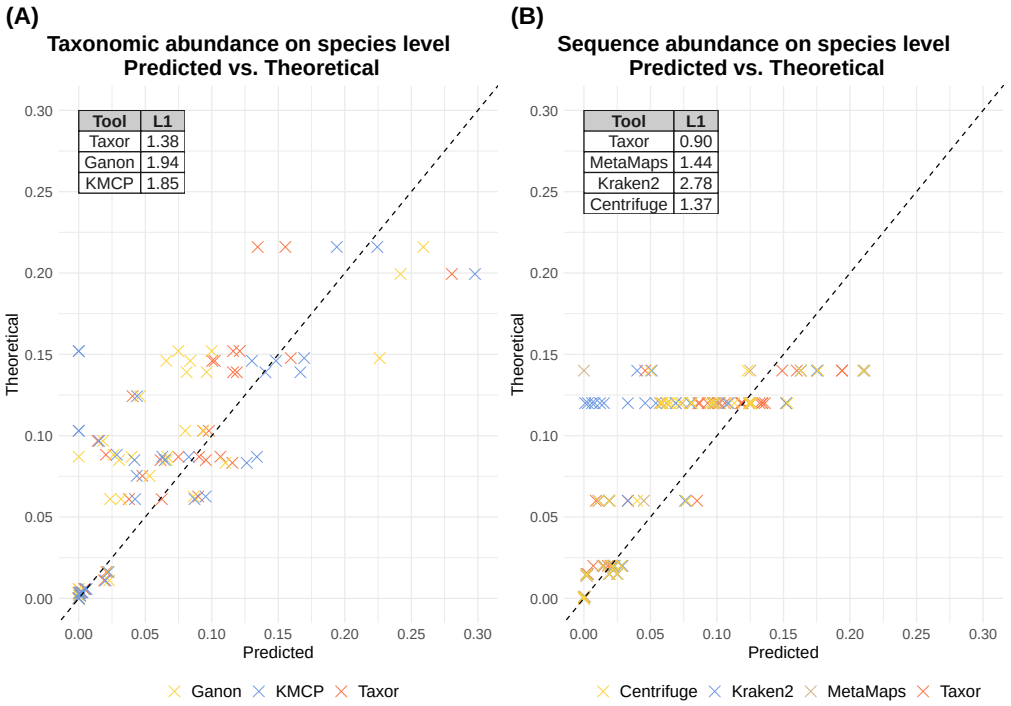

Figure 4.11: Comparison of theoretical abundances and predicted abundances. For each species in the three real mock communities, predicted abundances by the compared tools are plotted against the theoretical abundances. Crosses on the dashed line represent a perfect match between predicted and theoretical abundance. **(A)** Comparison of tools reporting taxonomic abundance. **(B)** Comparison of tools reporting sequence abundance. `Taxor` reports both abundance types and outperforms its competitors, having the smallest L1 distance between predicted and theoretical abundance.

needs 344 minutes and 385 GB RAM to build its index. `MetaMaps` is the second resource-hungry tool in our benchmarking, needing three times as much time and 25 times as much memory as `Taxor` to build its index. `MetaMaps` further has the largest index size of all tools, needing more than seven times the disk space as `Ganon` and 25 times as much as `Taxor`.

When querying the 426,213 ONT reads of the "ZymoR10.3" dataset against the respective database indexes, `Kraken2` performs fastest, needing less than 3 minutes to report read classification results. `Taxor` and `Ganon` are almost as fast as `Kraken2`, needing only 22 seconds (`Ganon`) and 45 seconds (`Taxor`) more query time than Kraken2. These three tools outperform the others, with a six (`Centrifuge`) to 12 (`KMCP`) times faster query time. `MetaMaps` is the most resource-hungry tool, needing 335 GB memory and more than 5 hours to query all reads against its index. Finally, we note that `Taxor`

Method	Build			Query	
	Time (mm:ss)	RAM (GiB)	Index Size (GiB)	Time (mm:ss)	RAM (GiB)
Centrifuge	344:50	385.30	18.4	20:14	18.4
MetaMaps	287:17	331.55	254.5	315:04	334.9
Kraken2	83:06	27.42	26.6	**2:50**	28.6
KMCP	**7:01**	**5.84**	16.1	36:49	17.2
Ganon	29:13	53.34	36.3	3:12	39.1
Taxor	81:39	13.38	**9.8**	3:35	**9.8**

Table 4.2: Results of Computational requirement benchmark. Reference indexes of a database consisting of 21,003 bacterial, viral, archaeal, and fungi genomes were built for all tools. We measured the elapsed time, peak memory usage, and index size for constructing the index. For the query benchmark, we measured the elapsed time and peak memory usage for classifying 426,213 ONT reads. Build and query times were measured using 30 threads on an HPC node.

outperforms all evaluated tools in terms of peak memory requirements when querying the index. Because of its small index size, `Taxor` can reduce the peak memory usage by approx. 50% compared to `Centrifuge` and `KMCP`, which both have much higher query times. Compared to `Kraken2` and `Ganon`, which have similar query times, `Taxor` can reduce the memory footprint by a factor of three (`Kraken2`) to four (`Ganon`). These results highlight the combination of fast and space-efficient read classification of `Taxor`, whereas other tools only provide fast querying or lower memory requirements.

4.4 Discussion

The increasing size of public reference genome databases such as NCBI RefSeq makes metagenomic profiling and read classification a computationally challenging task. In particular, reducing memory usage has become an objective of many studies during the last few years. However, state-of-the-art short- and long-read metagenomic classifiers still consume large amounts of memory, which can only be reduced by accepting a higher risk of false classifications. For Bloom Filter approaches like `Ganon` and `KMCP`, one can, for example, reduce the index size by accepting a higher false positive rate for the approximate membership queries of k-mers. However, higher false positive rates can lead to false classifications of viral reads, as proposed by Shen et al. (2023)
This issue motivated us to develop a new data structure that combines low memory requirements, a low false positive rate, and fast membership queries to facilitate precise metagenomic classification of long reads on large reference sequence databases.

Based on the work of Graf and Lemire (2020) and Mehringer et al. (2023), we created a hierarchical interleaved XOR filter (HIXF) data structure, which is implemented in the metagenomic classification tool called `Taxor`. Instead of relying solely on k-mers, `Taxor` can utilize syncmers as a k-mer selection scheme. Finally, our new tool comes with a profiling step, applying several filter strategies and an EM algorithm that refines the results of the initial read classification.

In this study, we present our work on `Taxor` and show benchmarking results of a comparison with five commonly used metagenomic profiling tools. `Taxor`'s read classification results are on par with state-of-the-art methods regarding recall while improving precision rates in almost every experiment on real data. Our new tool consistently shows the highest F_1- and $F_{0.5}$-scores on the species level for the three mock communities, indicating the robustness of the findings. We attribute this improvement to using syncmers instead of minimizers, applying several iterative filter steps, and our EM algorithm for read classification refinement. This assumption is based on the observation that `Taxor` and `KMCP` perform best regarding precision. Both use syncmers, similar filter steps, and a final EM algorithm for the profiling. They mainly differ in the underlying data structure for approximate membership querying and their implementation of the EM algorithm.

Contamination and environmental DNA are important aspects in many metagenomic studies, particularly if their genomes are not in the databases used by the metagenomic profilers. We have shown that `Taxor` is robust against these out-of-database genomes, minimizing the risk of false classifications of reads in such scenarios. Here, `Taxor` outperforms the mapping-based method `MetaMaps` and is on par with the Bloom Filter approaches of `Ganon` and `KMCP`. The false positive rate of 0.3%, which we used in this study for the Bloom Filter approaches, and our HIXF data structure seems to protect the three tools from falsely classifying reads from out-of-database genomes. Our observations suggest that removing host reads may not be necessary for these tools before the metagenomic classification of long reads in microbiome studies.

Correctly estimating the composition of microbial genomes in metagenomic samples is a main task in many microbiome studies, investigating the differential abundances of species between several gut or environmental samples. These differences can be attributed to environmental changes like global warming or anthropogenic pollution. Our benchmarking results show that none of the evaluated tools accurately estimates the abundance of all species in the real mock communities used for evaluation. Although `Taxor` also has problems estimating the species abundances correctly, its calculated sequence and taxonomic abundances are more consistent with the theoretical abundances of species in the communities. However, these results should be taken with a pinch of

salt as various factors, including different DNA extraction methods, can affect the final composition of DNA sequenced for metagenomic samples and potentially bias relative abundance estimates of the tools (Sui et al., 2020).
The small MinION sequencing devices invented by Oxford Nanopore Technologies (ONT) provide the possibility to sequence a sample at the place of its origin without the need to ship the sample to a laboratory. In such a point-of-care sequencing scenario, computing resources for metagenomic analysis are usually limited. Without a reliable internet connection to perform analysis in the cloud, the tools applied for metagenomic profiling must be as fast and memory efficient as possible while retaining high read classification accuracy. In this context, `Taxor` represents a significant improvement over state-of-the-art tools, requiring only 50% of the memory and disk space as the best competitor in our benchmarking. `Taxor` was also among the fastest tools regarding the query time while showing the highest average precision across the three real evaluation datasets. We expect `Taxor` to be a valuable tool for usage in real-time long-read metagenomic analysis pipelines like `minoTour` (R. Munro et al., 2022) or `WIMP` (Juul et al., 2015), both using `Centrifuge` for read classification.
Our new HIXF data structure significantly improves the interleaved Bloom filter concerning memory consumption and query time. It also reduces the memory requirements compared to the recently published hierarchical interleaved Bloom filter (Mehringer et al., 2023) when both use the same false positive rate. However, this comes at the cost of less flexibility and more time needed to build the index. The biggest drawback of the XOR filter is the ability to only index static sets of keys. That means all input data need to be known a priori, and the index cannot be updated after it has been built once. Since the XOR filter is the underlying data structure of our HIXF, this also applies to it. However, we argue that build times for the HIXF are still acceptable and that the lower memory footprint for querying reads compensates for that issue. We even envisage that tools currently relying on Bloom filters or interleaved Bloom filters can benefit from utilizing the HIXF as their index data structure, particularly in time-critical applications like nanopore adaptive sampling (Ulrich et al., 2022).
There are two important directions for the future development of `Taxor`. First, reducing the computational requirements by enhancing the underlying data structure is worth the effort because the number of genomes in public databases is constantly growing. One possibility would be using Binary Fuse or Ribbon filters instead of the XOR Filter in the hierarchical interleave data structure. Recent studies have shown that both filter types are practically smaller than XOR filters (Dillinger & Walzer, 2021; Graf & Lemire, 2022). Secondly, estimating the species abundances in the investigated metagenomic samples needs to be improved to reliably use our tool in microbiome studies relying on

differential abundance calculations. Here, enhancing the profiling would require further filtering and an improvement of our EM algorithm, which remains an open task for further research studies.

5 Summary and Conclusions

5.1 Summary

Long-read nanopore sequencing has changed the DNA sequencing landscape in many ways. On the one hand, nanopore sequencing produces the longest sequenced fragments across all available sequencing technologies on the market. These ultra-long reads are crucial for assembling complex repeat structures in the human genome, including long palindromes, tandem repeats, and segmental duplications (Jain et al., 2018; Miga et al., 2014; Skaletsky et al., 2003; Vollger et al., 2022). The long reads are also essential for identifying structural variations, which have been implicated in a wide range of genetic disorders (Cretu Stancu et al., 2017; Stankiewicz & Lupski, 2010). Compared to short-read sequencing, long nanopore reads also improve the read classification accuracy in metagenomics samples (Dilthey et al., 2019; Pearman et al., 2020) and allow the generation of near-finished bacterial genomes from pure cultures and metagenomes (Sereika et al., 2022). On the other hand, ONT's small MinION sequencer requires almost no capital investment, making DNA sequencing affordable for research groups from low- and middle-income countries (Pallerla et al., 2022; Pullen et al., 2019). The small device is not much bigger than a chocolate bar, which enables direct sequencing of samples at the place of their origin without shipping them to a sequencing facility (Runtuwene et al., 2019). In contrast to short-read sequencing technologies, where the complete sequence of a fragment is only known after the sequencing run has finished, nanopore sequencing offers the possibility for real-time analysis of the first sequenced fragments after some minutes of sequencing (Greninger et al., 2015). Although considerable efforts have been made to enable real-time analysis of intermediate short-read data during the sequencing run (Loka et al., 2019; Tausch et al., 2022; Tausch et al., 2018), nanopore sequencing outperforms current short-read technologies regarding the time to answer a biological or clinically relevant question (Euskirchen et al., 2017; Greninger et al., 2015; Quick et al., 2015). However, the exponential growth of reference sequence databases and the higher sequencing error rates make the real-time analysis of nanopore sequencing data challenging. In this thesis, I presented new methods for real-time metagenomic analysis of nanopore sequencing data with a specific application to targeted sequencing.

To this end, I first introduced ReadBouncer as a new tool for nanopore adaptive sampling in Chapter 2. Adaptive sampling is a method unique to nanopore sequencing, which enables selective sequencing of individual DNA molecules by rejecting uninteresting sequences from single nanopores. Therefore, real-time analysis of the sequence prefixes is critical to making fast rejection decisions. To accomplish this task, ReadBouncer implements a base-calling-and-mapping approach that pulls the raw signals from the sequencing device first. After live-base-calling of the signals, the read prefixes are classified with a pseudo-mapping approach based on Interleaved Bloom Filters (IBFs), and the classification decision is sent back to the nanopore sequencing device. In case a read was classified for rejection, the corresponding molecule is pulled out of the nanopore, releasing it for the following DNA molecule to be sequenced. We have shown that ReadBouncer improves read classification by combining IBFs with k-mer matching statistics. In particular, ReadBouncer shows a higher read classification sensitivity than other state-of-the-art classification tools for adaptive sampling while retaining a high specificity. Our tool also improves classification performance and memory usage compared to the other tools, which means that ReadBouncer can investigate more DNA molecules in the same amount of sequencing time. We developed our tool as an easy-to-install software application with a Graphical User Interface on Linux and Windows operating systems. Additionally, ReadBouncer supports fast CPU base-calling, providing even small sequencing facilities or in-field researchers that typically only have access to low-cost hardware the possibility to use the adaptive sampling feature of the MinION sequencer.
The application of nanopore adaptive sampling can reduce the time to answer a specific biological or clinical question. Additionally, it has been shown that adaptive sampling can enrich certain amplicons (Loose et al., 2016), human genes (Kovaka et al., 2021; Payne et al., 2021), and low-abundant species in metagenomic samples (Martin et al., 2022). These findings inspired us to investigate whether we could enrich low-abundant plasmids in bacterial isolate samples. Plasmids are mobile genetic elements (MGEs) that play an essential role in horizontal gene transfer and the spread of antimicrobial resistances (AMRs). Thus, they are an important subject for sequencing in clinical microbiology studies and outbreak investigations. However, they are often underrepresented in clinical samples and need to be enriched in the laboratory, which is expensive and time-consuming. In Chapter 3, I presented the results of a proof-of-concept study that showed the potential of adaptive sampling for the enrichment of low-abundant plasmid sequences by rejecting chromosomal sequences in bacterial isolate samples. Therefore, I used two adaptive sampling tools, namely ReadBouncer and ONT's MinKNOW sequencing control software, to investigate whether an enrichment can be reached independent

of the adaptive sampling tool used. Although different levels of plasmid enrichment were observed, the tools consistently enriched for low-abundant plasmid sequences. I demonstrated that the enrichment by yield, the most critical value for researchers, can reach up to 1.8x after 24 hours of sequencing on an ONT MinION sequencing device. In this context, I could also show that the difference between enrichment by yield, number of reads, and mean depth of coverage is negligible in all sequenced samples. Further investigations indicate that high-quality assemblies of plasmids are possible within two hours of sequencing with adaptive sampling and show even better results than plasmid assemblies without adaptive sampling. However, the study also highlights a potential issue with nanopore adaptive sampling if regions with high sequence identity are located on the chromosome and the plasmid. This can lead to false read rejections, which result in a depletion of the targeted plasmid sequences. In summary, the results reflect the benefit of adaptive sampling for the *in-silico* enrichment of low-abundant plasmids in bacterial isolate samples but also sound a note of caution if target and non-target sequences are similar.

In order to examine further cost-savings, I also inspected the possibility of using expired flow cells with fewer active sequencing pores for the *in-silico* enrichment via adaptive sampling in this study. Although the number of active sequencing pores decreases faster when using adaptive sampling, there was no negative impact on the enrichment of target sequences and the average quality of sequenced reads. This shows that flow cells with reduced active pores can be used in adaptive sampling experiments to increase the sustainability and cost-savings of research laboratories.

With the adaptive sampling tool presented in Chapter 2, it is possible to reject uninteresting sequences directly from an ongoing experiment. Although this method can remove host background reads from microbiome sequencing experiments, it acts like a binary classifier and does not perform real-time analysis of the target sequences. Existing taxonomic profilers struggle with the ever-increasing amount of reference genomes and need high-performance computing clusters to perform real-time analysis of metagenomics samples. To overcome this issue, I introduced `Taxor` as a fast and space-efficient taxonomic profiler for long reads in Chapter 4. Here, I described the hierarchical interleaved XOR filter (HIXF), a new data structure for approximate membership queries that I developed and implemented in `Taxor`. It combines low memory requirements, a low false positive rate, and fast membership queries to facilitate precise metagenomic classification of long reads on large reference sequence databases. Instead of relying solely on k-mers for the pseudo-mapping, `Taxor` utilizes open canonical syncmers as a k-mer selection scheme. For the final profiling step, I implemented several filter strategies and an EM algorithm to refine the initial read classification results.

To evaluate Taxor, Chapter 4 also includes benchmarking results of a comparison with five commonly used metagenomic profiling tools. Here, Taxor's read classification results are on par with state-of-the-art methods regarding recall while improving precision rates in almost every experiment on real data. Taxor consistently shows the highest F_1- and $F_{0.5}$-scores on the species level for the three mock communities, indicating the robustness of the findings. Finally, this new tool represents a significant improvement over state-of-the-art tools, requiring only 50% of the memory and disk space as the best competitor in the benchmarking. It was also among the fastest tools regarding query time while showing the highest average precision across the three real evaluation datasets.

In summary, the presented tools and methods provide real-time analysis solutions for nanopore metagenomics sequencing applications. These approaches range from host background removal over the enrichment of low-abundant sequences to real-time taxonomic profiling of microbiome samples and pathogen detection. Most notably, all described methods in this thesis rely on pseudo-mapping approaches using approximate filter data structures. They have been shown to provide high precision while providing fast and memory-efficient results, which makes pseudo-mapping approaches an excellent choice for the real-time classification of reads in metagenomics sequencing experiments.

