Algorithms and computational methods for transcriptome analysis

Laura H. Tung

CMU-CB-22-108

October 2022

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Carl Kingsford, Chair
Guillaume Marçais
James Faeder
Rob Patro

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE1745016 to Laura H. Tung (Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation); the T32 training grant of the US National Institutes of Health as part of the HHMI-NIBIB Interfaces Initiative under award number T32 EB009403 to Laura H. Tung; the US National Science Foundation under Grant No. DBI1937540 to Carl Kingsford; the Gordon and Betty Moore Foundation's Data-Driven Discovery Initiative through Grant GBMF4554 to Carl Kingsford; and the National Institute of General Medical Sciences (NIGMS) of the US National Institutes of Health under award number R01GM122935. The department specifically disclaims responsibility for any analyses, interpretations, or conclusions. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, donors or the U.S. Government.

Keywords: Transcript assembly, Third-generation sequencing, Long read, RNA-seq, Representative set selection, Genomics, Error correction, Deep learning, Neural network, Transcriptome analysis.

To my parents.

Abstract

Studying the transcriptome is crucial to understanding functional elements of the genome and elucidating biological pathways associated with disease. High-throughput sequencing such as RNA-seq has become a powerful tool for transcriptome analysis. Due to limited read lengths, identifying full-length transcripts from short reads remains challenging. As third-generation sequencing becomes increasingly important, single-molecule long reads have been used to improve transcriptome analyses such as isoform identification. However, not all single-molecule long reads represent full transcripts due to incomplete cDNA synthesis. This drives a need for transcript assembly on long reads. We developed a reference-based long-read transcript assembler, Scallop-LR, aiming to discover more novel isoforms. Analyzing a considerable number of RNA-seq long-read samples, we quantified the benefit of performing transcript assembly on long reads. We demonstrate that Scallop-LR identifies more known transcripts and potentially novel isoforms for the human transcriptome than Iso-Seq Analysis and StringTie, indicating that long-read transcript assembly by Scallop-LR can reveal a more complete human transcriptome. Nanopore sequencing has become a leading choice for long-read RNA-seq. However, Nanopore long reads have high error rates. For many non-model organisms without a high-quality reference, *de novo* (reference-free) error correction methods designed for RNA-seq long reads are needed. We developed a novel, error-profile-aware correction method, deepCorrRNA, for correcting RNA-seq long reads *de novo* using deep learning. deepCorrRNA combines a graph-based method and a deep neural network that incorporates the error profile related information systematically. We show that ML-based deepCorrRNA achieves comparable error-rate reductions to state-of-the-art ONT-specific isONcorrect. Across different organisms, deepCorrRNA demonstrates robust *de novo* error correction capability, which can benefit the transcriptome studies of non-model organisms. deepCorrRNA's method in principle is generalizable and may be applied to different technologies. To accelerate transcriptome analyses, RNA-seq analysis tools require comprehensive evaluation and parameter optimization. While the number of RNA-seq samples grows enormously at large sequence databases, most RNA-seq analysis tools are evaluated on limited RNA-seq samples. This leads to a need to select a representative subset from RNA-seq samples at large databases, which effectively summarizes the original collection of RNA-seq samples. We developed a novel hierarchical representative set selection method, to tackle the memory and runtime challenges in k-mer counting approaches for RNA-seq samples in a large database. We demonstrate that hierarchical representative set selection achieves summarization quality close to direct representative set selection, while largely reducing the runtime and memory usage, and substantially outperforms random sampling on the entire SRA set of human RNA-seq samples. The algorithms, methods, and analysis we have developed can be used to improve transcriptome analyses and further our understanding of complex transcriptomes.

Acknowledgments

I would like to express my deep gratitude to my advisor, Prof. Carl Kingsford, for his guidance, help, invaluable advice, and continuous support throughout the five years of my graduate study. Carl always encouraged me to think innovatively. When I joined the Kingsford group, Carl gave me a book, "Creativity, Inc." by Ed Catmull from Pixar Animation Studios. Behind the artistic work of Pixar, their seeking inspiration during setbacks resembles the adventure of research. Carl's mentoring echoes the spirit of this book and puts a lot of emphasis on creativity and finding inspiration. Carl gave me the independence to explore my interests and try out different ideas, and when I was stuck he gave me the inspirations and insights that I needed the most. His guidance and mentoring helped me in all aspects of my research; I cannot thank him enough for all the advice and help he has given to me.

I would like to thank the rest of my thesis committee, Prof. James Faeder, Dr. Guillaume Marçais, Prof. Rob Patro, and Prof. Christopher Langmead, for their great advice, insightful suggestions, and encouragement on my thesis research. I especially appreciate their spending time on my proposal and defense in their very busy schedules.

My special thanks go to Dr. Mingfu Shao for his help, guidance, and support on transcript assembly; it is my great pleasure to collaborate with Mingfu and to learn from him.

My special thanks also go to Dr. Guillaume Marçais for helping improve the speed of per-pair computation of k-mer similarity significantly and for always generously offering help.

I would also like to thank my fellow labmates, the Kingsford group members (both past and current), for their stimulating discussions and suggestions on my research projects, and feedback on my presentations and manuscripts during these five years.

I also want to thank our doctoral program manager Nicole Stenger for the help and support she has given to me in all these years.

I am thankful to my friends, both on campus and remote, for being supportive and caring, for their companionship, and for all the fun and laughs we have shared together.

Finally, I would like to thank my family, my parents and grandmother, for their constant support, encouragement, and love, and for always standing behind me and supporting me in spirit throughout my educational journey.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

More than 95% of human genes are alternatively spliced to generate multiple isoforms [73]. Gene regulation through alternative splicing can create different functions for a single gene and increase protein-coding capacity and proteomic diversity. Thus, studying the full transcriptome is crucial to understanding the functionality of the genome.

Transcriptome analysis is the study of the transcriptome under certain conditions or in specific cells using high-throughput methods. Transcriptomic strategies such as transcription profiling play an important role in biomedical research, including biomarker discovery, disease diagnosis/prognosis, risk assessment of new drugs or environmental chemicals, etc. Transcription profiling involves the quantification of gene expression of many genes in cells or tissue samples at the transcription level. Transcriptomic techniques and analyses are useful in identifying the functions of genes, finding the changes associated with the mutant phenotype, and identifying pathways responding to environmental factors [8]. Transcriptomic analyses are used for personalized medicine; the analyses deal with the classification of disease/tumor subtypes or stages, the mechanisms of pathogenesis, and outcome predictions on tumor prognosis and therapy response [8]. Along with the technology advancements, computational methods are needed to facilitate various analyses of transcriptomes.

In the past decade, high-throughput, short-read sequencing (next-generation sequencing, NGS) technologies such as RNA-seq have become powerful tools for the characterization, quantification, and analysis of the transcriptome. In short-read RNA-seq, mRNAs are extracted, fragmented, and reverse transcribed into cDNA fragments; the cDNA fragments are PCR-amplified and sequenced using high-throughput, short-read sequencing. However, due to limited read lengths in short-read RNA-seq, identifying full-length transcripts from short reads and assembling all spliced RNAs within a transcriptome remain challenging problems.