5.2 Outlook

The research field of nanopore sequencing has seen rapid changes during the last five years. Although this thesis describes current advanced methods for real-time analysis of nanopore sequencing data, many new ideas have emerged during the work on these approaches. In particular, improvements in nanopore adaptive sampling methods are needed to convince more scientists to adopt this method for the *in-silico* enrichment of interesting biological sequences. One possibility is circumventing the computationally expensive base-calling step by directly classifying the nanopore signals. Recently, Firtina et al. (2023) developed a method to compute hash values for stretches of nanopore raw signals and compare these hash values to translated hash values of given reference sequences using ONT's k-mer models. Although their current method shows weaknesses when classifying human nanopore reads, improving and coupling their hashing method for raw nanopore signals with hierarchical filter structures could benefit ReadBouncer's adaptive sampling process. This would speed up the classification process and reduce the complexity of adaptive sampling tools by avoiding the need for fast GPU base callers, making adaptive sampling more attractive for applications in low-resource settings.
For ReadBouncer, I foresee many different future applications. Most obviously, I envision combining ReadBouncer and Taxor into an interactive application for

metagenomics sequencing. This tool would sequence a given sample for the first 15 to 30 minutes and list all organisms identified by `Taxor` in a Graphical User Interface (GUI). Then, the sequencing run would pause for some minutes while the user can decide which organisms should be depleted or enriched by just clicking a checkbox in the GUI. Finally, `ReadBouncer` would take the reference sequences of the chosen organisms and start the adaptive sampling process in the background after the sequencing run continues. This workflow could easily remove host background reads, but more importantly, it benefits the removal of known contaminants when combined with tools like `GRIMER` (Piro & Renard, 2023). Furthermore, this tool could also be used for the real-time identification of new pathogens by using `DeePaC` (Bartoszewicz et al., 2020) to screen all reads that were unclassified by `Taxor` for their pathogenic potential. When combined with Oxford Nanopore Technologies (ONT)'s MinION sequencer, this software suite could offer a real-time point-of-care test for known and novel pathogens in clinical applications, but also in farmed animals and plants as well as in wildlife research.

As another road for improvement of `ReadBouncer`, I also envision reference-free host removal with adaptive sampling in cases where the reference sequence of the host is incomplete or unknown. A typical use case is the sequencing of avian malaria parasites, which are numerous and common in wildlife but challenging to sequence because typically less than 2% of the bird's red blood cells are infected by the parasite (Bensch et al., 2016). Since bird erythrocytes are nucleated, the parasite proportion of total DNA in such a sequenced sample is less than 0.1%. However, the avian hosts and the malaria parasites significantly differ in GC content (Videvall, 2018). Using the GC content distribution of simulated reads from known parasite and avian genomes, it would be possible to create a statistical test that decides whether a sequenced read is more likely to originate from a parasite or a bird. Rejecting the avian reads with adaptive sampling could enrich the parasite reads and improve the *de novo* assembly of unknown malaria parasites. The approach can also be extended to using different k-mer profiles observed for hosts and their pathogens (Bohlin et al., 2008; Zielezinski et al., 2017).

Besides DNA sequencing, nanopore sequencing can also be used for direct RNA sequencing. This technique benefits whole transcriptome sequencing studies by avoiding the otherwise mandatory Polymerase Chain Reaction (PCR) amplification step with reverse transcriptase to translate RNA into cDNA. Many RNA-Seq samples suffer from high-abundant, uninteresting RNAs that occupy valuable sequencing capacity (Naarmann-de Vries, Eschenbach, et al., 2022). Although it has been shown that adaptive sampling can help identify formerly unknown low-abundant transcripts by depleting the high-abundant uninteresting RNAs, it can take too much time to determine the identity of the RNA and to decide whether to continue the sequencing of the given transcript (Naarmann-de Vries,

Gjerga, et al., 2022; Wan et al., 2023). Here, ReadBouncer could fasten the decision-making process with its precise pseudo-alignment approach. When combined with a classification based on the raw signals, one could also use methylations to distinguish between interesting and uninteresting RNA molecules. Additionally, deep-learning approaches like RISER (Sneddon et al., 2022) could improve adaptive sampling for nanopore direct RNA sequencing.

Future applications of adaptive sampling are also foreseeable for nanopore single-molecule protein sequencing. Brinkerhoff et al. (2021) showed in a proof-of-concept study how DNA-peptide conjugates can be measured with existing nanopore sequencing devices. This new technology is expected to revolutionize the proteomics research field (Alfaro et al., 2021). ReadBouncer's pseudo-alignment approach is easily adaptable to work with amino acid sequences and could help to enrich low-abundant peptides by depleting the overrepresented peptide sequences in a given sample. This can be particularly beneficial for developing clinical tests based on sequencing blood serum proteins. The less abundant proteins are primarily tissue-derived and show pathology-specific abundance differences, and they likely represent biomarker candidates for the early detection of diseases. However, the main challenge is that 90% of the protein content of serum consists of only ten proteins, with albumin comprising up to 50% of the whole protein content (Tirumalai et al., 2003). Although wet lab methods for depleting these high-abundant plasma proteins have been developed, a large fraction of them remains in the sample, occupying sequencing capacity that could better be used for the low-abundant biomarkers (Viode et al., 2023). In-silico depletion of these proteins with adaptive sampling will be a cost-efficient method to eliminate these uninteresting protein sequences. Similarly to nanopore DNA and RNA sequencing, epigenetic modifications of peptides can be measured with this direct protein sequencing approach. The post-translational modifications could then also be used to decide whether a currently sequenced peptide should be rejected from the pore or sequenced as usual.

One of the lessons learned during the development of ReadBouncer is the importance of having good simulation software to test developed tools. Only two possibilities existed to test adaptive sampling without performing a real sequencing run in the laboratory when ReadBouncer was implemented. The ReadUntil simulator provided with UNCALLED (Kovaka et al., 2021) needs two real runs: one control run and one UNCALLED run. However, this will only simulate the results of an adaptive sampling run but not an interactive behavior needed for testing the real-time communication between the sequencer and the adaptive sampling tool. The second option is to use ONT's MinKNOW, which can be configured to playback a prerecorded sequencing run. This simulator requires a bulk fast5 file of the recorded sequencing run, which is rarely

available. Another limitation of this approach is that rejected reads are not actually removed from a pore, but the original read is fragmented at the point the read would have been unblocked. Both approaches cannot mimic the interactive, adaptive sampling behavior of a real nanopore sequencer for different data sets. The recently published tool `Icarust` (R. J. Munro et al., 2023) addresses these challenges by simulating raw nanopore signals directly from given reference sequences. However, this approach is impractical for benchmarking adaptive sampling tools because it will not create the same reads for two different simulations of the same input. For such a simulator, it would be more intuitive to provide raw fast5 files of sequenced reads and use the raw signals stored with them.

Real-time whole metagenomic sequencing is a method with a high potential in clinical applications but also for molecular surveillance. Nanopore sequencing has successfully been used for outbreak surveillance of viral pathogens like Ebola (Quick et al., 2016), Zika (Quick et al., 2017) and Lassa (Kafetzopoulou et al., 2019). However, during the last few years, researchers have also started using portable nanopore sequencing for microbial surveillance of wastewater (Andersen et al., 2017; Fuhrmeister et al., 2021) and freshwater quality monitoring (Acharya et al., 2020; Urban et al., 2021). These approaches have shown to be an early warning system for outbreak detection of pathogens (Abdeldayem et al., 2022). Since `Taxor` is a resource-efficient taxonomic profiling tool, I expect it to benefit the metagenomic analysis of wastewater and freshwater samples when combined with a portable laboratory. `Taxor` will significantly contribute to molecular surveillance systems when integrated into pipelines for fast pathogen detection or microbiome analysis. When coupled with ARG databases like CARD (Alcock et al., 2023), `Taxor` will also improve the real-time detection and outbreak surveillance of antibiotic-resistant bacteria.
Nanopore sequencing has not only been used to detect antibiotic-resistant bacteria but also to characterize the resistome in wastewater treatment plants (Wu et al., 2022). One of the main drivers for the dissemination of ARGs are mobile genetic elements like plasmids. In Chapter 3 of this thesis, I described how adaptive sampling could improve the characterization of bacterial plasmids by enriching the low-abundant plasmids. However, the current approach has two main disadvantages: either the bacterial chromosomes or the plasmid sequences need to be known upfront, and in a metagenomics sample, it is unclear from which bacterial species the plasmid originates. We can overcome the first issue by using `Taxor` to initially identify the bacterial species in the sample and then select their chromosomal sequences for depletion. However, this requires spending a lot of sequencing capacity for sequencing bacterial chromosomes to correctly

identify the species, which will consequently reduce the potential enrichment of the target plasmids. A better solution would be to develop a deep-learning-based plasmid classifier that directly uses the raw nanopore signals. This approach could provide a fast classification by skipping the otherwise mandatory base-calling step and would not require prior knowledge about the bacterial species that comprise the sequenced sample. To resolve the second issue, it is also inevitable to sequence a certain amount of the bacterial chromosomes to identify the bacterial species in the sample using `Taxor`. After they have been identified, one could analyze the methylation profiles of the different species and deplete the chromosomal sequences that will not add more information. Finally, the methylation profiles of the sequenced plasmids can be used to associate them with their bacterial host genomes as proposed by Beaulaurier et al. (2018). This is crucial in clinical diagnostics of antibiotic-resistant pathogens because detecting an ARG on a bacterial plasmid in a clinical sample does not necessarily mean a pathogen in that sample is resistant to antibiotic treatment. Only the association between a plasmid harboring the ARG and a pathogenic host will provide enough evidence for this pathogen to be resistant.

In Chapter 4, I describe how the utilization of XOR filters as an alternative to the widely adopted Bloom filter can reduce the memory and disk space requirements of indexed databases for metagenomic classification. Besides XOR filters, Graf and Lemire (2022) also developed binary fuse filters, which are even smaller and more compact than XOR filters. However, building these binary fuse filters often fails, and many different hash functions must be tested before the build process succeeds. This makes them inappropriate for interleaved filter structures where the same hash functions have to be used for each of the individual filters. However, they are a promising route for further improvements of pseudo-alignment tools used for metagenomics profiling.

Because of the increasing number of new bioinformatic software tools, benchmarking methods developed for the same application has become an important topic of many studies (Mangul et al., 2019). Often, these benchmarks are performed by the developers of a new tool to demonstrate performance improvements. However, this results in the authors reporting their method to perform best in an unreasonable number of cases (Norel et al., 2011). To reduce this bias, neutral benchmarking studies performed by independent groups or community challenges are precious for the research community (Weber et al., 2019). There has been a great effort by the metagenomics community to perform such neutral benchmarking studies (Marić et al., 2021; Portik et al., 2022; Simon et al., 2019) and community challenges as CAMI (Meyer et al., 2022; Sczyrba et al., 2017). However, these studies often do not include the latest tools and are rarely up-to-date. Thus, the metagenomics and microbiome research community needs a tool or repository where

developers can register their new tool, and automatic benchmarking against all other methods is performed using gold-standard data sets for different sequencing technologies. The authors of a new method can include the results of this independent benchmarking in their studies, which would also circumvent the self-reporting bias.

A Appendix

A.1 Precise and scalable nanopore adaptive sampling with ReadBouncer

DNA Sequencing

ZymoBIOMICS HMW DNA Standard was purchased from Zymo Research, from which 400 nanogram (ng) were transferred into a 1.5 milliliter (mL) DNA LoBind tube, and the volume was adjusted to 7.5 microliter (µL) with Nuclease-free water. The sample was mixed by flicking the tube, spun down briefly, and then transferred into a 0.2 mL PCR tube. 2.5 µL Fragmentation Mix was added, and the tube was mixed by flicking the tube and spun down. After incubating for 1 min at 30 °C and subsequently for 1 min at 80 °C, the tube was briefly put on ice. 1 µL RAP was added to the barcoded DNA.
After mixing by flicking the tube and spinning down, the sample was incubated for 5 min at room temperature. In a new DNA LoBind tube, 34 µL Sequencing Buffer (SQB), 25.5 µL Loading Beads (LB), 4.5 µL Nuclease-free water, and 11 µL DNA library were added. Meanwhile, for the priming of the flow cell FLO-MIN106 (R9.4 SpotON), 30 µL Flush Tether (FLT) was added directly to a tube of Flush Buffer (FB) and mixed by vortexing. 800 µL priming mix were loaded into the flow cell via priming port without introducing any air bubbles. After incubating for 5 min, the SpotON sample port was opened, and 200 µL priming mix were loaded into the flow cell via priming port. Finally, the prepared DNA library was mixed by pipetting up and down, and 75 µL of the sample volume was loaded into the flow cell via the SpotON sample port in a dropwise fashion. After closing the SpotON sample port and priming port and replacing the MinION lid, the sequencing experiment was started via `MinKNOW` software.

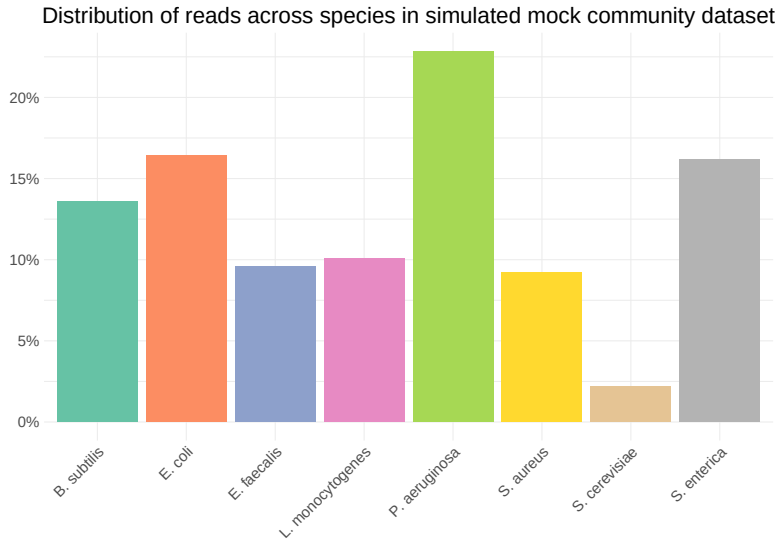

Figure A.1: Proportion of reads from each species in the simulated mock community dataset. Less than 2.5% of reads were simulated from *Saccharomyces cerevisiae*. We would aim to deplete bacterial reads in order to enrich for *Saccharomyces cerevisiae*.

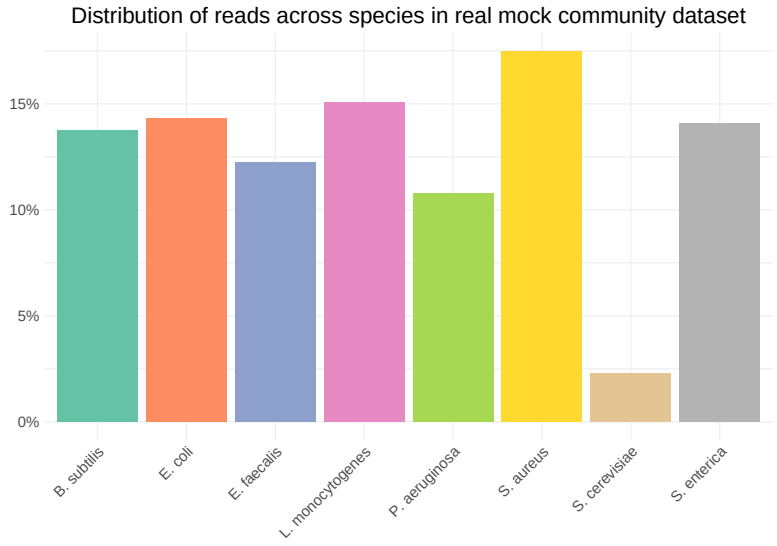

Figure A.2: Proportion of reads from each species in the real mock community dataset. Only 2.5% of reads sequenced from a real mock community originate from *Saccharomyces cerevisiae*. We would aim to deplete bacterial reads in order to enrich for *Saccharomyces cerevisiae*.

Table A.1: Comparing `ReadBouncer`, `SPUMONI`, and `minimap2` across various metrics on a simulated Zymo Mock Community consisting of seven bacterial species and *Saccharomyces cerevisiae*. We simulated 360 nucleotide long reads of varying levels of sequence accuracy for all eight organisms. All reads were mapped against the seven bacterial reference sequences to filter out only the bacterial reads. At the same time, we want to keep as much *Saccharomyces cerevisiae* reads, which corresponds to an enrichment of that organism in a real-world experiment.

Read Accuracy(%)	**80**			
Tool	**ReadBouncer**	**SPUMONI**	**minimap2** (k=15)	**minimap2** (k=13)
Accuracy	69.16	68.23	62.03	**70.52**
Precision	**100.00**	99.88	**100.00**	**100.00**
Recall	68.48	67.61	61.19	**69.87**
Specificity	99.98	96.32	**100.00**	**100.00**
F1-Score	**82.29**	80.64	75.92	82.26
MCC	**0.21**	0.20	0.18	**0.21**
Read Accuracy(%)	**85**			
Accuracy	**89.83**	89.15	86.19	89.24
Precision	**100.00**	99.90	**100.00**	**100.00**
Recall	**89.61**	89.00	85.89	89.00
Specificity	99.95	95.99	**100.00**	**100.00**
F1-Score	**94.51**	94.13	92.41	94.18
MCC	**0.40**	0.37	0.34	0.38
Read Accuracy(%)	**90**			
Accuracy	**96.74**	96.43	94.48	95.45
Precision	**100.00**	99.91	**100.00**	**100.00**
Recall	**96.67**	96.44	94.36	95.35
Specificity	99.94	96.01	**100.00**	**100.00**
F1-Score	**98.31**	98.14	97.10	97.62
MCC	**0.62**	0.59	0.52	0.55
Read Accuracy(%)	**95**			
Accuracy	**99.00**	98.84	97.19	97.53
Precision	**100.00**	99.91	**100.00**	**100.00**
Recall	**98.98**	98.90	97.13	97.47
Specificity	99.93	95.90	**100.00**	**100.00**
F1-Score	**99.49**	99.40	98.54	98.72
MCC	**0.82**	0.79	0.65	0.67
Read Accuracy(%)	**98**			
Accuracy	**99.28**	99.13	97.52	97.74
Precision	**100.00**	99.91	**100.00**	**100.00**
Recall	**99.27**	99.20	97.47	97.69
Specificity	99.88	95.98	**100.00**	**100.00**
F1-Score	**99.63**	99.56	98.72	98.83
MCC	**0.86**	0.83	0.67	0.69

A.2 Nanopore adaptive sampling effectively enriches bacterial plasmids

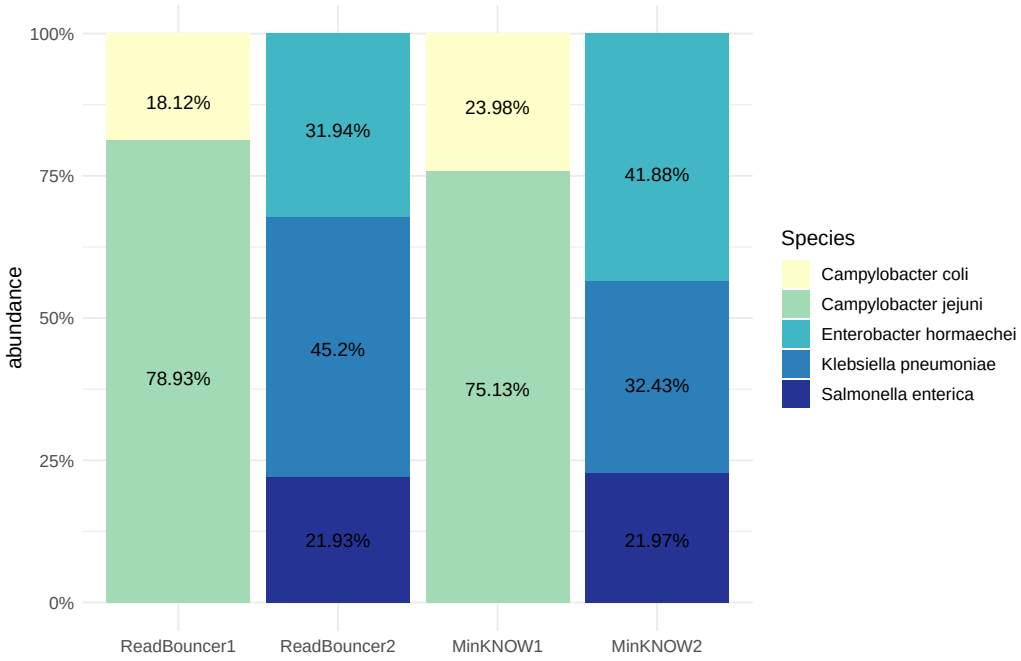

Figure A.3: Species abundances for each of the four MinION runs. Similar species abundances can be observed between ReadBouncer1 and MinKNOW1 with *Campylobacter jejuni* dominating in both runs. Abundances are also similar for ReadBouncer2 and MinKNOW2 except that *Enterobacter hormaechei* is most abundant species in MinKNOW2 and *Klebsiella pneumoniae* is most abundant in ReadBouncer2.