1.1 Third-generation sequencing

In recent years, third-generation sequencing technologies offered by Pacific Biosciences (PacBio) and Oxford Nanopore Technologies (ONT) produce sequences of full cDNA or RNA molecules, promising to improve isoform identification and reducing ambiguity in mapping reads [18]. Long reads offer various benefits such as covering the entire molecule in the majority of cases and de-

termining the allele from which the RNA molecule originated by identifying single nucleotide variations (SNVs) affecting each single RNA molecule [96]. Long reads are also able to capture gene structures accurately without annotation and identify novel splice patterns that are not found by short reads [18]. Long reads have been used for genome assembly and can be used to identify functional elements in genomes that are missed by short-read sequencing [43, 90, 117]. Hybrid sequencing combining long reads and short reads can improve isoform identification and transcriptome characterization [4, 109]. Hybrid genome assemblers taking advantages of both short and long reads have also been developed [2, 44, 110, 118]. Long reads are also useful in identifying novel long non-coding RNAs and fusion transcripts [105] and in studying specific disease-determinant genes [100]. RNA-seq long reads can improve transcriptome analyses such as isoform identification, characterization of complex transcriptional events, study of alternative splicing, and study of transcription initiation.

PacBio single-molecule real-time (SMRT) sequencing uses a template called SMRTbell that is a closed, single-stranded circular DNA created by ligating adaptors to both ends of a target double-stranded cDNA molecule (as shown in the top part of Figure 1.1). The sequencing is based on a SMRT Cell, a chip with consumable substrates comprising arrays of zero-mode waveguide (ZMW) nanostructures, and a SMRTbell diffuses into a sequencing unit ZMW on it. The real-time observation of a SMRT Cell is called a "movie" (a "movie" of light pulses). The light pulses corresponding to each ZMW can be interpreted as a sequence of bases that is called a continuous long read (CLR) [78]. The continuous long read (CLR) is the original polymerase read (obtained by reading a template with the DNA polymerase), and subreads are sequences generated by splitting the CLR by the adapters (a full-pass subread is flanked on both ends by adapters). PacBio's "ROI" ("Read of Insert", consensus reads) displays a higher quality than subreads. Circular Consensus Sequence (CCS) reads are a type of ROI and are generated by collapsing multiple subreads when ≥ 2 full-pass subreads are present (Figure 1.1).

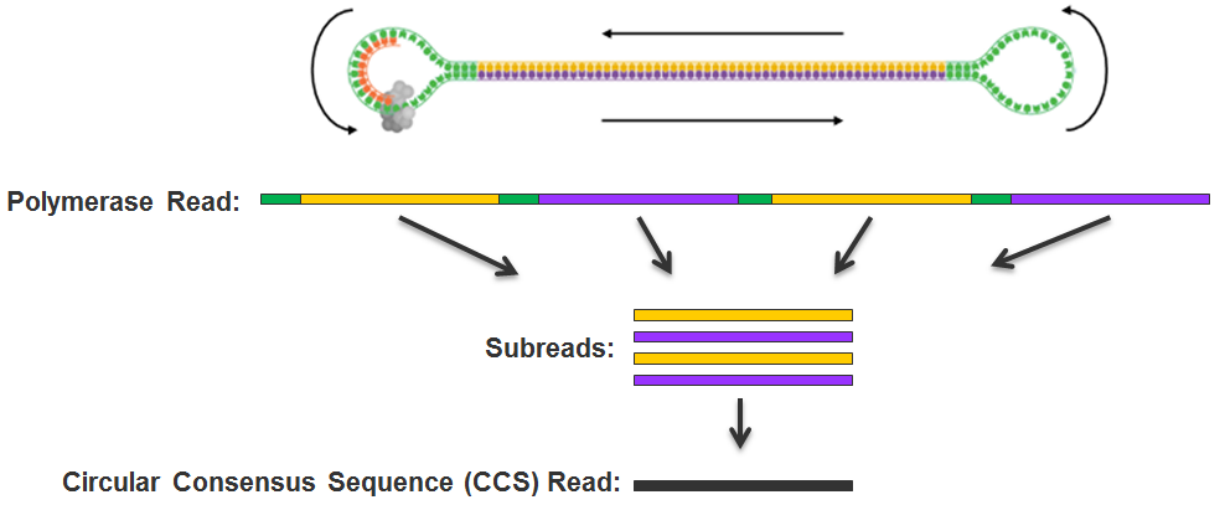


Figure 1.1: Illustration of PacBio SMRT sequencing (from Pacific Biosciences. Available at [accessed on 10/15/2022]: http://files.pacb.com/software/smrtanalysis/2.2.0/doc/smrtportal/help/!SSL!/Webhelp/Portal_PacBio_Glossary.htm).

For the SMRT sequencing, PacBio developed a software system, Iso-Seq Analysis [69], also called Iso-Seq informatics pipeline. Iso-Seq Analysis takes subreads as input and outputs polished isoforms (transcripts) through collapsing, clustering, consensus calling, etc.

Nanopore sequencing uses a nanoscale protein pore (nanopore) embedded in an electrically resistant polymer membrane inside the flow cell [108] (Figure 1.2). An ionic current is applied across the membrane driving the negatively charged, unwound single-stranded DNA or RNA molecules to move through the nanopore. Changes in the ionic current during translocation correspond to the nucleotide sequence and are measured continuously by an electronic chip. A basecaller converts the variations of the current to the DNA sequence. This enables real-time sequencing of single molecules.