A.3 Fast and space-efficient taxonomic classification of long reads with hierarchical interleaved XOR filters

Threshold calculation for the k-mer based model

For our new HIXF data structure, we use membership queries for all k-mers of a given read against all k-mers of a given reference sequence set. We consider a read a hit with a reference sequence if the number of matching k-mers is greater than or equal to a given threshold t. For the k-mer model, we calculate the threshold using the expected sequencing error rate e and the definition of a $(1-\alpha)$ confidence interval of the number of erroneous k-mers as provided by Blanca et al. (2022) For a given read r with length $len(r)$ and k-mer length k, we denote the number of k-mers of a read r as $L = len(r) - k + 1$, and define q by $(1-(1-e)^k)$. Then, the expected number of erroneous k-mers can be calculated as follows:

$$E[N_{err}] = L \times q \tag{A.1}$$

Let $Var(N_{err})$ be the variance for the number of erroneous k-mers. We can then calculate the upper bound of the $(1-\alpha)$ confidence interval by

$$E[N_{err}] + z_\alpha \sqrt{Var(N_{err})} \tag{A.2}$$

With $z_\alpha = \phi^{-1}(1-\frac{\alpha}{2})$, where we denote ϕ^{-1} as the inverse of the cumulative distribution function of the standard Gaussian distribution. Based on the calculation of the confidence interval for the number of erroneous k-mers, we define our threshold for the minimum number of matching k-mers for read r as:

$$min[N_{match}] = L - (E[N_{err}] + z_\alpha \sqrt{Var(N_{err})}) \tag{A.3}$$

We classify a read as a match with a reference sequence if the number of matching k-mers is bigger or equal to

$$t = min[N_{match}] \tag{A.4}$$

Profiling methods commands

To facilitate reproducible results, we provide general commands or instructions to run each method.

genome_updater

We used genome_updater version 0.5.2 (https://github.com/pirovc/genome_updater) to download all complete genome sequences and chromosomes of archaea, bacteria, viruses, and fungi from the NCBI RefSeq database (Release 217) (O'Leary et al., 2016)

```
genome_updater.sh -d refseq -g archaea,bacteria,fungi,\
viral -l complete genome,chromosome, -f genomic.fna.gz \
-o refseq-abfv -t 12 -A species:1 -m -a -p

mkdir -p refseq-abfv/2023-03-15_12-56-12/taxdump

tar -xvzf refseq-abfv/2023-03-15_12-56-12/taxdump.tar.gz \
-C refseq-abfv/2023-03-15_12-56-12/taxdump
```

Centrifuge

We ran Centrifuge version 1.0.4 (Kim et al., 2016). First, we created a Fasta file of all downloaded reference genomes and a corresponding conversion table. Then we used the prepared input data and the downloaded NCBI taxonomy dump to build a Centrifuge database index

```
cut -f 1,6 refseq-abfv/2023-03-15_12-56-12/assembly\
_summary.txt > refseq-abfv/2023-03-15_12-56-12/taxid.map

centrifuge_conversion_table.py -t refseq-abfv/2023-03-15\
_12-56-12/taxid.map -s refseq-abfv/2023-03-15_12-56-12/\
files -o refseq-abfv/2023-03-15_12-56-12/centrifuge_data/\
conversion_table.tsv

centrifuge-build --conversion-table refseq-abfv/2023-03-15\
_12-56-12/centrifuge_data/conversion_table.tsv \
--taxonomy-tree refseq-abfv/2023-03-15_12-56-12/taxdump/\
nodes.dmp --name-table refseq-abfv/2023-03-15_12-56-12/\
taxdump/names.dmp -p 30 refseq-abfv/2023-03-15_12-56-12/\
files/all.fna refseq-abfv/2023-03-15_12-56-12/\
centrifuge_data/refseq-abfv
```

Finally, we queried the fastq file of one of the samples against the index and create a Kraken report file from the Centrifuge output.

```
centrifuge -q --min-hitlen 22 -k 20 -t -p 30 -x refseq-\
abfv/2023-03-15_12-56-12/centrifuge_data/refseq-abfv -U \
SAMPLE.fastq.gz -S refseq-abfv/2023-03-15_12-56-12/\
centrifuge_data/SAMPLE.centrifuge.search.txt --report-file
refseq-abfv/2023-03-15_12-56-12/centrifuge_data/\
SAMPLE.centrifuge.report.tsv

centrifuge-kreport -x refseq-abfv/2023-03-15_12-56-12/\
centrifuge_data/refseq-abfv --no-lca refseq-abfv/2023-\
03-15_12-56-12/centrifuge_data/SAMPLE.centrifuge.search\
.txt > refseq-abfv/2023-03-15_12-56-12/centrifuge_data/\
SAMPLE.centrifuge.kreport.txt
```

Kraken2

We ran Kraken2 version 2.1.2 (Wood et al., 2019). We first needed to prepare input data by adding the Kraken header information to the fasta file using the conversion table we also created for building the Centrifuge index.

```
kraken2-build --download-taxonomy --db refseq-abfv/2023-\
03-15_12-56-12/kraken2_data/refseq-abfv

add_kraken_header.py -t refseq-abfv/2023-03-15_12-56-12/\
centrifuge_data/conversion_table.tsv -f refseq-abfv/2023-\
03-15_12-56-12/files/all.fna -o refseq-abfv/2023-03-15\
_12-56-12/kraken2_data/all_seq.fna

kraken2-build -add-to-library refseq-abfv/2023-03-15_12-\
56-12/kraken2_data/all_seq.fna --db refseq-abfv/2023-03-\
15_12-56-12/kraken2_data/refseq-abfv-k32-m22 --no-masking
```

```
kraken2-build --build --kmer-len 32 --minimizer-len 22 \
--minimizer-spaces 0 --threads 30 --db refseq-abfv/2023-\
03-15_12-56-12/kraken2_data/refseq-abfv-k32-m22
```

After building the database index we can query each sample against it, resulting in a kraken report file and an output txt with binning information per read.

```
kraken2 --db refseq-abfv/2023-03-15_12-56-12/kraken2\
_data/refseq-abfv-k32-m22 --threads 30 --report refseq-\
abfv/2023-03-15_12-56-12/kraken2_data/SAMPLE.report \
--output refseq-abfv/2023-03-15_12-56-12/kraken2_data/\
SAMPLE.output.txt --gzip-compressed SAMPLE.fq.gz
```

KMCP

We used KMCP version 0.9.1 (Shen et al., 2023). First, we needed to prepare the input data for creating the database index, as described in the KMCP wiki (https://bioinf.shenwei.me/kmcp/database/#refseq-viral-or-fungi) using the tools `rush` and `brename`, provided by the author of KMCP.

```
cut -f 1,6 refseq-abfv/2023-03-15_12-56-12/assembly\
_summary.txt > refseq-abfv/2023-03-15_12-56-12/taxid.map

cut -f 1,8 refseq-abfv/2023-03-15_12-56-12/assembly \
_summary.txt > refseq-abfv/2023-03-15_12-56-12/name.map

mkdir -p refseq-abfv/2023-03-15_12-56-12/kmcp_data/ \
files.renamed

cd refseq-abfv/2023-03-15_12-56-12/kmcp_data/files.renamed

find refseq-abfv/2023-03-15_12-56-12/files \
-name "*.fna.gz" | rush 'ln -s {}'

cd ..
```

```
brename -R -p '^(\w{3}_\d{9}\.\d+).+' -r '$1.fna.gz' \
refseq-abfv/2023-03-15_12-56-12/kmcp_data/files.renamed
```

In the next step, we compute the syncmers and create the database index with a false positive rate of 0.3%.

```
kmcp compute -I refseq-abfv/2023-03-15_12-56-12/kmcp_data\
/files.renamed -O refseq-abfv/2023-03-15_12-56-12/kmcp\
_data/refseq-abfv-k22-s12 -S 12 -k 22 --seq-name-filter \
plasmid --split-number 10 --split-overlap 150 --log \
refseq-abfv-k22-s12.log -j 30 --force

kmcp index -I refseq-abfv/2023-03-15_12-56-12/kmcp_data\
/refseq-abfv-k22-s12/ -O refseq-abfv/2023-03-15_12-56-12/\
kmcp_data/refseq-abfv-k22-s12.kmcp -j 30 -f 0.003 -n 3 \
-x 100K --log refseq-abfv-k22-s12.kmcp.log --force
```

Finally, the sample file is queried against the index and the profiling refines read assignments and report taxonomic abundances.

```
kmcp search --db-dir refseq-abfv/2023-03-15_12-56-12/\
kmcp_data/refseq-abfv-k22-s12.kmcp --threads 30 -f 0.003 \
--min-query-cov 0.12 --out-file refseq-abfv/2023-03-15\
_12-56-12/kmcp_data/SAMPLE.tsv.gz SAMPLE.fq.gz

kmcp profile --taxid-map refseq-abfv/2023-03-15_12-56-12/\
kmcp_data/taxid.map --taxdump refseq-abfv/2023-03-15\
_12-56-12/taxdump/ --level species --min-query-cov 0.12 \
-m 3 refseq-abfv/2023-03-15_12-56-12/kmcp_data/SAMPLE\
.tsv.gz --min-hic-ureads-qcov 0.2 --min-chunks-fraction \
0.2 --out-prefix refseq-abfv/2023-03-15_12-56-12/kmcp\
_data/SAMPLE.kmcp.profile --cami-report refseq-abfv/2023-\
03-15_12-56-12/kmcp_data/SAMPLE.cami.profile --sample-id \
SAMPLE_NAME --binning-result refseq-abfv/2023-03-15_12-\
56-12/kmcp_data/SAMPLE.binning.gz
```

Ganon

We used Ganon version 1.8.0 (Piro et al., 2020). Here, no further preprocessing step is needed to create the custom database index. We used the same minimizer and k-mer lengths as for Kraken2 and created the index with a false positive rate of 0.3%. When classifying the SAMPLE reads, Ganon reports read assignments and taxonomic abundances in CAMI report format.

```
ganon build-custom --db-prefix refseq-abfv/2023-03-15_12-\
56-12/ganon_data/refseq-abfv --input refseq-abfv/2023-03-\
15_12-56-12/files/ --level species --ncbi-file-info \
refseq-abfv/2023-03-15_12-56-12/assembly_summary.txt \
--threads 30 --max-fp 0.003 --kmer-size 22 --window-size \
32 --hash-functions 3 --hibf

ganon classify --db-prefix refseq-abfv/2023-03-15_12-56-\
12/ganon_data/refseq-abfv -s SAMPLE.fq.gz -o refseq-abfv/\
2023-03-15_12-56-12/ganon_data/SAMPLE.search --threads \
30 -a --output-all -c 0.12 -e 0.9
```

MetaMaps

We used MetaMaps version 0.1 (Dilthey et al., 2019). To create the custom database index, we followed the steps described at https://github.com/DiltheyLab/MetaMaps#databases.

```
mkdir refseq-abfv/2023-03-15_12-56-12/metamaps_data/\
download

perl downloadRefSeq.pl --seqencesOutDirectory refseq-\
abfv/2023-03-15_12-56-12/metamaps_data/download/refseq \
--taxonomyOutDirectory refseq-abfv/2023-03-15_12-56-12/\
metamaps_data/download/taxonomy -targetBranches archaea,\
bacteria,fungi,viral
```

Then, we needed to modify the following line in script `annotateRefSeqSequencesWithUniqueTaxonIDs.pl` from

```
next unless($assembly_level eq 'Complete Genome');
```

to

```
next unless($assembly_level eq 'Complete Genome' \
|| ($assembly_level eq 'Chromosome');
```

and execute the script using the taxonomy downloaded by `genome_updater` and build the database used for indexing.

```
perl annotateRefSeqSequencesWithUniqueTaxonIDs.pl \
--refSeqDirectory refseq-abfv/2023-03-15_12-56-12/\
metamaps_data/download/refseq --taxonomyInDirectory \
refseq-abfv/2023-03-15_12-56-12/taxdump/ \
--taxonomyOutDirectory refseq-abfv/2023-03-15_12-56-12/\
metamaps_data/download/taxonomy_uniqueIDs

mkdir -p refseq-abfv/2023-03-15_12-56-12/taxdump/ \
--taxonomyOutDirectory refseq-abfv/2023-03-15_12-56-12/\
metamaps_data/databases

perl buildDB.pl --DB refseq-abfv/2023-03-15_12-56-12/\
metamaps_data/databases/refseq-abfv --FASTAs refseq-abfv/\
2023-03-15_12-56-12/metamaps_data/download/refseq \
--taxonomy refseq-abfv/2023-03-15_12-56-12/metamaps_data/\
download/taxonomy_uniqueIDs
```

Then we finally index the created database with the following command.

```
metamaps index -r refseq-abfv/2023-03-15_12-56-12/\
metamaps_data/databases/refseq-abfv/DB.fa -t 30 -i \
refseq-abfv/2023-03-15_12-56-12/metamaps_data/\
refseq-abfv-k16
```

For querying sample reads against the created index, we used the following commands.

```
metamaps mapAgainstIndex --all -q SAMPLE.fq.gz -i \
```

```
refseq-abfv/2023-03-15_12-56-12/metamaps_data/databases/\
refseq-abfv-k16 -o refseq-abfv/2023-03-15_12-56-12/\
metamaps_data/SAMPLE.map.txt -t 30

metamaps classify --DB refseq-abfv/2023-03-15_12-56-12/\
metamaps_data/databases/refseq-abfv-k16 -t 30 --mappings \
refseq-abfv/2023-03-15_12-56-12/metamaps_data/\
SAMPLE.map.txt
```

Taxor

We used Taxor version 0.1.0. For preprocessing, we have to create a tab-separated file containing important taxonomic information using the tool `taxonkit`(https://github.com/shenwei356/taxonkit).

```
cut -f 1,7,20 refseq-abfv/2023-03-15_12-56-12/assembly\
_summary.txt | taxonkit lineage -i 2 -r -n -L --data-dir \
refseq-abfv/2023-03-15_12-56-12/taxdump | taxonkit \
reformat -I 2 -P -t --data-dir refseq-abfv/2023-03-15\
_12-56-12/taxdump | cut -f 1,2,3,4,6,7 > refseq-abfv/\
2023-03-15_12-56-12/taxor_data/refseq_accessions\
_taxonomy.csv
```

Then we build the Taxor index using the tab-separated file and the sequence files downloaded with `genome_updater`.

```
taxor build --input-file refseq-abfv/2023-03-15_12-56-12/\
taxor_data/refseq_accessions_taxonomy.csv --input-\
sequence_dir refseq-abfv/2023-03-15_12-56-12/files --\
output-filename refseq-abfv/2023-03-15_12-56-12/\
taxor_data/refseq-abfv-k22-s12.hixf --threads 30 \
--kmer-size 22 --syncmer-size 12 --use-syncmer
```

Finally, we query the sample fastq file against the index, allowing a sequencing error rate of 15%. The query result file is used as input for taxonomic profiling, which has three output files containing taxonomic abundances and sequence abundances in CAMI report format as well as a binning file with final read to reference assignments.

```
taxor search --index-file refseq-abfv/2023-03-15_12-56-\
12/taxor_data/refseq-abfv-k22-s12.hixf --query-file \
SAMPLE.fq.gz --output-file refseq-abfv/2023-03-15_12-\
56-12/taxor_data/SAMPLE.search.txt --error-rate 0.15 \
--threads 30

taxor profile --search-file refseq-abfv/2023-03-15\
_12-56-12/taxor_data/SAMPLE.search.txt --cami-report-file \
refseq-abfv/2023-03-15_12-56-12/taxor_data/SAMPLE.report \
--seq-abundance-file refseq-abfv/2023-03-15_12-56-12/\
taxor_data/SAMPLE.abundance --binning-file refseq-abfv/\
2023-03-15_12-56-12/taxor_data/SAMPLE.binning \
--sample-id SAMPLE
```

Evaluation Results

Table A.2: Comparison of Taxor and five different taxonomic profilers classifying simulated host background reads. We simulated nanopore reads from *Aedes aegypti* and *Toxoplasma gondii* with 95% read accuracy. Then, we taxonomically classified these reads using the indexed database consisting of archaea, bacteria, fungi and viruses and list the percentage of falsely assigned reads by each of the tools.

Experiment	**Mosquito**	**Toxoplasma**
Reads	685,303	142,677
Tool	**% unclassified**	
Centrifuge	22.90	26.74
Kraken2	5.67	8.84
MetaMaps	98.29	97.13
Ganon	99.83	99.95
KMCP	100.00	100.00
Taxor	99.99	99.99

Table A.3: Comparison of Taxor and five different taxonomic profilers across various metrics on a simulated community consisting of 100 bacterial species we simulated 1,124,128 nanopore reads from 100 randomly selected bacterial species from the GTDB database with a mean read accuracy of 95%. After the taxonomic classification of the reads by all six tools, we calculated precision, recall, and F-scores on five different taxonomic ranks.