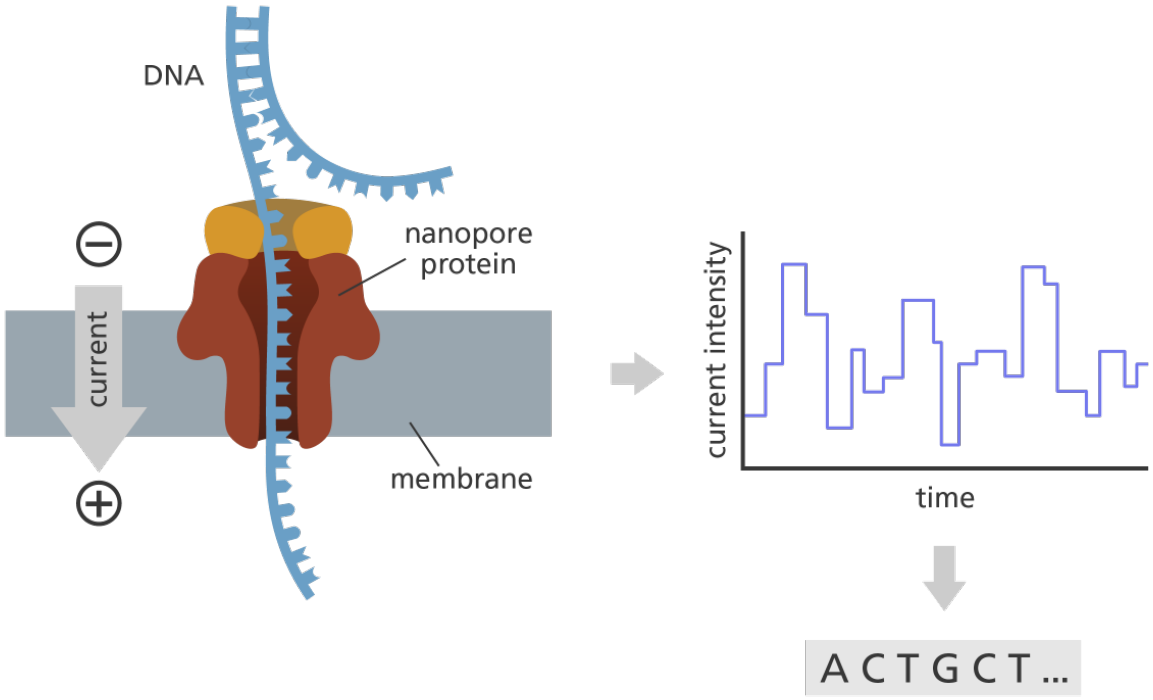


Figure 1.2: Illustration of nanopore sequencing (from yourgenome, Genome Research Limited, 2021, CC-BY 4.0. Available at [accessed on 10/15/2022]: https://www.yourgenome.org/facts/what-is-oxford-nanopore-technology-ont-sequencing/).

For RNA-seq, ONT has three categories of protocols: (1) cDNA sequencing by using full-length cDNA synthesis methods followed by PCR amplification (PCR-cDNA); (2) direct cDNA sequencing without PCR amplification (direct cDNA); (3) direct RNA sequencing that directly sequences native RNA molecules (direct RNA). PCR-cDNA has high throughput. Direct cDNA avoids PCR amplification bias but requires a large amount of input material. Direct RNA has lower accuracy than cDNA sequencing [108].

PacBio SMRT sequencing produces long reads with average lengths typically 30 kb (up to 40–70 kb) and the maximum length 200 kb. The read length in PacBio SMRT sequencing is

limited by the longevity of the polymerase [1]. Oxford Nanopore sequencing produces longer reads with read lengths typically 10–100 kb for long reads sequencing mode and 100–300 kb for ultra-long reads sequencing (up to 2.3 Mb [1]).

A main challenge associated with long-read technologies is that long reads generally have higher error rates than short reads. In PacBio sequencing, the error rate for subreads is $\sim$10–15%, and the error rate for CCS reads was previously reported to be $\sim$1–2%. However, the error rate of CCS reads has dramatically dropped in recent years—the CCS reads error rate is claimed to be reduced to $<$1% [1]. For ONT long reads, the error rates vary depending on the chemistry/kit/basecaller used and are up to $\sim$14% [56, 81]. The average accuracy of ONT sequencing is also improving steadily.

1.2 Transcript assembly

Transcript assembly reconstructs transcripts from RNA-seq reads. Transcript assembly is useful for differential gene expression analysis, novel gene discovery, novel isoform identification, etc. Transcript assemblers were initially designed for assembling RNA-seq short reads, as short reads usually do not span the full lengths of long transcripts. There are mainly two categories of approaches for transcript assembly: *de novo* transcript assembly and reference-based transcript assembly.

De novo transcript assembly generates contigs only based on the RNA-seq reads without using a reference genome. Most *de novo* transcript assemblers are based on de Bruijn graphs constructed from k-mer decompositions of the RNA-seq reads. The transcript is reconstructed by the overlap of the k-mers. The state-of-the-art *de novo* transcript assemblers for short reads include rnaSPAdes [12], Trinity [32], Trans-ABySS [79], Bridger [16], SOAPdenovo-Trans [115], and Oases [85]. A challenge in *de novo* transcript assembly is that it may produce fragmented transcripts due to sequencing errors, polymorphism, sequence repeats, etc.

Reference-based transcript assembly is more accurate than *de novo* transcript assembly but requires a reference genome. Reference-based transcript assembly first aligns the RNA-seq reads to the reference genome using splice-aware RNA-seq aligners, such as STAR [24], TopHat2 [38], HISAT2 [39], and BBMap [14] for short reads. From the alignments, exons and splice junctions are inferred and identified. Most reference-based transcript assemblers, such as short-read assemblers Cufflinks [99], Scripture [33], IsoLasso [55], Traph [98], Bayesembler [62], CIDANE [15], TransComb [58], StringTie [75], and Scallop [88], use the read alignments to build splice graphs (vertices are exons; edges are splice junctions), and then a set of paths are inferred from the splice graph as assembled transcripts. Scallop and StringTie use flow decomposition algorithms to decompose the splice graph to infer a small number of paths. StringTie generally outperforms Cufflinks, IsoLasso, Scripture, and Traph [75, 76]. Scallop generally outperforms StringTie and TransComb, and especially shows advantage in identifying lowly expressed transcripts [88].

In single-molecule long-read RNA-seq, although in the majority of cases long reads cover the entire cDNA molecule, cDNA synthesis can be incomplete with respect to the original mRNAs [89]. Consequently, a long read could correspond to a partial transcript as a result of incomplete cDNAs. Thus, not all single-molecule long reads represent full transcripts due to incomplete cDNA synthesis, and therefore transcript assembly is still needed for single-molecule long

reads in order to recover the original full transcripts.

Long-read transcript assembly deals with the special characteristics of long reads, such as long read lengths and high error rates. Scallop-LR [102] that we developed (Chapter 2) is the first long-read transcript assembler that is evaluated on real sequencing long reads. Scallop-LR is reference-based and is evolved from the short-read assembler Scallop. StringTie2 [45] is another reference-based long-read transcript assembler and is evolved from StringTie. For reference-based long-read transcript assembly, splice-aware long-read aligners including Minimap2 [52], Graphmap2 [63], and GMAP [112] can be used for aligning the RNA-seq long reads to the reference genome.

There are also two hybrid *de novo* transcript assemblers, IDP-denovo [28] and a new version of Trinity [32]. IDP-denovo and Trinity do not assemble long reads; they assemble short reads and use long reads to extend, supplement, or improve the assembly of short reads.

1.3 Long-read error correction

Long-read RNA-seq has become increasingly useful in transcriptome studies, especially for isoform identification and quantification, fusion transcript detection, etc. However, long reads' higher error rates than short reads' can affect transcriptome analysis, especially in detecting exon boundaries, quantifying similar isoforms and paralogous genes, etc. This drives a need for computational methods for error correcting RNA-seq long reads.

Long reads have three types of sequencing errors: substitutions, insertions, and deletions. While errors in short reads are primarily substitutions, in long reads a large percentage of errors are indels (insertions and deletions). Short-read error correctors are not applicable to long reads since they were designed to deal with substitutions and low error rates. Error correction methods for genomic long reads have been developed in recent years. There are mainly three categories of approaches for genomic error correction: hybrid correction, self-correction, and signal-based correction.

Hybrid correction uses short reads to correct long reads. The representative hybrid genomic error correctors include HALC [6], LoRDEC [82], NaS [60], proovread [34], Nanocorr [31], and PBcR [42]. Most hybrid correctors use a mapping strategy; the mapping strategy places short reads on long reads and corrects long read regions using the related short read sequences. Hybrid correction requires using short reads, while oftentimes short reads are unavailable, which limits the application of hybrid error correctors.

Self-correction uses only long reads to correct the errors in long reads. The representative self-correction genomic error correctors include Canu's error correction module [43], LoRMA [83], MECAT [114], daccord [97], pbdagcon [17], and CONSENT [65]. Most self-correctors use graph-based consensus strategy; the graph-based consensus strategy is based on read overlapping to generate corrected consensus sequences using graphs.

Signal-based correction uses raw electric signals, such as the measurement of current in nanopore sequencing. The representative signal-based genomic error correctors include Nanopolish [59] and NanoReviser [106]. Signal-based correctors re-analyze the raw electrical signals outputted from the sequencer to find the correct bases. However, raw signal data may not always be available.

Using genomic error correctors to correct RNA-seq long reads has undesirable effects, such as splitting reads in low coverage regions or removing/adding exons [56]. Thus, error correction methods that are designed for RNA-seq long reads are needed. The earliest error corrector specifically designed for RNA-seq long reads is LSC [3] that is a hybrid corrector for PacBio and requires using short reads. Afterwards, two reference-based error correctors that are specifically designed for RNA-seq long reads, TranscriptClean [113] and FLAIR [94], were developed. TranscriptClean and FLAIR correct base errors and/or splice sites for RNA-seq long reads, but require using a reference genome. However, for many non-model organisms, a high-quality reference genome is unavailable. Hence, *de novo* error correction for RNA-seq long reads is needed.

De novo error correction corrects RNA-seq long reads without using a reference genome, transcriptome, or annotation. Recently, two *de novo* error correctors specifically designed for ONT RNA-seq long reads, isONcorrect [81] and RATTLE [22], have been developed. isONcorrect and RATTLE are graph-based self-correction methods that use partial order alignment (POA) [49] for multiple sequence alignment (MSA) and consensus generation.

1.4 Representative set selection

For transcriptome analysis, the aforementioned bioinformatics tools including transcript assemblers, RNA-seq read aligners, RNA-seq read error correctors, etc. require comprehensive evaluation prior to being used in the study of the transcriptome. However, despite numerous RNA-seq samples available at large sequence databases such as the Sequence Read Archive (SRA) [51], most RNA-seq analysis tools are evaluated on a limited number of RNA-seq samples; thus, they may not be sufficiently evaluated by samples across a variety of cell/tissue types and disease conditions. This drives a need for computational methods to select a representative subset from a large collection of RNA-seq samples to facilitate comprehensive, unbiased evaluation of bioinformatics tools. Furthermore, using a selection of representative RNA-seq samples can benefit the parameter optimization of bioinformatics tools.

Representative set selection solves the problem of finding a subset of data points (representatives) that efficiently summarizes and describes the original collection of data. Various representative set selection methods have been developed in the fields of computer vision [26], signal/image processing [107], information retrieval [27, 72], and machine learning [25, 29]. Most common applications of representative set selection include image, video, and text summarizations. Machine learning tasks such as classification and regression can also improve in terms of fast training and reduced memory usage by using a representative subset as the training set [25, 29].

One category of representative set selection methods is clustering-based approaches [21, 27]. Clustering-based approaches first cluster the data points using a clustering method such as k-means, k-medoids, spectral clustering, or DBSCAN (Density-Based Spatial Clustering of Applications with Noise); the representative objects are then selected from the clusters. Usually the number of clusters is set to be the representative set size, and one representative object is selected from each cluster, such as the object closest to the mean or just the medoid of the cluster.

Another category of representative set selection methods is sparse modeling-based algorithms [26, 107]. Sparse modeling algorithms formulate the representative set selection as a

dictionary learning problem, based on the assumption that the entire set can be reconstructed by linear combinations of dictionary items. Additionally, there are also Rank Revealing QR algorithms [9] that use matrix factorization to find a subset of columns of the data matrix corresponding to the best conditioned submatrix, and algorithms using mutual information and relative entropy to search for a representative subset [72].

The representative subset selected by the aforementioned approaches generally follows the density distribution of the original full set, which is useful for video/photo summarizations. However, for RNA-seq analyses, the representative set having even coverage of the transcriptional space spanned by the original set is more important, such that rare cell types or conditions can be sufficiently represented. Thus, the above methods are generally not suitable for RNA-seq analyses.

In the field of RNA-seq analysis, a geometric sketching algorithm [35] was developed for single-cell RNA-seq. To accelerate single-cell analysis, geometric sketching summarizes the transcriptomic heterogeneity within a data set using a representative subset of cells. Using a covering algorithm, geometric sketching selects the representative subset that has even coverage of the transcriptional space spanned by the original set, so that rare cell types are sufficiently represented.

Recently, a Python package, apricot [84], was developed for selecting representative subsets using submodular optimization. Apricot maximizes a monotone submodular function's value to find a representative subset. Using facility location, apricot maximizes the sum of similarities between each sample and its closest representative sample. The representative set selected by apricot approximately evenly spans the space of the original data. Thus, apricot is well suited for RNA-seq analyses.

1.5 Contributions

This dissertation combines algorithms and computational methods development as well as data analysis for transcriptome study. Our contributions are summarized as follows:

- **Transcript assembly on single-molecule long reads** (Chapter 2). Since not all single-molecule long reads represent full transcripts due to incomplete cDNA synthesis, we developed the first reference-based long-read transcript assembler called Scallop-LR, evolved from Scallop [88]. Scallop-LR's algorithms are tailored to long-read technologies, dealing with the long read lengths (by representing long reads as long phasing paths and preserving phasing paths in assembly) and high error rates, and taking advantage of long-read-specific features such as the read boundary information to construct more accurate splice graphs. A post-assembly clustering algorithm is added in Scallop-LR to reduce false negatives. We analyzed 26 PacBio long-read data sets from the Sequence Read Archive (SRA) [51] with Scallop-LR, Iso-Seq Analysis [69], and StringTie [75], and quantified the benefit of performing transcript assembly on single-molecule long reads. Using combined multiple evaluation methods, we demonstrate that Scallop-LR is able to identify many more known transcripts and find more potential novel isoforms than Iso-Seq Analysis. We also show that Scallop-LR assembles more known transcripts and potential novel isoforms than StringTie for human samples. Our results indicate that long-read transcript assembly with

Scallop-LR can help reveal a more complete human transcriptome.