Tool	**Centrifuge**	**Kraken2**	**MetaMaps**	**Ganon**	**KMCP**	**Taxor**
Rank	**Species**					
Precision	0.9919	0.9869	0.9996	0.9971	0.9994	0.9999
Recall	0.9853	0.9814	0.9954	0.9382	0.8711	0.9669
F1-Score	0.9886	0.9841	0.9975	0.9668	0.9308	0.9831
F0.5-Score	0.9906	0.9858	0.9987	0.9848	0.9708	0.9931
Rank	**Genus**					
Precision	0.9985	0.9978	0.9997	0.9986	0.9999	1.0000
Recall	0.9960	0.9972	0.9954	0.9383	0.9004	0.9669
F1-Score	0.9972	0.9975	0.9975	0.9675	0.9476	0.9831
F0.5-Score	0.9980	0.9976	0.9988	0.9859	0.9783	0.9932
Rank	**Family**					
Precision	0.9988	0.9988	0.9998	0.9991	0.9999	1.0000
Recall	0.9987	0.9989	0.9954	0.9383	0.9014	0.9669
F1-Score	0.9987	0.9989	0.9976	0.9678	0.9481	0.9832
F0.5-Score	0.9988	0.9989	0.9989	0.9863	0.9785	0.9932
Rank	**Order**					
Precision	0.9992	0.9989	0.9998	0.9995	0.9999	1.0000
Recall	0.9989	0.9992	0.9954	0.9384	0.9020	0.9669
F1-Score	0.9990	0.9991	0.9976	0.9680	0.9485	0.9832
F0.5-Score	0.9991	0.9990	0.9989	0.9867	0.9787	0.9932
Rank	**Class**					
Precision	0.9993	0.9992	0.9998	0.9998	0.9999	1.0000
Recall	0.9991	0.9995	0.9954	0.9384	0.9026	0.9669
F1-Score	0.9992	0.9993	0.9976	0.9681	0.9488	0.9832
F0.5-Score	0.9993	0.9992	0.9989	0.9869	0.9788	0.9932

Table A.4: Comparison of Taxor and five different taxonomic profilers across various metrics classifying real nanopore reads from a ZymoBiomics mock community (ZymoR10.3)
We created a ground truth data set by mapping 426,213 nanopore reads against the reference sequences provided by the manufacturer. After the taxonomic classification of the reads by all six tools, we calculated precision, recall, and F-scores on five different taxonomic ranks.

Tool	**Centrifuge**	**Kraken2**	**MetaMaps**	**Ganon**	**KMCP**	**Taxor**
Rank	**Species**					
Precision	0.8761	0.7392	0.9361	0.8866	0.9431	0.9697
Recall	0.7407	0.7994	0.9893	0.8827	0.4277	0.8932
F1-Score	0.8027	0.7681	0.9619	0.8846	0.5885	0.9299
F0.5-Score	0.8452	0.7505	0.9462	0.8858	0.7599	0.9533
Rank	**Genus**					
Precision	0.9563	0.9623	0.9683	0.9676	0.9847	0.9779
Recall	0.9907	0.9913	0.9896	0.8914	0.6478	0.8940
F1-Score	0.9732	0.9766	0.9789	0.9280	0.7815	0.9341
F0.5-Score	0.9630	0.9679	0.9725	0.9514	0.8919	0.9599
Rank	**Family**					
Precision	0.9744	0.9768	0.9797	0.9745	0.9870	0.9851
Recall	0.9929	0.9941	0.9897	0.8921	0.7090	0.8947
F1-Score	0.9835	0.9854	0.9847	0.9315	0.8252	0.9378
F0.5-Score	0.9780	0.9802	0.9817	0.9568	0.9152	0.9656
Rank	**Order**					
Precision	0.9773	0.9798	0.9824	0.9790	0.9878	0.9875
Recall	0.9930	0.9943	0.9898	0.8925	0.7109	0.8950
F1-Score	0.9851	0.9870	0.9861	0.9338	0.8268	0.9390
F0.5-Score	0.9804	0.9826	0.9839	0.9604	0.9164	0.9675
Rank	**Class**					
Precision	0.9838	0.9849	0.9872	0.9888	0.9941	0.9915
Recall	0.9934	0.9946	0.9898	0.8935	0.7174	0.8953
F1-Score	0.9886	0.9897	0.9885	0.9387	0.8334	0.9410
F0.5-Score	0.9857	0.9868	0.9877	0.9681	0.9229	0.9706

Table A.5: Comparison of Taxor and five different taxonomic profilers across various metrics classifying real nanopore reads from a ZymoBiomics mock community (ZymoQ20) We created a ground truth data set by mapping 4,179,959 nanopore reads against the reference sequences provided by the manufacturer. After the taxonomic classification of the reads by all six tools, we calculated precision, recall, and F-scores on five different taxonomic ranks.

Tool	**Centrifuge**	**Kraken2**	**MetaMaps**	**Ganon**	**KMCP**	**Taxor**
Rank	**Species**					
Precision	0.8348	0.7295	0.8761	0.8717	0.8790	0.9095
Recall	0.6665	0.5855	0.9851	0.9436	0.2934	0.9472
F1-Score	0.7412	0.6496	0.9274	0.9063	0.4399	0.9280
F0.5-Score	0.7947	0.6953	0.8959	0.8852	0.6282	0.9168
Rank	**Genus**					
Precision	0.8976	0.9002	0.9120	0.8781	0.9569	0.9169
Recall	0.9877	0.9871	0.9856	0.9440	0.5984	0.9476
F1-Score	0.9405	0.9416	0.9474	0.9099	0.7363	0.9320
F0.5-Score	0.9143	0.9163	0.9258	0.8905	0.8545	0.9229
Rank	**Family**					
Precision	0.9215	0.9257	0.9280	0.8911	0.9645	0.9295
Recall	0.9908	0.9930	0.9859	0.9448	0.7101	0.9483
F1-Score	0.9549	0.9582	0.9561	0.9171	0.8180	0.9388
F0.5-Score	0.9345	0.9384	0.9390	0.9013	0.9000	0.9332
Rank	**Order**					
Precision	0.9288	0.9332	0.9348	0.9055	0.9661	0.9362
Recall	0.9911	0.9932	0.9860	0.9456	0.7158	0.9487
F1-Score	0.9589	0.9623	0.9597	0.9251	0.8224	0.9424
F0.5-Score	0.9406	0.9446	0.9446	0.9133	0.9030	0.9387
Rank	**Class**					
Precision	0.9444	0.9472	0.9477	0.9385	0.9781	0.9496
Recall	0.9918	0.9940	0.9862	0.9474	0.7405	0.9493
F1-Score	0.9675	0.9700	0.9666	0.9429	0.8429	0.9495
F0.5-Score	0.9535	0.9562	0.9551	0.9402	0.9191	0.9496

Table A.6: Comparison of Taxor and five different taxonomic profilers across various metrics classifying real PacBio HiFi reads from a ZymoBiomics mock community (HiFi_D6331)
We created a ground truth data set by mapping 1,978,408 reads against the reference sequences provided by the manufacturer. After the taxonomic classification of the reads by all six tools, we calculated precision, recall, and F-scores on five different taxonomic ranks.

Tool	**Centrifuge**	**Kraken2**	**MetaMaps**	**Ganon**	**KMCP**	**Taxor**
Rank	**Species**					
Precision	0.8835	0.8699	0.8859	0.8205	0.9872	0.9444
Recall	0.9402	0.9137	0.9459	0.9692	0.6746	0.9729
F1-Score	0.9109	0.8913	0.9149	0.8887	0.8015	0.9585
F0.5-Score	0.8943	0.8783	0.8973	0.8465	0.9035	0.9500
Rank	**Genus**					
Precision	0.9483	0.9476	0.9481	0.8341	0.9946	0.9752
Recall	0.9971	0.9957	0.9492	0.9697	0.7937	0.9737
F1-Score	0.9721	0.9711	0.9487	0.8968	0.8829	0.9745
F0.5-Score	0.9577	0.9569	0.9484	0.8581	0.9467	0.9749
Rank	**Family**					
Precision	0.9865	0.9894	0.9893	0.9949	0.9986	0.9942
Recall	0.9978	0.9971	0.9512	0.9745	0.9529	0.9742
F1-Score	0.9921	0.9933	0.9699	0.9846	0.9752	0.9841
F0.5-Score	0.9887	0.9910	0.9814	0.9907	0.9891	0.9902
Rank	**Order**					
Precision	0.9928	0.9951	0.9966	0.9992	0.9998	0.9985
Recall	0.9981	0.9977	0.9516	0.9746	0.9615	0.9743
F1-Score	0.9954	0.9964	0.9736	0.9867	0.9803	0.9862
F0.5-Score	0.9938	0.9956	0.9873	0.9941	0.9919	0.9935
Rank	**Class**					
Precision	0.9929	0.9948	0.9966	0.9992	0.9998	0.9985
Recall	0.9982	0.9980	0.9516	0.9746	0.9615	0.9743
F1-Score	0.9955	0.9964	0.9736	0.9867	0.9803	0.9862
F0.5-Score	0.9939	0.9955	0.9873	0.9942	0.9919	0.9935

Bibliography

Abdeldayem, O. M., Dabbish, A. M., Habashy, M. M., Mostafa, M. K., Elhefnawy, M., Amin, L., Al-Sakkari, E. G., Ragab, A., & Rene, E. R. (2022). Viral outbreaks detection and surveillance using wastewater-based epidemiology, viral air sampling, and machine learning techniques: A comprehensive review and outlook. *Science of The Total Environment*, *803*, 149834 (cit. on p. 103).

Acharya, K., Blackburn, A., Mohammed, J., Haile, A. T., Hiruy, A. M., & Werner, D. (2020). Metagenomic water quality monitoring with a portable laboratory. *Water research*, *184*, 116112 (cit. on p. 103).

Ahmed, O., Rossi, M., Kovaka, S., Schatz, M. C., Gagie, T., Boucher, C., & Langmead, B. (2021). Pan-genomic matching statistics for targeted nanopore sequencing. *iScience*, 102696 (cit. on p. 32).

Aislabie, J., Deslippe, J. R., & Dymond, J. (2013). Soil microbes and their contribution to soil services. *Ecosystem services in New Zealand–conditions and trends. Manaaki Whenua Press, Lincoln, New Zealand*, *1*(12), 143–161 (cit. on p. 3).

Akeson, M., Branton, D., Kasianowicz, J., Brandin, E., & Deamer, D. (1999). Microsecond time-scale discrimination among polycytidylic acid, polyadenylic acid, and polyuridylic acid as homopolymers or as segments within single rna molecules. *Biophysical journal*, *77*, 3227–33. https://doi.org/10.1016/S0006-3495(99)77153-5 (cit. on p. 9)

Alcock, B. P., Huynh, W., Chalil, R., Smith, K. W., Raphenya, A. R., Wlodarski, M. A., Edalatmand, A., Petkau, A., Syed, S. A., Tsang, K. K., et al. (2023). Card 2023: Expanded curation, support for machine learning, and resistome prediction at the comprehensive antibiotic resistance database. *Nucleic acids research*, *51*(D1), D690–D699 (cit. on p. 103).

Alfaro, J. A., Bohländer, P., Dai, M., Filius, M., Howard, C. J., Van Kooten, X. F., Ohayon, S., Pomorski, A., Schmid, S., Aksimentiev, A., et al. (2021). The emerging landscape of single-molecule protein sequencing technologies. *Nature methods*, *18*(6), 604–617 (cit. on p. 102).

Almeida, P. S., Baquero, C., Preguiça, N., & Hutchison, D. (2007). Scalable bloom filters. *Information Processing Letters*, *101*(6), 255–261 (cit. on p. 16).

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of molecular biology*, *215*(3), 403–410 (cit. on pp. 12, 13).

Andersen, M. H., Kirkegaard, R. H., & Albertsen, M. (2017). Rapid microbial surveillance using nanopore dna sequencing. *Nanopore Community Meeting* (cit. on p. 103).

Andrusch, A., Dabrowski, P. W., Klenner, J., Tausch, S. H., Kohl, C., Osman, A. A., Renard, B. Y., & Nitsche, A. (2018). Paipline: Pathogen identification in metagenomic and clinical next generation sequencing samples. *Bioinformatics*, *34*(17), i715–i721 (cit. on pp. 6, 68).

Arslan, S., Garcia, F. J., Guo, M., Kellinger, M. W., Kruglyak, S., LeVieux, J. A., Mah, A. H., Wang, H., Zhao, J., Zhou, C., et al. (2023). Sequencing by avidity enables high accuracy with low reagent consumption. *Nature Biotechnology*, 1–7 (cit. on p. 8).

Baker, D. N., & Langmead, B. (2019). Dashing: Fast and accurate genomic distances with hyperloglog. *Genome biology*, *20*, 1–12 (cit. on p. 75).

Bao, Y., Wadden, J., Erb-Downward, J. R., Ranjan, P., Zhou, W., McDonald, T. L., Mills, R. E., Boyle, A. P., Dickson, R. P., Blaauw, D., et al. (2021). Squigglenet: Real-time, direct classification of nanopore signals. *Genome biology*, *22*, 1–16 (cit. on pp. 10, 46, 47).

Bartoszewicz, J. M., Seidel, A., Rentzsch, R., & Renard, B. Y. (2020). Deepac: Predicting pathogenic potential of novel dna with reverse-complement neural networks. *Bioinformatics*, *36*(1), 81–89 (cit. on p. 101).

Beaulaurier, J., Zhu, S., Deikus, G., Mogno, I., Zhang, X.-S., Davis-Richardson, A., Canepa, R., Triplett, E. W., Faith, J. J., Sebra, R., et al. (2018). Metagenomic binning and association of plasmids with bacterial host genomes using dna methylation. *Nature biotechnology*, *36*(1), 61–69 (cit. on p. 104).

Bensch, S., Canbäck, B., DeBarry, J. D., Johansson, T., Hellgren, O., Kissinger, J. C., Palinauskas, V., Videvall, E., & Valkiūnas, G. (2016). The genome of haemoproteus tartakovskyi and its relationship to human malaria parasites. *Genome Biology and Evolution*, *8*(5), 1361–1373 (cit. on p. 101).

Bentley, D. R., Balasubramanian, S., Swerdlow, H. P., Smith, G. P., Milton, J., Brown, C. G., Hall, K. P., Evers, D. J., Barnes, C. L., Bignell, H. R., et al. (2008). Accurate whole human genome sequencing using reversible terminator chemistry. *nature*, *456*(7218), 53–59 (cit. on p. 7).

Bingmann, T., Bradley, P., Gauger, F., & Iqbal, Z. (2019). Cobs: A compact bit-sliced signature index. *String Processing and Information Retrieval: 26th International*

Symposium, SPIRE 2019, Segovia, Spain, October 7–9, 2019, Proceedings 26, 285–303 (cit. on pp. 16, 69).

Blanca, A., Harris, R. S., Koslicki, D., & Medvedev, P. (2022). The statistics of k-mers from a sequence undergoing a simple mutation process without spurious matches. *Journal of Computational Biology*, *29*(2), 155–168 (cit. on pp. 28, 111).

Blanco-Míguez, A., Beghini, F., Cumbo, F., McIver, L. J., Thompson, K. N., Zolfo, M., Manghi, P., Dubois, L., Huang, K. D., Thomas, A. M., et al. (2023). Extending and improving metagenomic taxonomic profiling with uncharacterized species using metaphlan 4. *Nature Biotechnology*, 1–12 (cit. on p. 12).

Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, *13*(7), 422–426 (cit. on pp. 16, 24, 69, 71).

Bohlin, J., Skjerve, E., & Ussery, D. W. (2008). Reliability and applications of statistical methods based on oligonucleotide frequencies in bacterial and archaeal genomes. *BMC genomics*, *9*(1), 1–18 (cit. on p. 101).

Botelho, F. C., Pagh, R., & Ziviani, N. (2007). Simple and space-efficient minimal perfect hash functions. *Algorithms and Data Structures: 10th International Workshop, WADS 2007, Halifax, Canada, August 15-17, 2007. Proceedings 10*, 139–150 (cit. on p. 71).

Boykin Okalebo, L., Sseruwagi, P., Alicai, T., Ateka, E., MOHAMMED, I., Stanton, J.-A., Kayuki, C., Mark, D., Fute, Erasto, Bachwenkizi, H., Muga, Mumo, N., Mwangi, Abidrabo, P., Okao-Okuja, Omuut, Akol, Apio, H., & Ndunguru, J. (2019). Tree lab: Portable genomics for early detection of plant viruses and pests in sub-saharan africa. *Genes*, *10*, 632. https://doi.org/10.3390/genes10090632 (cit. on p. 6)

Boza, V., Peresini, P., Brejova, B., & Vinař, T. (2021). Dynamic pooling improves nanopore base calling accuracy. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *PP*, 1–1. https://doi.org/10.1109/TCBB.2021.3128366 (cit. on p. 10)

Boža, V., Perešíni, P., Brejová, B., & Vinař, T. (2020). Deepnano-blitz: A fast base caller for minion nanopore sequencers. *Bioinformatics*, *36*(14), 4191–4192 (cit. on pp. 22, 23, 30, 41).

Bradley, P., Den Bakker, H. C., Rocha, E. P., McVean, G., & Iqbal, Z. (2019). Ultrafast search of all deposited bacterial and viral genomic data. *Nature biotechnology*, *37*(2), 152–159 (cit. on pp. 16, 69).

Branton, D., & Deamer, D. (2019). *Nanopore sequencing: An introduction*. World Scientific Publishing Company. https://books.google.de/books?id=o-aWDwAAQBAJ. (Cit. on pp. 9, 10)

Bray, N. L., Pimentel, H., Melsted, P., & Pachter, L. (2016). Near-optimal probabilistic rna-seq quantification. *Nature biotechnology*, *34*(5), 525–527 (cit. on p. 14).

Breitwieser, F. P., Lu, J., & Salzberg, S. L. (2019). A review of methods and databases for metagenomic classification and assembly. *Briefings in bioinformatics*, *20*(4), 1125–1136 (cit. on pp. 8, 12).

Breslow, A. D., & Jayasena, N. S. (2018). Morton filters: Faster, space-efficient cuckoo filters via biasing, compression, and decoupled logical sparsity. *Proceedings of the VLDB Endowment*, *11*(9), 1041–1055 (cit. on p. 17).

Brinkerhoff, H., Kang, A. S., Liu, J., Aksimentiev, A., & Dekker, C. (2021). Multiple rereads of single proteins at single–amino acid resolution using nanopores. *Science*, eabl4381 (cit. on pp. 42, 102).

Broder, A., & Mitzenmacher, M. (2004). Network applications of bloom filters: A survey. *Internet mathematics*, *1*(4), 485–509 (cit. on p. 16).

Broder, A. Z. (1997). On the resemblance and containment of documents. *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*, 21–29 (cit. on pp. 14, 24).

Brolund, A., & Sandegren, L. (2016). Characterization of esbl disseminating plasmids. *Infectious diseases*, *48*(1), 18–25 (cit. on p. 5).