- **Representative set selection of RNA-seq samples** (Chapter 3). To facilitate comprehensive, unbiased evaluation and robust parameter optimization of bioinformatics tools for RNA-seq analyses, we developed a novel method to select a representative subset from all available RNA-seq samples at a large sequence database. We designed a sequence-based, k-mer counting approach that selects a representative subset based on k-mer similarities between RNA-seq samples. Because of the large numbers of available RNA-seq samples and of k-mers in each sample, computing the full similarity matrix using k-mers for the entire set of RNA-seq samples in a large database has memory and runtime challenges, making direct representative set selection infeasible. To tackle this challenge, we developed a novel computational method called "hierarchical representative set selection." Hierarchical representative set selection is an algorithm that breaks representative set selection into sub-selections and hierarchically selects representative samples through multiple levels. We developed a novel seeded-chunking method along with a weighting scheme. We demonstrate that hierarchical representative set selection achieves summarization quality close to direct representative set selection, while largely reducing runtime and memory requirements of computing the full similarity matrix. We show that hierarchical representative set selection substantially outperforms random sampling on the entire SRA set of human RNA-seq samples, making it a practical solution to representative set selection on large databases. Our method is the first approach to this problem that can scale to collections of the size of the full set of human RNA-seq samples in the SRA.
- ***De novo* error correction for RNA-seq long reads** (Chapter 4). To reduce sequencing errors of RNA-seq long reads for many non-model organisms without a high-quality reference, we developed a novel *de novo* (reference-free) error correction method for RNA-seq long reads. While prior findings suggest that the error profile information (i.e. factors associated with error occurrences or error types) is useful for error correction, existing *de novo* self-correction error correctors for Nanopore RNA-seq reads have not systematically taken into account the error profile information. Thus, we developed a novel error correction method called deepCorrRNA for self-correcting RNA-seq long reads *de novo* using deep learning. deepCorrRNA is an error-profile-aware, generalizable correction method that combines a graph-based MSA-POA and a Bi-LSTMs/LSTMs-based deep neural network that incorporates the error profile related information systematically. deepCorrRNA is the first ML-based method for *de novo* RNA-seq long-read error correction and is the first *de novo* RNA-seq long-read correction method that incorporates the error profile related information systematically. We evaluated deepCorrRNA on five ONT RNA-seq data sets of three organisms. Our results show that ML-based deepCorrRNA achieves similar reductions in the total error rates to state-of-the-art ONT-specific RNA-seq error corrector isONcorrect [81]. While the model of deepCorrRNA is trained with human data, we show that its correction performance on different organisms (mouse and Drosophila) demonstrates its transferability and robust *de novo* error correction capability, which can benefit the transcriptome analysis of non-model organisms. Using generalized (technology/organism independent) error profile related features systematically and with comparable error reductions to the state of the art, deepCorrRNA offers a generalizable method that may be

applied to different technologies.

Together, these contributions provide new algorithms and computational methods to improve transcriptome analysis and advance our understanding of complex transcriptomes.

Chapter 2

Transcript assembly on single-molecule long reads

A version of this chapter was published in *Genome Biology* [102] and is joint work with Mingfu Shao and Carl Kingsford.

2.1 Background

Single-molecule long reads have been widely used for genome assembly [43, 90, 117]. Long reads have advantages in genome assembly as they span long repeats, polymorphic regions, transposable elements, etc. and can be used to identify functional elements in genomes that are missed by short-read sequencing. To take advantages of both short reads and long reads, hybrid genome assemblers have been developed [2, 44, 110, 118]. For transcriptome studies, single-molecule long reads can improve isoform identification [1], identify novel splice patterns that are not found by short reads [18], and capture gene structures without annotation. Long reads are useful in identifying fusion transcripts [105], investigating disease-determinant genes [100], studying transcription initiation, and characterizing complex transcriptional events. Hybrid sequencing combining long reads and short reads can improve isoform identification and transcriptome characterization [4, 109].

We focus on transcript assembly of long reads, aiming to discover more novel isoforms. Although it is often thought that long reads are full-length transcripts and isoforms with no assembly required [70], in fact the success rate of the sequenced cDNA molecules containing all splice sites of the original transcripts depends on the completeness of cDNA synthesis [89]. Sharon et al. [89] found that a CCS read could correspond to an incomplete transcript as a consequence of incomplete cDNA synthesis, although a CCS read represents the full cDNA molecule. They found that, in their experiment, for transcripts > 2.5 kb, full-length reads that represent the original transcripts are less likely to be observed than those for transcripts < 2.5 kb. Tilgner et al. [96] also found that, in their experiment, reads representing all splice sites of the original transcripts are more likely to be observed for transcripts $\leq$ 3 kb. The cDNA synthesis methods impose limitations on long reads [47] even though with increasing performance the sequencing technologies can be capable of sequencing long full-length transcripts. In addition, long reads may still be

limited by the sequencing length limit of the platform [78]. Thus, incomplete cDNA synthesis plus the sequencing length limit could cause PacBio's consensus long reads to miss a substantial number of true transcripts [78], especially longer transcripts. This suggests that the transcript assembly of long reads is still needed, since it is possible that those CCS reads corresponding to incomplete transcripts could be assembled together to recover the original full transcripts.

Long read lengths and high error rates pose computational challenges to transcript assembly. No published transcript assembler had been adapted and systematically tested on the challenges of long-read transcript assembly yet when the work in this chapter was first published. Aiming to handle these challenges, we developed a reference-based long-read transcript assembler called Scallop-LR, evolved from Scallop, an accurate short-read transcript assembler [88]. Scallop-LR is designed for PacBio long reads. Scallop-LR's algorithms are tailored to long-read technologies, dealing with the long read lengths and high error rates as well as taking advantage of long-read-specific features such as the read boundary information to construct more accurate splice graphs. A post-assembly clustering algorithm is also added in Scallop-LR to reduce false negatives.

We analyzed 26 long-read datasets from NIH's Sequence Read Archive (SRA) [51] with Scallop-LR, Iso-Seq Analysis [69], and StringTie [75, 76]. Iso-Seq Analysis, also known as Iso-Seq informatics pipeline, is a software system developed by PacBio that takes subreads as input and outputs polished isoforms (transcripts) through collapsing, clustering, consensus calling, etc. Iso-Seq Analysis does not perform assembly per se. The clustering algorithm in Iso-Seq Analysis clusters reads based on their isoform of origin. StringTie was originally designed as a short-read transcript assembler but can also assemble long reads. StringTie outperforms many leading short-read transcript assemblers [75].