Brown, S. P., Cornforth, D. M., & Mideo, N. (2012). Evolution of virulence in opportunistic pathogens: Generalism, plasticity, and control. *Trends in microbiology*, *20*(7), 336–342 (cit. on p. 5).

Buytaers, F. E., Saltykova, A., Denayer, S., Verhaegen, B., Vanneste, K., Roosens, N. H. C., Piérard, D., Marchal, K., & De Keersmaecker, S. C. J. (2021). Towards real-time and affordable strain-level metagenomics-based foodborne outbreak investigations using oxford nanopore sequencing technologies. *Frontiers in Microbiology*, *12*. https://doi.org/10.3389/fmicb.2021.738284 (cit. on p. 5)

Byrd, A., Matlock, D., Bagchi, D., Aarattuthodi, S., Harrison, D., Croquette, V., & Raney, K. (2012). Dda helicase tightly couples translocation on single-stranded dna to unwinding of duplex dna: Dda is an optimally active helicase. *Journal of molecular biology*, *420*, 141–54. https://doi.org/10.1016/j.jmb.2012.04.007 (cit. on p. 9)

Calderoni, L., Palmieri, P., & Maio, D. (2015). Location privacy without mutual trust: The spatial bloom filter. *Computer communications*, *68*, 4–16 (cit. on p. 16).

Campanaro, S., Treu, L., Kougias, P. G., Zhu, X., & Angelidaki, I. (2018). Taxonomy of anaerobic digestion microbiome reveals biases associated with the applied high throughput sequencing strategies. *Scientific reports*, *8*(1), 1926 (cit. on p. 11).

Carattoli, A. (2013). Plasmids and the spread of resistance. *International journal of medical microbiology*, *303*(6-7), 298–304 (cit. on p. 46).

Castro-Wallace, S. L., Chiu, C. Y., John, K. K., Stahl, S. E., Rubins, K. H., McIntyre, A. B., Dworkin, J. P., Lupisella, M. L., Smith, D. J., Botkin, D. J., et al. (2017). Nanopore dna sequencing and genome assembly on the international space station. *Scientific reports*, *7*(1), 18022 (cit. on p. 8).

Chakaya, J., Khan, M., Ntoumi, F., Aklillu, E., Fatima, R., Mwaba, P., Kapata, N., Mfinanga, S., Hasnain, S. E., Katoto, P. D., et al. (2021). Global tuberculosis report 2020–reflections on the global tb burden, treatment and prevention efforts. *International journal of infectious diseases*, *113*, S7–S12 (cit. on p. 4).

Chazelle, B., Kilian, J., Rubinfeld, R., & Tal, A. (2004). The bloomier filter: An efficient data structure for static support lookup tables. *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, 30–39 (cit. on p. 17).

Cherf, G., Lieberman, K., Rashid, H., Lam, C., Karplus, K., & Akeson, M. (2012). Automated forward and reverse ratcheting of dna in a nanopore at five angstrom precision. *Nature biotechnology*, *30*, 344–8. https://doi.org/10.1038/nbt.2147 (cit. on p. 9)

Cimino, A. (2022). Here are 2 pieces of good news for illumina [Accessed: (20.07.2023)]. (Cit. on p. 7).

Clamp, M., Fry, B., Kamal, M., Xie, X., Cuff, J., Lin, M. F., Kellis, M., Lindblad-Toh, K., & Lander, E. S. (2007). Distinguishing protein-coding and noncoding genes in the human genome. *Proceedings of the National Academy of Sciences*, *104*(49), 19428–19433. https://doi.org/10.1073/pnas.0709013104 (cit. on p. 6)

Clarke, J., Wu, H.-C., Jayasinghe, L., Patel, A., Reid, S., & Bayley, H. (2009). Continuous base identification for single-molecule nanopore dna sequencing. *Nature nanotechnology*, *4*(4), 265–270 (cit. on p. 8).

Coates, A., Hu, Y., Bax, R., & Page, C. (2002). The future challenges facing the development of new antimicrobial drugs. *Nature reviews Drug discovery*, *1*(11), 895–910 (cit. on p. 4).

Compeau, P. E., Pevzner, P. A., & Tesler, G. (2011). How to apply de bruijn graphs to genome assembly. *Nature biotechnology*, *29*(11), 987–991 (cit. on p. 14).

Coutinho, F., Gregoracci, G., Walter, J., Thompson, C., & Thompson, F. (2018). Metagenomics sheds light on the ecology of marine microbes and their viruses. *Trends in Microbiology*, *26*. https://doi.org/10.1016/j.tim.2018.05.015 (cit. on p. 5)

Cretu Stancu, M., Van Roosmalen, M. J., Renkens, I., Nieboer, M. M., Middelkamp, S., De Ligt, J., Pregno, G., Giachino, D., Mandrile, G., Espejo Valle-Inclan, J., et al. (2017). Mapping and phasing of structural variation in patient genomes using nanopore sequencing. *Nature communications*, *8*(1), 1326 (cit. on p. 97).

Dadi, T. H., Renard, B. Y., Wieler, L. H., Semmler, T., & Reinert, K. (2017). Slimm: Species level identification of microorganisms from metagenomes. *PeerJ*, *5*, e3138 (cit. on p. 68).

Dadi, T. H., Siragusa, E., Piro, V. C., Andrusch, A., Seiler, E., Renard, B. Y., & Reinert, K. (2018). Dream-yara: An exact read mapper for very large databases with short update time. *Bioinformatics*, *34*(17), i766–i772 (cit. on pp. 16, 23, 25, 28, 69, 71).

Danko, D., Bezdan, D., Afshin, E. E., Ahsanuddin, S., Bhattacharya, C., Butler, D. J., Chng, K. R., Donnellan, D., Hecht, J., Jackson, K., et al. (2021). A global metagenomic map of urban microbiomes and antimicrobial resistance. *Cell*, *184*(13), 3376–3393 (cit. on p. 5).

Daszak, P., Cunningham, A. A., & Hyatt, A. D. (2001). Anthropogenic environmental change and the emergence of infectious diseases in wildlife. *Acta tropica*, *78*(2), 103–116 (cit. on p. 4).

de Bernardi Schneider, A., Damodaran, L., & Janies, D. (2017). Transmission network analyses of infectious diseases outbreaks in the genomic era (cit. on p. 7).

Delahaye, C., & Nicolas, J. (2021). Sequencing dna with nanopores: Troubles and biases. *PloS one*, *16*(10), e0257521 (cit. on pp. 9, 10).

Delcher, A. L., Kasif, S., Fleischmann, R. D., Peterson, J., White, O., & Salzberg, S. L. (1999). Alignment of whole genomes. *Nucleic acids research*, *27*(11), 2369–2376 (cit. on p. 13).

Diao, Z., Han, D., Zhang, R., & Li, J. (2022). Metagenomics next-generation sequencing tests take the stage in the diagnosis of lower respiratory tract infections. *Journal of Advanced Research*, *38*, 201–212 (cit. on p. 8).

Dillinger, P. C., & Walzer, S. (2021). Ribbon filter: Practically smaller than bloom and xor. *arXiv preprint arXiv:2103.02515* (cit. on p. 95).

Dilthey, A. T., Jain, C., Koren, S., & Phillippy, A. M. (2019). Strain-level metagenomic assignment and compositional estimation for long reads with metamaps. *Nature communications*, *10*(1), 3066 (cit. on pp. 68, 81, 97, 116).

Douglas, A. E. (2019). Simple animal models for microbiome research. *Nature Reviews Microbiology*, *17*(12), 764–775 (cit. on p. 3).

Doytchinov, V. V., & Dimov, S. G. (2022). Microbial community composition of the antarctic ecosystems: Review of the bacteria, fungi, and archaea identified through an ngs-based metagenomics approach. *Life*, *12*(6), 916 (cit. on p. 68).

Drmanac, R., Sparks, A. B., Callow, M. J., Halpern, A. L., Burns, N. L., Kermani, B. G., Carnevali, P., Nazarenko, I., Nilsen, G. B., Yeung, G., et al. (2010). Human genome sequencing using unchained base reads on self-assembling dna nanoarrays. *Science*, *327*(5961), 78–81 (cit. on p. 7).

Durazzi, F., Sala, C., Castellani, G., Manfreda, G., Remondini, D., & Cesare, A. (2021). Comparison between 16s rrna and shotgun sequencing data for the taxonomic characterization of the gut microbiota. *Scientific Reports*, *11*. https://doi.org/10.1038/s41598-021-82726-y (cit. on p. 11)

Dutta, A., Pellow, D., & Shamir, R. (2022). Parameterized syncmer schemes improve long-read mapping. *PLOS Computational Biology*, *18*(10), e1010638 (cit. on pp. 15, 70, 78).

Edgar, R. (2021). Syncmers are more sensitive than minimizers for selecting conserved k-mers in biological sequences. *PeerJ*, *9*, e10805 (cit. on pp. 15, 70, 78).

Egholm, M., Margulies, M., Altman, W., Attiya, S., Bader, J., Bemben, L., Berka, J., Braverman, M., Chen, Y., Chen, Z., et al. (2005). Genome sequencing in open microfabricated high density picoliter reactors. *Nature*, *437*, 376–380 (cit. on p. 7).

Eid, J., Fehr, A., Gray, J., Luong, K., Lyle, J., Otto, G., Peluso, P., Rank, D., Baybayan, P., Bettman, B., et al. (2009). Real-time dna sequencing from single polymerase molecules. *Science*, *323*(5910), 133–138 (cit. on p. 8).

Ellabaan, M. M., Munck, C., Porse, A., Imamovic, L., & Sommer, M. O. (2021). Forecasting the dissemination of antibiotic resistance genes across bacterial genomes. *Nature communications*, *12*(1), 2435 (cit. on p. 5).

Estrada-Peña, A., Ostfeld, R. S., Peterson, A. T., Poulin, R., & de la Fuente, J. (2014). Effects of environmental change on zoonotic disease risk: An ecological primer. *Trends in parasitology*, *30*(4), 205–214 (cit. on p. 4).

Euskirchen, P., Bielle, F., Labreche, K., Kloosterman, W. P., Rosenberg, S., Daniau, M., Schmitt, C., Masliah-Planchon, J., Bourdeaut, F., Dehais, C., et al. (2017). Same-day genomic and epigenomic diagnosis of brain tumors using real-time nanopore sequencing. *Acta neuropathologica*, *134*, 691–703 (cit. on p. 97).

Fan, B., Andersen, D. G., Kaminsky, M., & Mitzenmacher, M. D. (2014). Cuckoo filter: Practically better than bloom. *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, 75–88 (cit. on pp. 17, 69).

Fan, L., Cao, P., Almeida, J., & Broder, A. Z. (2000). Summary cache: A scalable wide-area web cache sharing protocol. *IEEE/ACM transactions on networking*, *8*(3), 281–293 (cit. on p. 16).

Ferguson, S., McLay, T., Andrew, R. L., Bruhl, J. J., Schwessinger, B., Borevitz, J., & Jones, A. (2022). Species-specific basecallers improve actual accuracy of nanopore sequencing in plants. *Plant Methods*, *18*(1), 1–11 (cit. on p. 82).

Fijan, S. (2014). Microorganisms with claimed probiotic properties: An overview of recent literature. *International journal of environmental research and public health*, *11*(5), 4745–4767 (cit. on p. 4).

Firtina, C., Mansouri Ghiasi, N., Lindegger, J., Singh, G., Cavlak, M. B., Mao, H., & Mutlu, O. (2023). Rawhash: Enabling fast and accurate real-time analysis of raw nanopore signals for large genomes. *Bioinformatics*, *39*(Supplement_1), i297–i307 (cit. on p. 100).

Fischer, M., Strauch, B., & Renard, B. Y. (2017). Abundance estimation and differential testing on strain level in metagenomics data. *Bioinformatics*, *33*(14), i124–i132 (cit. on p. 68).

Flajolet, P., Fusy, É., Gandouet, O., & Meunier, F. (2007). Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm. *Discrete Mathematics and Theoretical Computer Science*, 137–156 (cit. on p. 75).

Fleming, A. (1941). Penicillin. *British medical journal*, *2*(4210), 386 (cit. on p. 4).

Fouhy, F., Clooney, A. G., Stanton, C., Claesson, M. J., & Cotter, P. D. (2016). 16s rrna gene sequencing of mock microbial populations-impact of dna extraction method, primer choice and sequencing platform. *BMC microbiology*, *16*(1), 1–13 (cit. on p. 11).

Frost, F., Weiss, F. U., Sendler, M., Kacprowski, T., Rühlemann, M., Bang, C., Franke, A., Völker, U., Völzke, H., Lamprecht, G., et al. (2020). The gut microbiome in patients with chronic pancreatitis is characterized by significant dysbiosis and overgrowth by opportunistic pathogens. *Clinical and Translational Gastroenterology*, *11*(9) (cit. on p. 5).

Fuhrmeister, E. R., Voth-Gaeddert, L. E., Metilda, A., Tai, A., Batorsky, R. E., Veeraraghavan, B., Ward, H. D., Kang, G., & Pickering, A. J. (2021). Surveillance of potential pathogens and antibiotic resistance in wastewater and surface water from boston, usa and vellore, india using long-read metagenomic sequencing. *medRxiv*, 2021–04 (cit. on p. 103).

Gardy, J., & Loman, N. (2017). Towards a genomics-informed, real-time, global pathogen surveillance system. *Nature Reviews Genetics*, *19*, nrg.2017.88. https://doi.org/10.1038/nrg.2017.88 (cit. on p. 6)

Gerba, C. P., & Smith, J. E. (2005). Sources of pathogenic microorganisms and their fate during land application of wastes. *Journal of environmental quality*, *34*(1), 42–48 (cit. on p. 4).

Gilbert, J., O'Dor, R., King, N., & Vogel, T. (2011). The importance of metagenomic surveys to microbial ecology: Or why darwin would have been a metagenomic scientist. *Microbial informatics and experimentation*, *1*, 5. https://doi.org/10.1186/2042-5783-1-5 (cit. on p. 5)

Gilbert, W. (1986). Origin of life: The rna world. *Nature*, *319*, 618–618 (cit. on p. 6).

Glatter, K. A., & Finkelman, P. (2021). History of the plague: An ancient pandemic for the age of covid-19. *The American journal of medicine*, *134*(2), 176–181 (cit. on p. 4).

Gonçalves, O., Rosa, J., França, K., Bossé, J., Santana, M., Langford, P. R., Bazzolli, D., et al. (2021). Mobile genetic elements drive antimicrobial resistance gene spread in pasteurellaceae species. *Frontiers in microbiology*, *12*, 773284–773284 (cit. on p. 46).

Goodwin, S., McPherson, J. D., & McCombie, W. R. (2016). Coming of age: Ten years of next-generation sequencing technologies. *Nature Reviews Genetics*, *17*(6), 333–351 (cit. on pp. 7, 11).

Gougoulias, C., Clark, J. M., & Shaw, L. J. (2014). The role of soil microbes in the global carbon cycle: Tracking the below-ground microbial processing of plant-derived carbon for manipulating carbon dynamics in agricultural systems. *Journal of the Science of Food and Agriculture*, *94*(12), 2362–2371 (cit. on p. 3).

Gowers, G.-O. F., Vince, O., Charles, J.-H., Klarenberg, I., Ellis, T., & Edwards, A. (2019). Entirely off-grid and solar-powered dna sequencing of microbial communities during an ice cap traverse expedition. *Genes*, *10*(11), 902 (cit. on p. 8).

Graf, T. M., & Lemire, D. (2020). Xor filters: Faster and smaller than bloom and cuckoo filters. *Journal of Experimental Algorithmics (JEA)*, *25*, 1–16 (cit. on pp. 17, 69, 71, 73, 94).

Graf, T. M., & Lemire, D. (2022). Binary fuse filters: Fast and smaller than xor filters. *Journal of Experimental Algorithmics (JEA)*, *27*(1), 1–15 (cit. on pp. 17, 69, 95, 104).

Greninger, A., Naccache, S., Federman, S., Yu, G., Mbala, P., Bres, V., Stryke, D., Bouquet, J., Somasekar, S., Linnen, J., Dodd, R., Mulembakani, P., Schneider, B., Muyembe-Tamfum, J.-J., Stramer, S., & Chiu, C. (2015). Rapid metagenomic identification of viral pathogens in clinical samples by real-time nanopore se-

quencing analysis. *Genome Medicine*, *7*. https://doi.org/10.1186/s13073-015-0220-9 (cit. on pp. 6, 8, 97)

Groussin, M., Poyet, M., Sistiaga, A., Kearney, S. M., Moniz, K., Noel, M., Hooker, J., Gibbons, S. M., Segurel, L., Froment, A., et al. (2021). Elevated rates of horizontal gene transfer in the industrialized human microbiome. *Cell*, *184*(8), 2053–2067 (cit. on p. 5).

Gu, W., Miller, S., & Chiu, C. Y. (2019). Clinical metagenomic next-generation sequencing for pathogen detection. *Annual Review of Pathology: Mechanisms of Disease*, *14*, 319–338 (cit. on p. 5).

Gupta, A., Gupta, R., & Singh, R. L. (2017). Microbes and environment. *Principles and applications of environmental biotechnology for a sustainable future*, 43–84 (cit. on p. 4).

Gurevich, A., Saveliev, V., Vyahhi, N., & Tesler, G. (2013). Quast: Quality assessment tool for genome assemblies. *Bioinformatics*, *29*(8), 1072–1075 (cit. on pp. 51, 64).

Harris, R. S., & Medvedev, P. (2020). Improved representation of sequence bloom trees. *Bioinformatics*, *36*(3), 721–727 (cit. on p. 69).

Hendriksen, R. S., Munk, P., Njage, P., Van Bunnik, B., McNally, L., Lukjancenko, O., Röder, T., Nieuwenhuijse, D., Pedersen, S. K., Kjeldgaard, J., et al. (2019). Global monitoring of antimicrobial resistance based on metagenomics analyses of urban sewage. *Nature communications*, *10*(1), 1124 (cit. on p. 5).

Hermans, S. M., Buckley, H. L., Case, B. S., Curran-Cournane, F., Taylor, M., & Lear, G. (2017). Bacteria as emerging indicators of soil condition. *Applied and environmental microbiology*, *83*(1), e02826–16 (cit. on p. 3).

Hess, W. R. (2004). Genome analysis of marine photosynthetic microbes and their global role. *Current opinion in biotechnology*, *15*(3), 191–198 (cit. on p. 3).

Hidalgo, L., de Been, M., Rogers, M. R., Schürch, A. C., Scharringa, J., van der Zee, A., Bonten, M. J., & Fluit, A. C. (2019). Sequence-based epidemiology of an oxa-48 plasmid during a hospital outbreak. *Antimicrobial Agents and Chemotherapy*, *63*(12), e01204–19 (cit. on p. 46).