Through combined evaluation methods, we demonstrate that Scallop-LR is able to find more known transcripts and novel isoforms that are missed by Iso-Seq Analysis. We show that Scallop-LR can identify 2100–4000 more known transcripts (in each of 18 human datasets) or 1100–2200 more known transcripts (in each of eight mouse datasets) than Iso-Seq Analysis. The sensitivity of Scallop-LR is 1.33–1.71 times higher (for the human datasets) or 1.43–1.72 times higher (for the mouse datasets) than that of Iso-Seq Analysis. Scallop-LR also finds 2.53–4.23 times more (for the human datasets) or 2.38–4.36 times more (for the mouse datasets) potential novel isoforms than Iso-Seq Analysis. Further, Scallop-LR assembles 950–3770 more known transcripts and 1.37–2.47 times more potential novel isoforms than StringTie, and has 1.14–1.42 times higher sensitivity than StringTie for the human datasets.

2.2 Methods

2.2.1 Scallop-LR algorithms for long-read transcript assembly

Scallop-LR is a reference-based transcript assembler that follows the standard paradigm of alignment and splice graphs but has a computational formulation dealing with "phasing paths." "Phasing paths" are a set of paths that carry the phasing information derived from the reads spanning more than two exons. The reads are first aligned to a reference genome and the alignments are transformed into splice graphs, in which vertices are inferred (partial) exons, edges are splice

junctions, the coverage of exon is taken as the vertex weight, and the abundance of splice junction is used as the edge weight. We decompose the splice graph to infer a small number of paths (i.e. predicted transcripts) that cover the topology and fit the weights of the splice graph.

Scallop-LR represents long reads as long phasing paths, preserved in assembly

Unlike short reads, most long reads span more than two exons. Thus, if the multi-exon paths of long reads are broken when decomposing splice graphs (which is more likely to occur since the majority of long reads span large numbers of exons), many long reads would not be correctly covered by assembled transcripts. Thus, Scallop-LR represents long reads as long phasing paths and preserves phasing paths in assembly. This is particularly important since we want every phasing path (and thus every long read) to be covered by some transcript so that the assembly can represent the original mRNAs. Scallop-LR adapted the phasing-path preservation algorithm from Scallop when decomposing splice graphs into transcripts. The Scallop algorithm uses an iterative strategy to gradually decompose the splice graph while achieving three objectives simultaneously:

a) Preserving all phasing paths in assembled transcripts when decomposing the splice graph;

b) Minimizing the read coverage deviation using linear programming;

c) Minimizing the number of predicted transcripts by reducing an upper bound on the number of required paths.

Figure 2.1 shows a simple example of a splice graph by representing long reads as phasing paths and its decomposition without and with preservation of long reads' phasing paths. The example illustrates that when decomposing the splice graph without preserving long reads' phasing paths, the multi-exon paths of some long reads are broken, and thus not all long reads are correctly covered by assembled transcripts. When decomposing the splice graph by preserving long reads' phasing paths, all long reads are correctly covered by assembled transcripts.

By representing long reads as long phasing paths, Scallop-LR makes full use of the information in long reads through phasing-path preservation, so that assembled transcripts can best represent the input long reads.

Additional Scallop-LR algorithms

To improve long-read assembly accuracy, Scallop-LR extracts the boundary information from long reads and identifies transcript boundaries to build a more accurate splice graph. In single-molecule sequencing, there are two types of long reads produced: full-length reads and non-full-length reads. Full-length reads are the reads that have a 5' primer, 3' primer, and polyA tail, which are the reads that represent full-length transcripts they originated from. Non-full-length reads do not represent full-length transcripts. We further classify non-full-length reads into two types: non-full-length boundary reads and non-full-length internal reads. Non-full-length boundary reads are the reads that either have a 5' primer but not the 3' primer, or have a 3' primer but not the 5' primer (i.e. reads that come from either the 5' or 3' end but do not reach the other end). Non-full-length internal reads are the reads that have neither of the 5' primer and 3' primer (i.e. reads that do not come from either end). Scallop-LR treats non-full-length internal

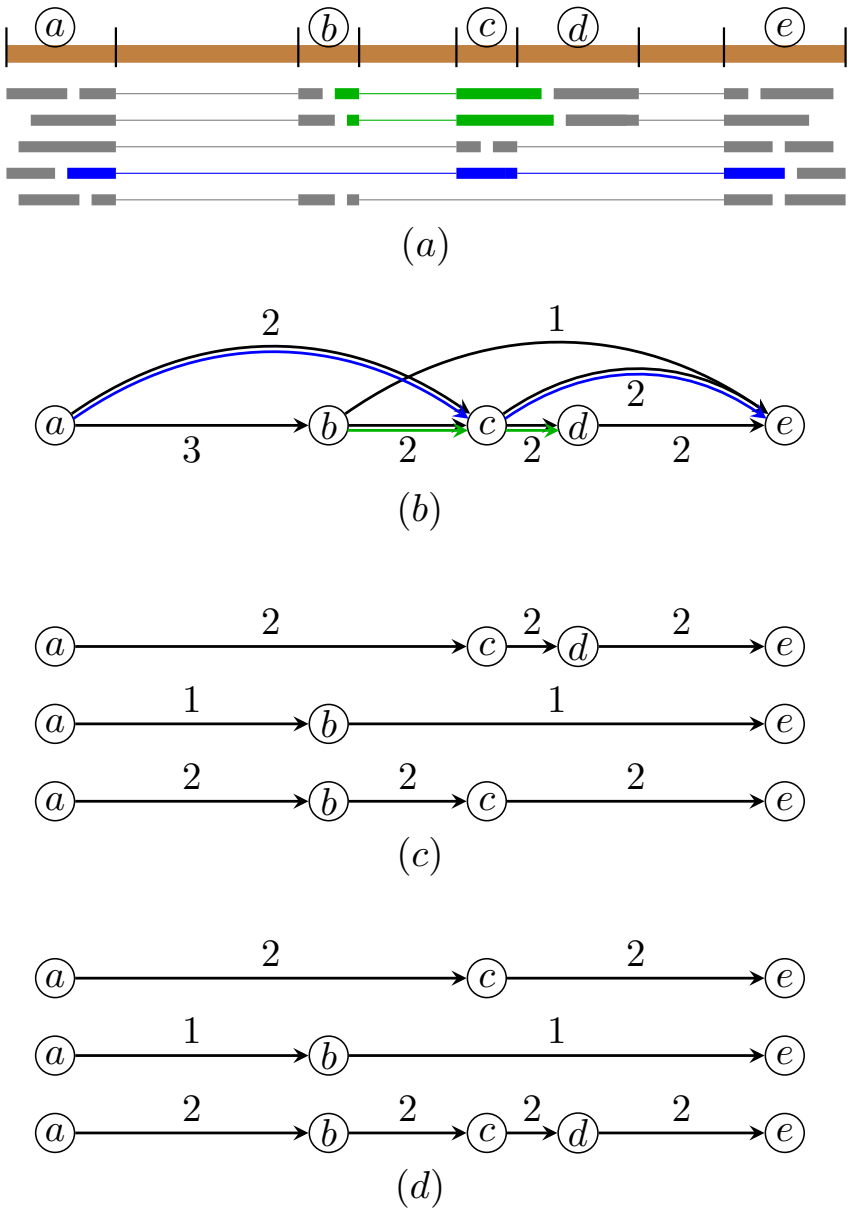


Figure 2.1: Example of a splice graph by representing long reads as phasing paths and its decomposition with and without preservation of long reads' phasing paths. (a) Alignment of reads to the reference genome. Inferred (partial) exons are marked with letters. Green and blue colored reads are long reads spanning more than two exons. Scallop-LR represents these long reads as a set of phasing paths: $\{(a, c, e), (b, c, d)\}$. (b) The corresponding splice graph (with weights for all edges) and associated phasing paths (in green and blue). (c) Decomposition of the splice graph without preservation of long reads' phasing paths. Although all weights are perfectly matched, both phasing paths are "broken" (none of the three decomposed paths contains (b, c, d) or (a, c, e)). (d) Decomposition of the splice graph with preservation of long reads' phasing paths. All phasing paths are correctly covered by assembled transcripts.

reads like short reads when constructing the splice graph.

We refer to non-full-length boundary reads (with one side boundary) and full-length reads (with two side boundaries) as "boundary reads" for the side they have a boundary. We use the *Classify* tool in Iso-Seq Analysis to obtain full-length and non-full-length CCS reads. The Scallop-LR algorithm extracts the boundary information of each read from the Classify results and uses it to deduce starting/ending boundaries in the splice graph. Specifically, when there are a certain number of boundary reads whose boundaries align within an exonic region in the genome with very similar boundary positions (the default minimum number is 3), the algorithm defines it as a starting or ending boundary:

> Suppose there are some 5' end boundary reads aligned to the genome at positions $[a+\delta_1, x_1]$, $[a+\delta_2, x_2]$, $[a+\delta_3, x_3]$, etc., where $|\delta_1|$, $|\delta_2|$, $|\delta_3|$, ... are within a predefined allowance of difference for matching positions and x_1, x_2, x_3, ... are the ending positions of the aligned genomic regions of these reads, then this is a signal that position a corresponds to a starting position of a transcript. Thus, in the splice graph, we add an edge connecting the source s to the vertex corresponding to the exonic region $[a, c]$ in the genome (where c is the ending position of this exonic region).

Similarly,

> Suppose there are some 3' end boundary reads aligned to the genome at positions $[x_1, b+\delta_1]$, $[x_2, b+\delta_2]$, $[x_3, b+\delta_3]$, etc., where $|\delta_1|$, $|\delta_2|$, $|\delta_3|$, ... are within a predefined allowance of difference for matching positions and x_1, x_2, x_3, ... are the starting positions of the aligned genomic regions of these reads, then this is a signal that position b corresponds to an ending position of a transcript. Thus, in the splice graph, we add an edge connecting the vertex corresponding to the exonic region $[d, b]$ in the genome (where d is the starting position of this exonic region) to the target t.

This is for the forward strand. For the reverse strand, the situation is opposite. Specifically, the algorithm first sorts all boundary positions from boundary reads together with splice positions. The algorithm identifies a new transcript boundary if the number of closely adjacent boundary positions of the same type (i.e. not separated by any different type of boundary or splice position in the sorted list) reaches a threshold (by default 3). For these closely adjacent boundary positions of the same type in the sorted list, if they are 5' boundary positions, the algorithm reports the leftmost one as the 5' transcript boundary coordinate. Similarly, if they are 3' boundary positions, the algorithm reports the rightmost one as the 3' transcript boundary coordinate.

To increase the precision of long-read assembly, Scallop-LR uses a post-assembly clustering algorithm to reduce the false negatives in the final predicted transcripts. For transcripts with very similar splice positions, the algorithm clusters them into a single transcript. "Very similar splice positions" means (a) these transcripts have the same number of splice positions; (b) for each splice position, their position differences are within a predefined allowance (the default allowance is 10 bp; the allowance can be set in a parameter). This allowance is for the sum of the difference (absolute value) of starting position and the difference of ending position for a splice position. We use a single-linkage clustering method to group the assembled transcripts. Specifically, we first build an undirected graph in which vertices represent all assembled transcripts. We iterate through all pairs of assembled transcripts, and if any two transcripts are "very similar" (i.e. all their splice positions' differences are less than a predefined allowance), we add an edge between these two transcripts (i.e. vertices). We then find all connected components in this graph; each

connected component is a cluster. For each cluster, we identify the transcript with the highest (predicted) abundance and use this transcript to represent this cluster. The abundance of this consensus transcript is then set to the sum of the abundances of all transcripts in this cluster. We modify this consensus transcript so it spans the transcripts in the cluster by extending the boundary positions of its two end-exons as needed: its left position is set to the leftmost position among all transcripts in the cluster; its right position is set to the rightmost position among all transcripts in the cluster. This clustering collapses "nearly redundant" transcripts and thus increases the precision of assembly.

The Scallop-LR algorithm deals with the high error rates in long reads when building the splice graph. Errors in long reads are mostly insertions and deletions, which may lead to mis-alignments around splice positions. When identifying splice positions from long-read alignments during the construction of the splice graph, the algorithm takes into account that a single insertion or deletion in the middle of the alignment may be caused by sequencing errors in long reads and therefore ignore these small indels (by treating them as alignment match and counting towards to the coverage of the corresponding vertex) when determining the splice positions. Moreover, long deletions due to sequencing errors may be falsely marked as splice junctions by aligners. Thus, Scallop-LR introduces a parameter (by default 50) as the minimum size of introns to filter out such false-negative splice junctions.

2.2.2 Combined evaluation methods

We use multiple transcript evaluation methods to examine the quality of predicted transcripts from transcript assemblers (i.e. Scallop-LR and StringTie) and Iso-Seq Analysis. The combined evaluation methods allow us to assess predicted transcripts using various metrics as well as cross-verify the findings obtained from different methods.

Gffcompare [74] is used to identify correctly predicted transcripts and the resulting sensitivity and precision by comparing the intron chains of predicted transcripts to the reference annotation for matching intron-exon structures. A correctly predicted known transcript has an exact intron-chain matching with a reference transcript. Sensitivity is the ratio of the number of correctly predicted known transcripts over the total number of known transcripts, and precision is the ratio of the number of correctly predicted known transcripts over the total number of predicted transcripts. We generate the precision-recall curve (PR-curve) based on the results of Gffcompare by varying the set of predicted transcripts sorted with coverage, and compute the metric PR-AUC (area under the PR-curve) which measures the overall performance. Gffcompare also reports "potential novel isoforms" that are predicted transcripts sharing at least one splice junction with reference transcripts, though this criterion for potential novel isoforms is weak when transcripts contain many splice junctions.