Holley, G., & Melsted, P. (2020). Bifrost: Highly parallel construction and indexing of colored and compacted de bruijn graphs. *Genome biology*, *21*(1), 1–20 (cit. on p. 69).

Holmes, E. (2009). *The evolution and emergence of rna viruses*. OUP Oxford. https://books.google.de/books?id=fpoUDAAAQBAJ. (Cit. on p. 6)

Hong, C., Manimaran, S., Shen, Y., Perez-Rogers, J. F., Byrd, A. L., Castro-Nallar, E., Crandall, K. A., & Johnson, W. E. (2014). Pathoscope 2.0: A complete

computational framework for strain identification in environmental or clinical sequencing samples. *Microbiome*, *2*(1), 1–15 (cit. on pp. 13, 68).

Hugenholtz, P., Skarshewski, A., & Parks, D. (2016). Genome-based microbial taxonomy coming of age. *Cold Spring Harbor Perspectives in Biology*, *8*, a018085. https://doi.org/10.1101/cshperspect.a018085 (cit. on p. 7)

Hugenholtz, P., & Tyson, G. W. (2008). Metagenomics. *Nature*, *455*(7212), 481–483 (cit. on p. 5).

Huse, S. M., Dethlefsen, L., Huber, J. A., Welch, D. M., Relman, D. A., & Sogin, M. L. (2008). Exploring microbial diversity and taxonomy using ssu rrna hypervariable tag sequencing. *PLoS genetics*, *4*(11), e1000255 (cit. on p. 11).

Jaccard, P. (1912). The distribution of the flora in the alpine zone. 1. *New phytologist*, *11*(2), 37–50 (cit. on p. 14).

Jain, M., Koren, S., Miga, K. H., Quick, J., Rand, A. C., Sasani, T. A., Tyson, J. R., Beggs, A. D., Dilthey, A. T., Fiddes, I. T., et al. (2018). Nanopore sequencing and assembly of a human genome with ultra-long reads. *Nature biotechnology*, *36*(4), 338–345 (cit. on p. 97).

Johnson, S., Zaikova, E., Goerlitz, D., Bai, Y., & Tighe, S. (2017). Real-time dna sequencing in the antarctic dry valleys using the oxford nanopore sequencer. *Journal of biomolecular techniques : JBT*, *28*. https://doi.org/10.7171/jbt.17-2801-009 (cit. on p. 6)

Juul, S., Izquierdo, F., Hurst, A., Dai, X., Wright, A., Kulesha, E., Pettett, R., & Turner, D. J. (2015). What's in my pot? real-time species identification on the minion. *BioRxiv*, 030742 (cit. on pp. 81, 95).

Kafetzopoulou, L., Pullan, S., Lemey, P., Suchard, M., Ehichioya, D., Pahlmann, M., Thielebein, A., Hinzmann, J., Oestereich, L., Wozniak, D., et al. (2019). Metagenomic sequencing at the epicenter of the nigeria 2018 lassa fever outbreak. *Science*, *363*(6422), 74–77 (cit. on p. 103).

Kalsoom, M., Rehman, F., Shafique, T., Junaid, S., Khalid, N., Adnan, M., Zafar, I., Tariq, M. A., Raza, M., Zahra, A., et al. (2020). Biological importance of microbes in agriculture, food and pharmaceutical industry: A review. *Innovare Journal of Life Sciences*, *8*(6) (cit. on p. 4).

Kasianowicz, J. J., Brandin, E., Branton, D., & Deamer, D. W. (1996). Characterization of individual polynucleotide molecules using a membrane channel. *Proceedings of the National Academy of Sciences*, *93*(24), 13770–13773 (cit. on pp. 8, 9).

Kav, A. B., Benhar, I., & Mizrahi, I. (2013). A method for purifying high quality and high yield plasmid dna for metagenomic and deep sequencing approaches. *Journal of microbiological methods*, *95*(2), 272–279 (cit. on p. 46).

Kelly, M. S., Bunyavanich, S., Phipatanakul, W., & Lai, P. S. (2022). The environmental microbiome, allergic disease, and asthma. *The Journal of Allergy and Clinical Immunology: In Practice*, *10*(9), 2206–2217 (cit. on p. 5).

Kent, W. J. (2002). Blat—the blast-like alignment tool. *Genome research*, *12*(4), 656–664 (cit. on p. 13).

Kim, D., Song, L., Breitwieser, F. P., & Salzberg, S. L. (2016). Centrifuge: Rapid and sensitive classification of metagenomic sequences. *Genome research*, *26*(12), 1721–1729 (cit. on pp. 68, 81, 112).

Kipp, E. J., Lindsey, L. L., Khoo, B., Faulk, C., Oliver, J. D., & Larsen, P. A. (2023). Metagenomic surveillance for bacterial tick-borne pathogens using nanopore adaptive sampling. *Scientific reports*, *13*(1), 10991 (cit. on p. 47).

Kirchman, D. L. (2018). Degradation of organic matter. In *Processes in Microbial Ecology*. Oxford University Press. https://doi.org/10.1093/oso/9780198789406.003.0007. (Cit. on p. 3)

Knoll, A. H. (2015). Paleobiological perspectives on early microbial evolution. *Cold Spring Harbor Perspectives in Biology*, *7*(7), a018093 (cit. on p. 3).

Ko, K., Chng, K. R., & Nagarajan, N. (2022). Metagenomics-enabled microbial surveillance. *Nature Microbiology*, *7*, 486–496. https://doi.org/10.1038/s41564-022-01089-w (cit. on p. 5)

Kolmogorov, M., Bickhart, D. M., Behsaz, B., Gurevich, A., Rayko, M., Shin, S. B., Kuhn, K., Yuan, J., Polevikov, E., Smith, T. P., et al. (2020). Metaflye: Scalable long-read metagenome assembly using repeat graphs. *Nature Methods*, *17*(11), 1103–1110 (cit. on pp. 50, 64).

Kolmogorov, M., Yuan, J., Lin, Y., & Pevzner, P. A. (2019). Assembly of long, error-prone reads using repeat graphs. *Nature biotechnology*, *37*(5), 540–546 (cit. on p. 51).

Koslicki, D., & Zabeti, H. (2019). Improving minhash via the containment index with applications to metagenomic analysis. *Applied Mathematics and Computation*, *354*, 206–215 (cit. on pp. 14, 24).

Kovaka, S., Fan, Y., Ni, B., Timp, W., & Schatz, M. C. (2021). Targeted nanopore sequencing by real-time mapping of raw electrical signal with uncalled. *Nature Biotechnology*, *39*(4), 431–441 (cit. on pp. 10, 22, 46, 98, 102).

Kumamoto, C. A., Gresnigt, M. S., & Hube, B. (2020). The gut, the bad and the harmless: Candida albicans as a commensal and opportunistic pathogen in the intestine. *Current opinion in microbiology*, *56*, 7–15 (cit. on p. 5).

Ladizinski, B., McLean, R., Lee, K. C., Elpern, D. J., & Eron, L. (2014). The human skin microbiome. *International journal of dermatology*, *53*(9), 1177–1179 (cit. on p. 4).

Lander, E., Chen, C., Linton, L., Birren, B., Nusbaum, C., Zody, M., Baldwin, J., Devon, K., Dewar, K., Doyle, M., Fitzhugh, W., Funke, R., Gaige, D., Harris, K., Heaford, A., Howland, J., Kann, L., Lehoczky, J., LeVine, R., & Rowen, L. (2001). Initial sequencing and analysis of the human genome. *Nature*, *409* (cit. on p. 7).

Lane, D. J., Pace, B., Olsen, G. J., Stahl, D. A., Sogin, M. L., & Pace, N. R. (1985). Rapid determination of 16s ribosomal rna sequences for phylogenetic analyses. *Proceedings of the National Academy of Sciences*, *82*(20), 6955–6959 (cit. on p. 11).

Langmead, B., & Salzberg, S. L. (2012). Fast gapped-read alignment with bowtie 2. *Nature methods*, *9*(4), 357–359 (cit. on pp. 13, 68).

Leggett, R. M., & Clark, M. D. (2017). A world of opportunities with nanopore sequencing. *Journal of Experimental Botany*, *68*(20), 5419–5429 (cit. on p. 21).

Leung, C.-M., Li, D., Xin, Y., Law, W.-C., Zhang, Y., Ting, H.-F., Luo, R., & Lam, T.-W. (2020). Megapath: Sensitive and rapid pathogen detection using metagenomic ngs data. *BMC genomics*, *21*(6), 1–9 (cit. on p. 79).

Levene, M. J., Korlach, J., Turner, S. W., Foquet, M., Craighead, H. G., & Webb, W. W. (2003). Zero-mode waveguides for single-molecule analysis at high concentrations. *science*, *299*(5607), 682–686 (cit. on p. 8).

Levenshtein, V. I., et al. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, *10*(8), 707–710 (cit. on p. 13).

Li, H. (2018). Minimap2: Pairwise alignment for nucleotide sequences. *Bioinformatics*, *34*(18), 3094–3100 (cit. on pp. 13, 14, 22, 24, 31, 47–49, 62, 83).

Li, H., & Durbin, R. (2010). Fast and accurate long-read alignment with burrows–wheeler transform. *Bioinformatics*, *26*(5), 589–595 (cit. on p. 13).

Li, H., Handsaker, B., Wysoker, A., Fennell, T., Ruan, J., Homer, N., Marth, G., Abecasis, G., & Durbin, R. (2009). The sequence alignment/map format and samtools. *Bioinformatics*, *25*(16), 2078–2079 (cit. on pp. 50, 62).

Li, H. (2015). Microbiome, metagenomics, and high-dimensional compositional data analysis. *Annual Review of Statistics and Its Application*, *2*, 73–94 (cit. on p. 5).

Liang, Q., Bible, P. W., Liu, Y., Zou, B., & Wei, L. (2020). Deepmicrobes: Taxonomic classification for metagenomics with deep learning. *NAR Genomics and Bioinformatics*, *2*(1), lqaa009 (cit. on p. 68).

Lindner, M. S., & Renard, B. Y. (2013). Metagenomic abundance estimation and diagnostic testing on species level. *Nucleic acids research*, *41*(1), e10–e10 (cit. on p. 68).

Liu, Y., & Schmidt, B. (2012). Long read alignment based on maximal exact match seeds. *Bioinformatics*, *28*(18), i318–i324 (cit. on p. 13).

Loka, T. P., Tausch, S. H., & Renard, B. Y. (2019). Reliable variant calling during runtime of illumina sequencing. *Scientific Reports*, *9*(1), 16502 (cit. on p. 97).

Loman, N. J., Constantinidou, C., Christner, M., Rohde, H., Chan, J. Z.-M., Quick, J., Weir, J. C., Quince, C., Smith, G. P., Betley, J. R., Aepfelbacher, M., & Pallen, M. J. (2013). A Culture-Independent Sequence-Based Metagenomics Approach to the Investigation of an Outbreak of Shiga-Toxigenic Escherichia coli O104:H4. *JAMA*, *309*(14), 1502–1510. https://doi.org/10.1001/jama.2013.3231 (cit. on p. 5)

Loose, M., Malla, S., & Stout, M. (2016). Real-time selective sequencing using nanopore technology. *Nature methods*, *13*(9), 751–754 (cit. on pp. 10, 22, 46, 98).

Lowry, C. A., Smith, D. G., Siebler, P. H., Schmidt, D., Stamper, C. E., Hassell, J. E., Yamashita, P. S., Fox, J. H., Reber, S. O., Brenner, L. A., et al. (2016). The microbiota, immunoregulation, and mental health: Implications for public health. *Current environmental health reports*, *3*, 270–286 (cit. on p. 5).

Lynch, M. D., & Neufeld, J. D. (2015). Ecology and exploration of the rare biosphere. *Nature Reviews Microbiology*, *13*(4), 217–229 (cit. on p. 46).

Malmstrom, C. M., Martin, M. D., & Gagnevin, L. (2022). Exploring the emergence and evolution of plant pathogenic microbes using historical and paleontological sources. *Annual Review of Phytopathology*, *60*, 187–209 (cit. on p. 4).

Manaia, C. M. (2017). Assessing the risk of antibiotic resistance transmission from the environment to humans: Non-direct proportionality between abundance and risk. *Trends in microbiology*, *25*(3), 173–181 (cit. on p. 4).

Mangul, S., Martin, L. S., Hill, B. L., Lam, A. K.-M., Distler, M. G., Zelikovsky, A., Eskin, E., & Flint, J. (2019). Systematic benchmarking of omics computational tools. *Nature communications*, *10*(1), 1393 (cit. on p. 104).

Marçais, G., Delcher, A. L., Phillippy, A. M., Coston, R., Salzberg, S. L., & Zimin, A. (2018). Mummer4: A fast and versatile genome alignment system. *PLoS computational biology*, *14*(1), e1005944 (cit. on p. 13).

Marchesi, J. R., & Ravel, J. (2015). The vocabulary of microbiome research: A proposal. (Cit. on p. 11).

Marchet, C., Boucher, C., Puglisi, S. J., Medvedev, P., Salson, M., & Chikhi, R. (2021). Data structures based on k-mers for querying large collections of sequencing data sets. *Genome Research*, *31*(1), 1–12 (cit. on p. 14).

Marić, J., Križanović, K., Riondet, S., Nagarajan, N., & Šikić, M. (2021). Benchmarking metagenomic classification tools for long-read sequencing data. *BioRxiv*, 2020–11 (cit. on p. 104).

Marquet, M., Zöllkau, J., Pastuschek, J., Viehweger, A., Schleußner, E., Makarewicz, O., Pletz, M. W., Ehricht, R., & Brandt, C. (2022). Evaluation of microbiome enrichment and host dna depletion in human vaginal samples using oxford nanopore's adaptive sequencing. *Scientific reports*, *12*(1), 1–10 (cit. on pp. 22, 42, 46).

Martin, S., Heavens, D., Lan, Y., Horsfield, S., Clark, M. D., & Leggett, R. M. (2022). Nanopore adaptive sampling: A tool for enrichment of low abundance species in metagenomic samples. *Genome Biology*, *23*(1), 1–27 (cit. on pp. 42, 46, 47, 50, 54, 55, 59, 65, 98).

Martínez, J. L., Coque, T. M., & Baquero, F. (2015). What is a resistance gene? ranking risk in resistomes. *Nature Reviews Microbiology*, *13*(2), 116–123 (cit. on p. 4).

Mattick, J. S. (2011). The central role of rna in human development and cognition [Turin Special Issue: Biochemistry for Tomorrow's Medicine]. *FEBS Letters*, *585*(11), 1600–1616. https://doi.org/https://doi.org/10.1016/j.febslet.2011.05.001 (cit. on p. 6)

Maxam, A. M., & Gilbert, W. (1977). A new method for sequencing dna. *Proceedings of the National Academy of Sciences*, *74*(2), 560–564. https://doi.org/10.1073/pnas.74.2.560 (cit. on p. 7)

McIntyre, A., Ounit, R., Afshinnekoo, E., Prill, R., Henaff, E., Alexander, N., Minot, S., Danko, D., Foox, J., Ahsanuddin, S., Tighe, S., Hasan, N., Subramanian, P., Moffat, K., Levy, S., Lonardi, S., Greenfield, N., Colwell, R., Rosen, G., & Mason, C. (2017). Comprehensive benchmarking and ensemble approaches for metagenomic classifiers. *Genome Biology*, *18*, 182. https://doi.org/10.1186/s13059-017-1299-7 (cit. on p. 11)

Mehringer, S., Seiler, E., Droop, F., Darvish, M., Rahn, R., Vingron, M., & Reinert, K. (2023). Hierarchical interleaved bloom filter: Enabling ultrafast, approximate sequence queries. *Genome Biology*, *24*(1), 1–25 (cit. on pp. 69, 75, 94, 95).

Meller, A., Nivón, L., & Branton, D. (2001). Voltage-driven dna translocations through a nanopore. *Physical review letters*, *86*, 3435–8. https://doi.org/10.1103/PhysRevLett.86.3435 (cit. on p. 10)

Menzel, P., Ng, K. L., & Krogh, A. (2016). Fast and sensitive taxonomic classification for metagenomics with kaiju. *Nature communications*, *7*(1), 11257 (cit. on p. 12).

Meyer, F., Fritz, A., Deng, Z.-L., Koslicki, D., Lesker, T. R., Gurevich, A., Robertson, G., Alser, M., Antipov, D., Beghini, F., et al. (2022). Critical assessment of metagenome interpretation: The second round of challenges. *Nature methods*, *19*(4), 429–440 (cit. on pp. 13, 104).

Miga, K. H., Newton, Y., Jain, M., Altemose, N., Willard, H. F., & Kent, W. J. (2014). Centromere reference models for human chromosomes x and y satellite arrays. *Genome research*, *24*(4), 697–707 (cit. on p. 97).

Mikheyev, A. S., & Tin, M. M. (2014). A first look at the oxford nanopore minion sequencer. *Molecular ecology resources*, *14*(6), 1097–1102 (cit. on pp. 8, 21).

Milanese, A., Mende, D. R., Paoli, L., Salazar, G., Ruscheweyh, H.-J., Cuenca, M., Hingamp, P., Alves, R., Costea, P. I., Coelho, L. P., et al. (2019). Microbial abundance, activity and population genomic profiling with motus2. *Nature communications*, *10*(1), 1014 (cit. on p. 68).

Mir, M. A. (2022). *Human pathogenic microbes: Diseases and concerns*. Academic Press. (Cit. on p. 4).

Mitzenmacher, M. (2001). Compressed bloom filters. *Proceedings of the twentieth annual ACM symposium on Principles of distributed computing*, 144–150 (cit. on p. 16).

Mitzenmacher, M., Pontarelli, S., & Reviriego, P. (2020). Adaptive cuckoo filters. (Cit. on p. 69).

Mock, F., Kretschmer, F., Kriese, A., Böcker, S., & Marz, M. (2022). Taxonomic classification of dna sequences beyond sequence similarity using deep neural networks. *Proceedings of the National Academy of Sciences*, *119*(35), e2122636119 (cit. on p. 68).

Mongan, A. E., Tuda, J. S. B., & Runtuwene, L. R. (2020). Portable sequencer in the fight against infectious disease. *Journal of human genetics*, *65*(1), 35–40 (cit. on p. 22).

Munro, R., Santos, R., Payne, A., Forey, T., Osei, S., Holmes, N., & Loose, M. (2022). Minotour, real-time monitoring and analysis for nanopore sequencers. *Bioinformatics*, *38*(4), 1133–1135 (cit. on p. 95).

Munro, R. J., Payne, A., & Loose, M. W. (2023). Icarust, a real-time simulator for oxford nanopore adaptive sampling. *bioRxiv*, 2023–05 (cit. on p. 103).

Murray, C. J., Ikuta, K. S., Sharara, F., Swetschinski, L., Aguilar, G. R., Gray, A., Han, C., Bisignano, C., Rao, P., Wool, E., et al. (2022). Global burden of bacterial

antimicrobial resistance in 2019: A systematic analysis. *The Lancet* (cit. on p. 45).

Muthukumar, M. (2016). *Polymer translocation*. CRC Press. https://books.google.de/books?id=PcuT-ibRtRIC. (Cit. on p. 9)

Naarmann-de Vries, I. S., Eschenbach, J., & Dieterich, C. (2022). Improved nanopore direct rna sequencing of cardiac myocyte samples by selective mt-rna depletion. *Journal of molecular and cellular cardiology*, *163*, 175–186 (cit. on p. 101).

Naarmann-de Vries, I. S., Gjerga, E., Gandor, C. L., & Dieterich, C. (2022). Adaptive sampling as tool for nanopore direct rna-sequencing. *bioRxiv*, 2022–10 (cit. on p. 101).

Naicker, P. R. (2011). The impact of climate change and other factors on zoonotic diseases. *Archives of Clinical Microbiology*, *2*(2) (cit. on p. 4).

Nasko, D. J., Koren, S., Phillippy, A. M., & Treangen, T. J. (2018). Refseq database growth influences the accuracy of k-mer-based lowest common ancestor species identification. *Genome biology*, *19*(1), 1–10 (cit. on p. 12).

Nearing, J. T., Comeau, A. M., & Langille, M. G. (2021). Identifying biases and their potential solutions in human microbiome studies. *Microbiome*, *9*(1), 1–22 (cit. on p. 12).

Needleman, S. B., & 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 (cit. on p. 13).

Nicholls, S. M., Quick, J. C., Tang, S., & Loman, N. J. (2019). Ultra-deep, long-read nanopore sequencing of mock microbial community standards. *Gigascience*, *8*(5), giz043 (cit. on pp. 32, 33, 82).

Nicolaou, K. C., & Rigol, S. (2018). A brief history of antibiotics and select advances in their synthesis. *The Journal of antibiotics*, *71*(2), 153–184 (cit. on p. 4).

Nishino, K., Yamasaki, S., Nakashima, R., Zwama, M., & Hayashi-Nishino, M. (2021). Function and inhibitory mechanisms of multidrug efflux pumps. *Frontiers in Microbiology*, *12*, 737288 (cit. on p. 4).

Norel, R., Rice, J. J., & Stolovitzky, G. (2011). The self-assessment trap: Can we all be better than average? *Molecular systems biology*, *7*(1), 537 (cit. on p. 104).

Norman, A., Hansen, L. H., & Sørensen, S. J. (2009). Conjugative plasmids: Vessels of the communal gene pool. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *364*(1527), 2275–2289 (cit. on p. 5).

Nurk, S., Koren, S., Rhie, A., Rautiainen, M., Bzikadze, A. V., Mikheenko, A., Vollger, M. R., Altemose, N., Uralsky, L., Gershman, A., et al. (2022). The complete sequence of a human genome. *Science*, *376*(6588), 44–53 (cit. on pp. 8, 37).

O'Brien, H. E., Parrent, J. L., Jackson, J. A., Moncalvo, J.-M., & Vilgalys, R. (2005). Fungal community analysis by large-scale sequencing of environmental samples. *Applied and environmental microbiology*, *71*(9), 5544–5550 (cit. on p. 11).

Olasagasti, F., Lieberman, K. R., Benner, S., Cherf, G. M., Dahl, J. M., Deamer, D. W., & Akeson, M. (2010). Replication of individual dna molecules under electronic control using a protein nanopore. *Nature nanotechnology*, *5*(11), 798–806 (cit. on p. 8).

O'Leary, N. A., Wright, M. W., Brister, J. R., Ciufo, S., Haddad, D., McVeigh, R., Rajput, B., Robbertse, B., Smith-White, B., Ako-Adjei, D., et al. (2016). Reference sequence (refseq) database at ncbi: Current status, taxonomic expansion, and functional annotation. *Nucleic acids research*, *44*(D1), D733–D745 (cit. on pp. 68, 81, 112).

Ondov, B. D., Treangen, T. J., Melsted, P., Mallonee, A. B., Bergman, N. H., Koren, S., & Phillippy, A. M. (2016). Mash: Fast genome and metagenome distance estimation using minhash. *Genome biology*, *17*(1), 1–14 (cit. on p. 24).

O'Neil, J. (2014). Tackling a crisis for the health and wealth of nations. *World Health Organization* (cit. on p. 45).

O'Neill, J. (2016). Tackling drug-resistant infections globally: Final report and recommendations. *Review on Antimicrobial Resistance* (cit. on p. 45).

Ono, Y., Asai, K., & Hamada, M. (2021). Pbsim2: A simulator for long-read sequencers with a novel generative model of quality scores. *Bioinformatics*, *37*(5), 589–595 (cit. on pp. 34, 82).

Oren, A. (2008). Microbial life at high salt concentrations: Phylogenetic and metabolic diversity. *Saline systems*, *4*, 1–13 (cit. on p. 3).

Orlek, A., Stoesser, N., Anjum, M. F., Doumith, M., Ellington, M. J., Peto, T., Crook, D., Woodford, N., Walker, A. S., Phan, H., et al. (2017). Plasmid classification in an era of whole-genome sequencing: Application in studies of antibiotic resistance epidemiology. *Frontiers in microbiology*, *8*, 182 (cit. on p. 46).

Pallerla, S. R., Van Dong, D., Linh, L. T. K., Van Son, T., Quyen, D. T., Hoan, P. Q., Trung, N. T., Rüter, J., Boutin, S., Nurjadi, D., et al. (2022). Diagnosis of pathogens causing bacterial meningitis using nanopore sequencing in a resource-limited setting. *Annals of Clinical Microbiology and Antimicrobials*, *21*(1), 1–8 (cit. on p. 97).

Pandey, P., Almodaresi, F., Bender, M. A., Ferdman, M., Johnson, R., & Patro, R. (2018). Mantis: A fast, small, and exact large-scale sequence-search index. *Cell systems*, *7*(2), 201–207 (cit. on p. 69).

Pandey, P., Bender, M. A., Johnson, R., & Patro, R. (2017). A general-purpose counting filter: Making every bit count. *Proceedings of the 2017 ACM international conference on Management of Data*, 775–787 (cit. on p. 17).

Paniagua Voirol, L. R., Valsamakis, G., Yu, M., Johnston, P. R., & Hilker, M. (2021). How the 'kitome'influences the characterization of bacterial communities in lepidopteran samples with low bacterial biomass. *Journal of Applied Microbiology*, *130*(6), 1780–1793 (cit. on p. 12).

Parks, D. H., Chuvochina, M., Rinke, C., Mussig, A. J., Chaumeil, P.-A., & Hugenholtz, P. (2022). Gtdb: An ongoing census of bacterial and archaeal diversity through a phylogenetically consistent, rank normalized and complete genome-based taxonomy. *Nucleic acids research*, *50*(D1), D785–D794 (cit. on pp. 68, 78).

Partridge, S. R., Kwong, S. M., Firth, N., & Jensen, S. O. (2018). Mobile genetic elements associated with antimicrobial resistance. *Clinical microbiology reviews*, *31*(4), e00088–17 (cit. on p. 46).

Patel, N., Ferns, B., Nastouli, E., Kozlakidis, Z., Kellam, P., & Morris, S. (2016). Cost analysis of standard sanger sequencing versus next generation sequencing in the iconic study. *The Lancet*, *388*, S86. https://doi.org/10.1016/S0140-6736(16)32322-4 (cit. on p. 7)

Patro, J. N., Ramachandran, P., Barnaba, T., Mammel, M. K., Lewis, J. L., & Elkins, C. A. (2016). Culture-independent metagenomic surveillance of commercially available probiotics with high-throughput next-generation sequencing. *MSphere*, *1*(2), 10–1128 (cit. on p. 8).

Payne, A., Holmes, N., Clarke, T., Munro, R., Debebe, B. J., & Loose, M. (2021). Readfish enables targeted nanopore sequencing of gigabase-sized genomes. *Nature biotechnology*, *39*(4), 442–450 (cit. on pp. 10, 22–24, 37, 46–48, 54, 55, 98).

Payne, A., Holmes, N., Rakyan, V., & Loose, M. (2019). Bulkvis: A graphical viewer for oxford nanopore bulk fast5 files. *Bioinformatics*, *35*(13), 2193–2198 (cit. on pp. 8, 36).

Pearman, W. S., Freed, N. E., & Silander, O. K. (2020). Testing the advantages and disadvantages of short-and long-read eukaryotic metagenomics using simulated reads. *BMC bioinformatics*, *21*(1), 1–15 (cit. on pp. 8, 97).

Pingale, S., & Virkar, P. (2013). Study of influence of phosphate dissolving microorganisms on yield and phosphate uptake by crops. *Eur. J. Exp. Biol*, *3*, 191–193 (cit. on p. 3).

Piro, V. C., Dadi, T. H., Seiler, E., Reinert, K., & Renard, B. Y. (2020). Ganon: Precise metagenomics classification against large and up-to-date sets of reference se-

quences. *Bioinformatics*, *36*(Supplement_1), i12–i20 (cit. on pp. 16, 23, 68, 69, 78, 81, 116).

Piro, V. C., Lindner, M. S., & Renard, B. Y. (2016). Dudes: A top-down taxonomic profiler for metagenomics. *Bioinformatics*, *32*(15), 2272–2280 (cit. on pp. 13, 68).

Piro, V. C., Matschkowski, M., & Renard, B. Y. (2017). Metameta: Integrating metagenome analysis tools to improve taxonomic profiling. *Microbiome*, *5*(1), 1–11 (cit. on p. 42).

Piro, V. C., & Renard, B. Y. (2023). Contamination detection and microbiome exploration with grimer. *GigaScience*, *12*, giad017 (cit. on p. 101).

Plesniarski, A., Siddik, A. B., & Su, R.-C. (2021). The microbiome as a key regulator of female genital tract barrier function. *Frontiers in Cellular and Infection Microbiology*, *11*, 1292 (cit. on p. 5).

Portik, D. M., Brown, C. T., & Pierce-Ward, N. T. (2022). Evaluation of taxonomic classification and profiling methods for long-read shotgun metagenomic sequencing datasets. *BMC bioinformatics*, *23*(1), 541 (cit. on pp. 8, 11, 12, 85, 104).

Pullen, M. F., Boulware, D. R., Sreevatsan, S., & Bazira, J. (2019). Tuberculosis at the animal–human interface in the ugandan cattle corridor using a third-generation sequencing platform: A cross-sectional analysis study. *BMJ open*, *9*(4), e024221 (cit. on p. 97).

Putze, F., Sanders, P., & Singler, J. (2010). Cache-, hash-, and space-efficient bloom filters. *Journal of Experimental Algorithmics (JEA)*, *14*, 4–4 (cit. on pp. 16, 17).

Quick, J., Ashton, P., Calus, S., Chatt, C., Gossain, S., Hawker, J., Nair, S., Neal, K., Nye, K., Peters, T., et al. (2015). Rapid draft sequencing and real-time nanopore sequencing in a hospital outbreak of salmonella. *Genome biology*, *16*(1), 1–14 (cit. on p. 97).

Quick, J., Grubaugh, N. D., Pullan, S. T., Claro, I. M., Smith, A. D., Gangavarapu, K., Oliveira, G., Robles-Sikisaka, R., Rogers, T. F., Beutler, N. A., et al. (2017). Multiplex pcr method for minion and illumina sequencing of zika and other virus genomes directly from clinical samples. *Nature protocols*, *12*(6), 1261–1276 (cit. on p. 103).

Quick, J., Loman, N. J., Duraffour, S., Simpson, J. T., Severi, E., Cowley, L., Bore, J. A., Koundouno, R., Dudas, G., Mikhail, A., et al. (2016). Real-time, portable genome sequencing for ebola surveillance. *Nature*, *530*(7589), 228–232 (cit. on pp. 8, 22, 103).

Rang, F. J., Kloosterman, W. P., & de Ridder, J. (2018). From squiggle to basepair: Computational approaches for improving nanopore sequencing read accuracy. *Genome biology*, *19*(1), 1–11 (cit. on pp. 9, 22).

Reinert, K., Dadi, T. H., Ehrhardt, M., Hauswedell, H., Mehringer, S., Rahn, R., Kim, J., Pockrandt, C., Winkler, J., Siragusa, E., et al. (2017). The seqan c++ template library for efficient sequence analysis: A resource for programmers. *Journal of biotechnology*, *261*, 157–168 (cit. on p. 69).

Reppell, M., & Novembre, J. (2018). Using pseudoalignment and base quality to accurately quantify microbial community composition. *PLoS computational biology*, *14*(4), e1006096 (cit. on p. 14).

Rhie, A., Nurk, S., Cechova, M., Hoyt, S. J., Taylor, D. J., Altemose, N., Hook, P. W., Koren, S., Rautiainen, M., Alexandrov, I. A., et al. (2023). The complete sequence of a human y chromosome. *Nature*, 1–11 (cit. on p. 8).

Riiser, A. (2015). The human microbiome, asthma, and allergy. *Allergy, Asthma & Clinical Immunology*, *11*(1), 1–7 (cit. on p. 5).

Robbins, G., Tripathy, V. M., Misra, V. N., Mohanty, R. K., Shinde, V. S., Gray, K. M., & Schug, M. D. (2009). Ancient skeletal evidence for leprosy in india (2000 bc). *PloS one*, *4*(5), e5669 (cit. on p. 4).

Roberts, M., Hayes, W., Hunt, B. R., Mount, S. M., & Yorke, J. A. (2004). Reducing storage requirements for biological sequence comparison. *Bioinformatics*, *20*(18), 3363–3369 (cit. on p. 14).

Rodríguez-Beltrán, J., DelaFuente, J., Leon-Sampedro, R., MacLean, R. C., & San Millan, A. (2021). Beyond horizontal gene transfer: The role of plasmids in bacterial evolution. *Nature Reviews Microbiology*, *19*(6), 347–359 (cit. on p. 46).

Rogozin, I., Makarova, K., Natale, D., Spiridonov, A., Tatusov, R., Wolf, Y., Yin, J., & Koonin, E. (2002). Congruent evolution of different classes of non-coding dna in prokaryotic genomes. *Nucleic acids research*, *30*, 4264–71 (cit. on p. 6).

Rosenberg, N., & Nordborg, M. (2002). Genealogical trees, coalescent theory and the analysis of genetic polymorphisms. *Nature reviews. Genetics*, *3*, 380–90. https://doi.org/10.1038/nrg795 (cit. on p. 7)

Rothberg, J. M., Hinz, W., Rearick, T. M., Schultz, J., Mileski, W., Davey, M., Leamon, J. H., Johnson, K., Milgrew, M. J., Edwards, M., et al. (2011). An integrated semiconductor device enabling non-optical genome sequencing. *Nature*, *475*(7356), 348–352 (cit. on p. 7).

Roux, S., Matthijnssens, J., & Dutilh, B. (2019). Metagenomics in virology. https://doi.org/10.1016/B978-0-12-809633-8.20957-6. (Cit. on p. 11)

Runtuwene, L. R., Tuda, J. S. B., Mongan, A. E., & Suzuki, Y. (2019). On-site minion sequencing. In Y. Suzuki (Ed.), *Single molecule and single cell sequencing* (pp. 143–150). Springer Singapore. https://doi.org/10.1007/978-981-13-6037-4_10. (Cit. on pp. 22, 97)

Sanger, F., Nicklen, S., & Coulson, A. (1978). Dna sequencing with chain terminating inhibitors. *Proceedings of the National Academy of Sciences of the United States of America*, *74*, 5463–7. https://doi.org/10.1073/pnas.74.12.5463 (cit. on p. 7)

Sanz, J. L. (2011). Microorganism. In M. Gargaud, R. Amils, J. C. Quintanilla, H. J. (Cleaves, W. M. Irvine, D. L. Pinti, & M. Viso (Eds.), *Encyclopedia of astrobiology* (pp. 1061–1061). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-11274-4_992. (Cit. on p. 3)

Schaeffer, L., Pimentel, H., Bray, N., Melsted, P., & Pachter, L. (2017). Pseudoalignment for metagenomic read assignment. *Bioinformatics*, *33*(14), 2082–2088 (cit. on p. 14).

Schmartz, G. P., Hartung, A., Hirsch, P., Kern, F., Fehlmann, T., Müller, R., & Keller, A. (2022). Plsdb: Advancing a comprehensive database of bacterial plasmids. *Nucleic Acids Research*, *50*(D1), D273–D278 (cit. on p. 66).

Sczyrba, A., Hofmann, P., Belmann, P., Koslicki, D., Janssen, S., Dröge, J., Gregor, I., Majda, S., Fiedler, J., Dahms, E., Bremges, A., Fritz, A., Garrido-Oter, R., Jørgensen, T., Shapiro, N., Blood, P., Gurevich, A., Bai, Y., Turaev, D., & McHardy, A. (2017). Critical assessment of metagenome interpretation - a benchmark of metagenomics software. *Nature Methods*, *14*. https://doi.org/10.1038/nmeth.4458 (cit. on pp. 11, 104)

Sellers, P. H. (1974). On the theory and computation of evolutionary distances. *SIAM Journal on Applied Mathematics*, *26*(4), 787–793 (cit. on p. 13).

Senanayake, A., Gamaarachchi, H., Herath, D., & Ragel, R. (2023). Deepselectnet: Deep neural network based selective sequencing for oxford nanopore sequencing. *BMC bioinformatics*, *24*(1), 31 (cit. on p. 46).

Sereika, M., Kirkegaard, R. H., Karst, S. M., Michaelsen, T. Y., Sørensen, E. A., Wollenberg, R. D., & Albertsen, M. (2022). Oxford nanopore r10. 4 long-read sequencing enables the generation of near-finished bacterial genomes from pure cultures and metagenomes without short-read or reference polishing. *Nature methods*, *19*(7), 823–826 (cit. on p. 97).

Shaw, J., & Yu, Y. W. (2022). Theory of local k-mer selection with applications to long-read alignment. *Bioinformatics*, *38*(20), 4659–4669 (cit. on p. 15).

Shen, W., Xiang, H., Huang, T., Tang, H., Peng, M., Cai, D., Hu, P., & Ren, H. (2023). Kmcp: Accurate metagenomic profiling of both prokaryotic and viral popula-

tions by pseudo-mapping. *Bioinformatics*, *39*(1), btac845 (cit. on pp. 68, 69, 81, 93, 114).

Shi, W., Friedman, A., & Baker, L. (2016). Nanopore sensing. *Analytical Chemistry*, *89*. https://doi.org/10.1021/acs.analchem.6b04260 (cit. on p. 10)

Sim, J., & Chapman, B. (2019). In-field whole genome sequencing using the minion nanopore sequencer to detect the presence of high-prized military targets. *Australian Journal of Forensic Sciences*, *51*(sup1), S86–S90 (cit. on p. 22).

Simon, H. Y., Siddle, K. J., Park, D. J., & Sabeti, P. C. (2019). Benchmarking metagenomics tools for taxonomic classification. *Cell*, *178*(4), 779–794 (cit. on pp. 12, 104).

Singh, A., Garg, S., Kaur, R., Batra, S., Kumar, N., & Zomaya, A. Y. (2020). Probabilistic data structures for big data analytics: A comprehensive review. *Knowledge-Based Systems*, *188*, 104987 (cit. on p. 15).

Skaletsky, H., Kuroda-Kawaguchi, T., Minx, P. J., Cordum, H. S., Hillier, L., Brown, L. G., Repping, S., Pyntikova, T., Ali, J., Bieri, T., et al. (2003). The male-specific region of the human y chromosome is a mosaic of discrete sequence classes. *Nature*, *423*(6942), 825–837 (cit. on p. 97).

Smith, T. P., Thomas, T. J., García-Carreras, B., Sal, S., Yvon-Durocher, G., Bell, T., & Pawar, S. (2019). Community-level respiration of prokaryotic microbes may rise with global warming. *Nature communications*, *10*(1), 5124 (cit. on p. 3).

Sneddon, A., Ravindran, A., Hein, N., Shirokikh, N. E., & Eyras, E. (2022). Real-time biochemical-free targeted sequencing of rna species with riser. *bioRxiv*, 2022–11 (cit. on p. 102).

Sogin, M. L., Morrison, H. G., Huber, J. A., Welch, D. M., Huse, S. M., Neal, P. R., Arrieta, J. M., & Herndl, G. J. (2006). Microbial diversity in the deep sea and the underexplored "rare biosphere". *Proceedings of the National Academy of Sciences*, *103*(32), 12115–12120 (cit. on p. 3).

Solomon, B., & Kingsford, C. (2016). Fast search of thousands of short-read sequencing experiments. *Nature biotechnology*, *34*(3), 300–302 (cit. on p. 16).

Solomon, B., & Kingsford, C. (2018). Improved search of large transcriptomic sequencing databases using split sequence bloom trees. *Journal of Computational Biology*, *25*(7), 755–765 (cit. on p. 69).

Sommer, D. D., Delcher, A. L., Salzberg, S. L., & Pop, M. (2007). Minimus: A fast, lightweight genome assembler. *BMC bioinformatics*, *8*(1), 1–11 (cit. on p. 14).

Stankiewicz, P., & Lupski, J. R. (2010). Structural variation in the human genome and its role in disease. *Annual review of medicine*, *61*, 437–455 (cit. on p. 97).

Stark, L. A. (2010). Beneficial microorganisms: Countering microbephobia. *CBE—Life Sciences Education*, *9*(4), 387–389 (cit. on p. 4).

Staudigel, H., Furnes, H., McLoughlin, N., Banerjee, N. R., Connell, L. B., & Templeton, A. (2008). 3.5 billion years of glass bioalteration: Volcanic rocks as a basis for microbial life? *Earth-Science Reviews*, *89*(3-4), 156–176 (cit. on p. 3).

Stoddart, D., Heron, A. J., Mikhailova, E., Maglia, G., & Bayley, H. (2009). Single-nucleotide discrimination in immobilized dna oligonucleotides with a biological nanopore. *Proceedings of the National Academy of Sciences*, *106*(19), 7702–7707 (cit. on pp. 8, 9).

Stohr, J. J., Verweij, J. J., Buiting, A. G., Rossen, J. W., & Kluytmans, J. A. (2020). Within-patient plasmid dynamics in klebsiella pneumoniae during an outbreak of a carbapenemase-producing klebsiella pneumoniae. *PLoS One*, *15*(5), e0233313 (cit. on p. 46).

Sui, H.-y., Weil, A. A., Nuwagira, E., Qadri, F., Ryan, E. T., Mezzari, M. P., Phipatanakul, W., & Lai, P. S. (2020). Impact of dna extraction method on variation in human and built environment microbial community and functional profiles assessed by shotgun metagenomics sequencing. *Frontiers in microbiology*, *11*, 953 (cit. on p. 95).

Sun, C., Harris, R. S., Chikhi, R., & Medvedev, P. (2018). Allsome sequence bloom trees. *Journal of Computational Biology*, *25*(5), 467–479 (cit. on p. 69).

Sun, Z., Huang, S., Zhang, M., Zhu, Q., Haiminen, N., Carrieri, A. P., Vázquez-Baeza, Y., Parida, L., Kim, H.-C., Knight, R., et al. (2021). Challenges in benchmarking metagenomic profilers. *Nature methods*, *18*(6), 618–626 (cit. on pp. 12, 85).

Tanimoto, T. T. (1958). Elementary mathematical theory of classification and prediction (cit. on p. 14).

Tausch, S. H., Loka, T. P., Schulze, J. M., Andrusch, A., Klenner, J., Dabrowski, P. W., Lindner, M. S., Nitsche, A., & Renard, B. Y. (2022). Patholive—real-time pathogen identification from metagenomic illumina datasets. *Life*, *12*(9), 1345 (cit. on p. 97).

Tausch, S. H., Strauch, B., Andrusch, A., Loka, T. P., Lindner, M. S., Nitsche, A., & Renard, B. Y. (2018). Livekraken—-real-time metagenomic classification of illumina data. *Bioinformatics*, *34*(21), 3750–3752 (cit. on p. 97).

Taylor, T. L., Volkening, J. D., DeJesus, E., Simmons, M., Dimitrov, K. M., Tillman, G. E., Suarez, D. L., & Afonso, C. L. (2019). Rapid, multiplexed, whole genome and plasmid sequencing of foodborne pathogens using long-read nanopore technology. *Scientific reports*, *9*(1), 1–11 (cit. on pp. 46, 63).

Tegtmeyer, N., Soltan Esmaeili, D., Sharafutdinov, I., Knorr, J., Naumann, M., Alter, T., & Backert, S. (2022). Importance of cortactin for efficient epithelial nf-κb activation by helicobacter pylori, salmonella enterica and pseudomonas aeruginosa, but not campylobacter spp. *European Journal of Microbiology and Immunology*, *11*(4), 95–103 (cit. on p. 47).

Tirumalai, R. S., Chan, K. C., Prieto, D. A., Issaq, H. J., Conrads, T. P., & Veenstra, T. D. (2003). Characterization of the low molecular weight human serum proteome* s. *Molecular & cellular proteomics*, *2*(10), 1096–1103 (cit. on p. 102).

Totomoch-Serra, A., Marquez, M., & Cervantes-Barragan, D. (2017). Sanger sequencing as a first-line approach for molecular diagnosis of andersen-tawil syndrome. *F1000Research*, *6*, 1016. https://doi.org/10.12688/f1000research.11610.1 (cit. on p. 7)

Truong, D. T., Franzosa, E. A., Tickle, T. L., Scholz, M., Weingart, G., Pasolli, E., Tett, A., Huttenhower, C., & Segata, N. (2015). Metaphlan2 for enhanced metagenomic taxonomic profiling. *Nature methods*, *12*(10), 902–903 (cit. on p. 68).

Ulrich, J.-U., Epping, L., Pilz, T., Walther, B., Stingl, K., Semmler, T., & Renard, B. Y. (2023). Nanopore adaptive sampling effectively enriches bacterial plasmids. *bioRxiv*. https://doi.org/10.1101/2022.10.03.510741 (cit. on pp. 19, 45)

Ulrich, J.-U., Lutfi, A., Rutzen, K., & Renard, B. Y. (2022). ReadBouncer: precise and scalable adaptive sampling for nanopore sequencing. *Bioinformatics*, *38*(Supplement_1), i153–i160. https://doi.org/10.1093/bioinformatics/btac223 (cit. on pp. 18, 21, 46–48, 95)

Ulrich, J.-U., & Renard, B. Y. (2023). Taxor: Fast and space-efficient taxonomic classification of long reads with hierarchical interleaved xor filters. *bioRxiv*, 2023–07. https://doi.org/10.1101/2023.07.20.549822 (cit. on pp. 20, 67)

Urban, L., Holzer, A., Baronas, J. J., Hall, M. B., Braeuninger-Weimer, P., Scherm, M. J., Kunz, D. J., Perera, S. N., Martin-Herranz, D. E., Tipper, E. T., et al. (2021). Freshwater monitoring by nanopore sequencing. *Elife*, *10*, e61504 (cit. on pp. 8, 103).

Valouev, A., Ichikawa, J., Tonthat, T., Stuart, J., Ranade, S., Peckham, H., Zeng, K., Malek, J. A., Costa, G., McKernan, K., et al. (2008). A high-resolution, nucleosome position map of c. elegans reveals a lack of universal sequence-dictated positioning. *Genome research*, *18*(7), 1051–1063 (cit. on p. 7).

Venter, J. C., Adams, M. D., Myers, E. W., Li, P. W., Mural, R. J., Sutton, G. G., Smith, H. O., Yandell, M., Evans, C. A., Holt, R. A., Gocayne, J. D., Amanatides, P., Ballew, R. M., Huson, D. H., Wortman, J. R., Zhang, Q., Kodira, C. D., Zheng, X. H., Chen, L., … Zhu, X. (2001). The sequence of the human genome.

Science, *291*(5507), 1304–1351. https://doi.org/10.1126/science.1058040 (cit. on p. 7)

Ventola, C. L. (2015). The antibiotic resistance crisis: Part 1: Causes and threats. *Pharmacy and therapeutics*, *40*(4), 277 (cit. on p. 4).

Videvall, E. (2018). Plasmodium parasites of birds have the most at-rich genes of eukaryotes. *Microbial genomics*, *4*(2) (cit. on p. 101).

Viehweger, A., Marquet, M., Hölzer, M., Dietze, N., Pletz, M. W., & Brandt, C. (2023). Nanopore-based enrichment of antimicrobial resistance genes–a case-based study. *GigaByte*, *2023* (cit. on pp. 46, 47).

Villarreal, L. P. (2004). Are viruses alive? *Scientific American*, *291*(6), 100–105 (cit. on p. 3).

Viode, A., van Zalm, P., Smolen, K. K., Fatou, B., Stevenson, D., Jha, M., Levy, O., Steen, J., Steen, H., & Network, I. (2023). A simple, time-and cost-effective, high-throughput depletion strategy for deep plasma proteomics. *Science advances*, *9*(13), eadf9717 (cit. on p. 102).

Vollger, M. R., Guitart, X., Dishuck, P. C., Mercuri, L., Harvey, W. T., Gershman, A., Diekhans, M., Sulovari, A., Munson, K. M., Lewis, A. P., et al. (2022). Segmental duplications and their variation in a complete human genome. *Science*, *376*(6588), eabj6965 (cit. on p. 97).

Wan, Y., Yang, L., Wang, J. X., Tham, C.-Y., de Sessions, P. F., & Cheng, A. (2023). Direct rna sequencing coupled with adaptive sampling enriches rnas of interest in the transcriptome (cit. on p. 102).

Ward, D. M., Ferris, M. J., Nold, S. C., & Bateson, M. M. (1998). A natural view of microbial biodiversity within hot spring cyanobacterial mat communities. *Microbiology and Molecular Biology Reviews*, *62*(4), 1353–1370 (cit. on p. 3).

Watson, J. D., & Crick, F. H. (1953). Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *Nature*, *171*, 737–738 (cit. on p. 6).

Weber, L. M., Saelens, W., Cannoodt, R., Soneson, C., Hapfelmeier, A., Gardner, P. P., Boulesteix, A.-L., Saeys, Y., & Robinson, M. D. (2019). Essential guidelines for computational method benchmarking. *Genome biology*, *20*, 1–12 (cit. on p. 104).

Weilguny, L., De Maio, N., Munro, R., Manser, C., Birney, E., Loose, M., & Goldman, N. (2023). Dynamic, adaptive sampling during nanopore sequencing using bayesian experimental design. *Nature Biotechnology*, 1–8 (cit. on p. 66).

Wenger, A. M., Peluso, P., Rowell, W. J., Chang, P.-C., Hall, R. J., Concepcion, G. T., Ebler, J., Fungtammasan, A., Kolesnikov, A., Olson, N. D., et al. (2019). Accurate circular consensus long-read sequencing improves variant detection and

assembly of a human genome. *Nature biotechnology*, *37*(10), 1155–1162 (cit. on p. 8).

Wick, R. R. (2019). Badread: Simulation of error-prone long reads. *Journal of Open Source Software*, *4*(36), 1316 (cit. on p. 78).

Wick, R. R., Judd, L. M., & Holt, K. E. (2019). Performance of neural network basecalling tools for oxford nanopore sequencing. *Genome biology*, *20*(1), 1–10 (cit. on p. 22).

Wick, R. R., Judd, L. M., Wyres, K. L., & Holt, K. E. (2021). Recovery of small plasmid sequences via oxford nanopore sequencing. *Microbial genomics*, *7*(8) (cit. on p. 46).

Wood, D. E., Lu, J., & Langmead, B. (2019). Improved metagenomic analysis with kraken 2. *Genome biology*, *20*, 1–13 (cit. on pp. 12, 14, 68, 69, 78, 81, 113).

Wood, D. E., & Salzberg, S. L. (2014). Kraken: Ultrafast metagenomic sequence classification using exact alignments. *Genome biology*, *15*(3), 1–12 (cit. on p. 14).

Wooley, J. C., Godzik, A., & Friedberg, I. (2010). A primer on metagenomics. *PLoS computational biology*, *6*(2), e1000667 (cit. on p. 5).

Wu, Z., Che, Y., Dang, C., Zhang, M., Zhang, X., Sun, Y., Li, X., Zhang, T., & Xia, Y. (2022). Nanopore-based long-read metagenomics uncover the resistome intrusion by antibiotic resistant bacteria from treated wastewater in receiving water body. *Water Research*, *226*, 119282 (cit. on p. 103).

Yang, J., Yang, F., Ren, L., Xiong, Z., Wu, Z., Dong, J., Sun, L., Zhang, T., Hu, Y., Du, J., et al. (2011). Unbiased parallel detection of viral pathogens in clinical samples by use of a metagenomic approach. *Journal of clinical microbiology*, *49*(10), 3463–3469 (cit. on p. 5).

Ye, C., Ma, Z. S., Cannon, C. H., Pop, M., & Yu, D. W. (2012). Exploiting sparseness in de novo genome assembly. *BMC bioinformatics*, *13*(6), 1–8 (cit. on p. 14).

Yu, L. C.-H. (2018). Microbiota dysbiosis and barrier dysfunction in inflammatory bowel disease and colorectal cancers: Exploring a common ground hypothesis. *Journal of biomedical science*, *25*(1), 1–14 (cit. on p. 5).

Zhou, M., Wu, Y., Kudinha, T., Jia, P., Wang, L., Xu, Y., & Yang, Q. (2021). Comprehensive pathogen identification, antibiotic resistance, and virulence genes prediction directly from simulated blood samples and positive blood cultures by nanopore metagenomic sequencing. *Frontiers in genetics*, *12*, 244 (cit. on p. 42).

Zielezinski, A., Vinga, S., Almeida, J., & Karlowski, W. M. (2017). Alignment-free sequence comparison: Benefits, applications, and tools. *Genome biology*, *18*, 1–17 (cit. on p. 101).

Zusammenfassung

Die metagenomische Sequenzierung erlaubt es sämtliche genetische Information aller Organismen in einer komplizierten Probe zu erhalten. Diese Methode ermöglicht sowohl die Identifikation von Krankheitserregern in klinischen Proben als auch die Untersuchung mikrobieller Diversität in verschiedensten Lebensräumen. Durch die Entwicklung der Nanopore-Sequenzierung haben sich viele neue Möglichkeiten für Mikrobiologen eröffnet. Besonders die Portabilität der kleinen Nanopore-Sequenziergeräte und die Möglichkeit gezielt bestimmte DNA-Moleküle zu sequenzieren, haben das Forschungsfeld enorm verändert. Die Anwendung dieser beiden Möglichkeiten erfordert jedoch speichereffiziente Algorithmen, die eine Echtzeitanalyse auf herkömmlichen Laptops ermöglichen. In der vorliegenden Arbeit präsentiere ich neue Methoden zur Echtzeitanalyse von Nanopore-Sequenzierdaten im metagenomischen Kontext. Diese Methoden basieren auf optimierten algorithmischen Ansätzen zum Vergleich der Sequenzierdaten mit großen Referenzsequenz-Datenbanken. Das Hauptziel der Arbeit ist es die Sequenzierung und Analyse von unterrepräsentierten Organismen in metagenomischen Proben zu verbessern und Analysen direkt am Ort der Probennahme zu ermöglichen, wo es kaum Zugang zu leistungsstarken Servern oder Internet gibt.
Zuerst präsentiere ich ReadBouncer, ein neues Tool zur Anwendung von Nanopore Adaptive Sampling, das es ermöglicht die Sequenzierung uninteressanter DNA-Moleküle abzubrechen. ReadBouncer verbessert nicht nur die Klassifikation der sequenzierten Fragmente im Vergleich zu anderen Tools, es verringert auch den Speicherbedarf. Diese Verbesserungen ermöglichen eine bessere Anreicherung der unterrepräsentierten DNA und die Anwendung von Adaptive Sampling am Ort der Probennahme. Im weiteren zeige ich wie Adaptive Sampling nicht nur Wirts-DNA verringern kann, sondern auch unterrepäsentierte Plasmide in bakteriellen Proben anreichert. Diese Plasmide spielen eine entscheidende Rolle bei der Verbreitung von Antibiotikaresistenzen, sind aber nur mit Hilfe teurer und zeitintensiver Laborprotokolle charakterisierbar. Hier beschreibe ich Adaptive Sampling als kostengünstige Methode zur Anreicherung von Plasmiden, die einen entscheidenden Beitrag für die Sequenzierung und Charakterisierung von bakteriellen Krankheitserregern darstellen kann. Schlussendlich präsentiere ich eine speichereffiziente Methode für die taxonomische Echtzeit-Charakterisierung von Nanopore Sequenzierdaten, die ich in einem Tool namens Taxor implementiert habe. Taxor verbessert die taxonomische Klassifikation im Vergleich zu ähnlichen Tools und reduziert den Speicherbedarf dabei erheblich. Der resultierende Datenbank-Index für tausende mikrobielle Referenzgenome ist klein genug um in den Hauptspeicher eines üblichen Laptops zu passen und ermöglicht somit eine metagenomische Echtzeitanalyse am Ort der Probennahme.

Selbstständigkeitserklärung

Name: Ulrich
Vorname: Jens-Uwe

Ich erkläre gegenüber der Freien Universität Berlin, dass ich die vorliegende Dissertation selbstständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Die vorliegende Arbeit ist frei von Plagiaten. Alle Ausführungen, die wörtlich oder inhaltlich aus anderen Schriften entnommen sind, habe ich als solche kenntlich gemacht. Diese Dissertation wurde in gleicher oder ähnlicher Form noch in keinem früheren Promotionsverfahren eingereicht.

Mit einer Prüfung meiner Arbeit durch ein Plagiatsprüfungsprogramm erkläre ich mich einverstanden.

Jens-Uwe Ulrich, Berlin, 19. September 2023