To further examine novel isoforms, we use the evaluation method SQANTI [95] that classifies novel isoforms into Novel in Catalog (NIC) and Novel Not in Catalog (NNC). A transcript classified as NIC either contains new combinations of known splice junctions or contains novel splice junctions formed from known donors and acceptors. NNC contains novel splice junctions formed from novel donors and/or novel acceptors. The criterion for NIC is stronger compared with that of potential novel isoforms in Gffcompare, and we conjecture that NICs may be more likely to be true novel isoforms than wrongly assembled transcripts. SQANTI also reports Full

Splice Match (FSM) that is a predicted transcript matching a reference transcript at all splice junctions, and Incomplete Splice Match (ISM) that is a predicted transcript matching consecutive, but not all, splice junctions of a reference transcript.

Gffcompare and SQANTI report transcripts that fully match, partially match, or do not match reference transcripts, but do not report how many transcripts, for example, have 75–95% or 50–75% of bases matching a reference transcript. These ranges of matched fractions would give us a more detailed view of the overall quality of assembly. Thus, we use rnaQUAST [13] that measures the fraction of a predicted transcript matching a reference transcript. rnaQUAST maps predicted transcripts sequences to the reference genome using GMAP [112] and matches the alignments to the reference transcripts' coordinates from the gene annotation database. rnaQUAST measures the fraction of a reference transcript that is covered by a single predicted transcript, and the fraction of a predicted transcript that matches a reference transcript. Based on the results of rnaQUAST, we compute the distribution of predicted transcripts in different ranges of fractions matching reference transcripts, and the distribution of reference transcripts in different ranges of fractions covered by predicted transcripts. rnaQUAST also reports unaligned transcripts (transcripts without any significant alignments), misassembled transcripts (transcripts that have discordant best-scored alignments, i.e. partial alignments that are mapped to different strands, different chromosomes, in reverse order, or too far away), and unannotated transcripts (predicted transcripts that do not cover any reference transcript).

We use Transrate [92] for sequence-based evaluation to obtain statistics of predicted transcripts such as the minimum, maximum, and mean lengths, the number of bases in the assembly, numbers of transcripts in different size ranges, etc.

The reference annotations we use in Gffcompare, rnaQUAST, and SQANTI are Ensembl *Homo sapiens* GRCh38.90 and *Mus musculus* GRCm38.92. The reference genomes we use are Ensembl GRCh38 for human and GRCm38 for mouse when running rnaQUAST and SQANTI or aligning long reads to the genome (Section 2.2.4).

2.2.3 Data acquisition and preprocessing

We obtained PacBio datasets for *Homo sapiens* and *Mus musculus* from SRA [36, 41, 51, 66, 86, 90]. In most of the PacBio datasets in SRA, one BioSample has multiple SRA Runs because the experimenters used multiple "movies" to increase the coverage so that low-abundance, long isoforms can be captured in analysis. The experimenters also used a size selection sequencing strategy, and thus different SRA Runs are designated for different size ranges. Therefore, we use one BioSample instead of one SRA Run to represent one dataset in our analysis, and we merge multiple SRA Runs that belong to the same BioSample into that dataset. (See Appendix Section 2.5.1 about "movies" and size selection strategy.)

We collected the SRA PacBio datasets that meet the following conditions: (a) The datasets should be transcriptomic and use the cDNA library preparation. (b) The datasets should have the *hdf5* raw data uploaded. This is because if using *fastq-dump* in SRA Toolkit to extract the sequences from SRA, the output sequences lose the original PacBio sequence names even using the sequence-name preserving option. The original PacBio sequence name is critical since it contains information such as the movie, the identification of subreads or CCS reads, etc. (c) The datasets should not be "targeted sequencing" focusing on a specific gene or a small genomic

region. (d) The datasets should use the Iso-Seq2-supported sequencing-chemistry combinations. (e) For a BioSample, the number of SRA Runs should be ≤ 50. This is because a huge dataset is very computationally expensive for Iso-Seq Analysis. With the above conditions, we identified and extracted 18 human datasets and eight mouse datasets—a total of 26 PacBio datasets from SRA. These 26 datasets are sequenced using RS II or RS platform, and their SRA information is in Appendix Table 2.33.

We convert the PacBio raw data to subreads and merge the subreads from multiple movies belonging to the same BioSample into a large dataset for analysis.

2.2.4 Analysis workflow for analyzing the SRA PacBio datasets

Combining our long-read transcript assembly pipeline with the Iso-Seq Analysis pipeline (Iso-Seq2), we build an analysis workflow to analyze the SRA datasets, as shown in Figure 2.2.

After obtaining subreads and creating the merged dataset, we generate CCS reads from subreads. After classifying the CCS reads into full-length and non-full-length reads, the full-length CCS reads are clustered—they are run through the ICE (Iterative Clustering and Error correction) algorithm to generate clusters of isoforms. Afterwards, the non-full-length CCS reads are attributed to the clusters, and the clusters are polished using Quiver or Arrow. Quiver is an algorithm for calling accurate consensus from multiple reads, using a pair-HMM exploiting the basecalls and QV (quality values) metrics to infer the true underlying sequence [68]. Quiver is used for RS and RS II data (for data from the Sequel platform, an improved consensus model Arrow is used). Finally, the polished consensus isoforms are mapped to the genome using GMAP to remove the redundancy, and the final polished isoforms sequences and annotated isoforms are generated.

The right side of the analysis workflow in Figure 2.2 is our long-read transcript assembly pipeline. We chose Minimap2 [52] and GMAP as the long-read aligners. GMAP has been shown to outperform RNA-seq aligners STAR [24], TopHat2 [38], HISAT2 [39], and BBMap [14] in aligning long reads [46]. The RNA-seq aligner Minimap2 is specifically designed for long reads. Minimap2 outperforms GMAP, STAR, and SpAln in junction accuracy, and is 40X faster than GMAP [52]. We did a pre-assessment on the accuracy of Minimap2 vs. GMAP on a set of datasets which are either error-corrected or not error-corrected (results are not shown). Comparing the assembly results, we found that Minimap2 is more accurate than GMAP for long reads without error corrections, and Minimap2 and GMAP have nearly the same accuracy for long reads with error corrections. Thus, we use Minimap2 to align CCS reads (which are not error-corrected), while in the Iso-Seq Analysis pipeline, GMAP is used to align polished isoforms (which are error-corrected). For assembly performance comparison, we choose StringTie as a counterpart, as StringTie outperforms leading transcript assemblers Cufflinks, IsoLasso, Scripture and Traph in short-read assembly [75, 76].

We use the full-length CCS and non-full-length CCS reads as the input of our long-read transcript assembly pipeline for Scallop-LR (v0.9.1) and StringTie (v1.3.2d) to assemble those CCS reads. We first align those CCS reads to the reference genome using Minimap2, and then the alignments are assembled by the transcript assemblers. In addition to taking the alignments as input, Scallop-LR also extracts the boundary information (see Section 2.2.1) from CCS reads.

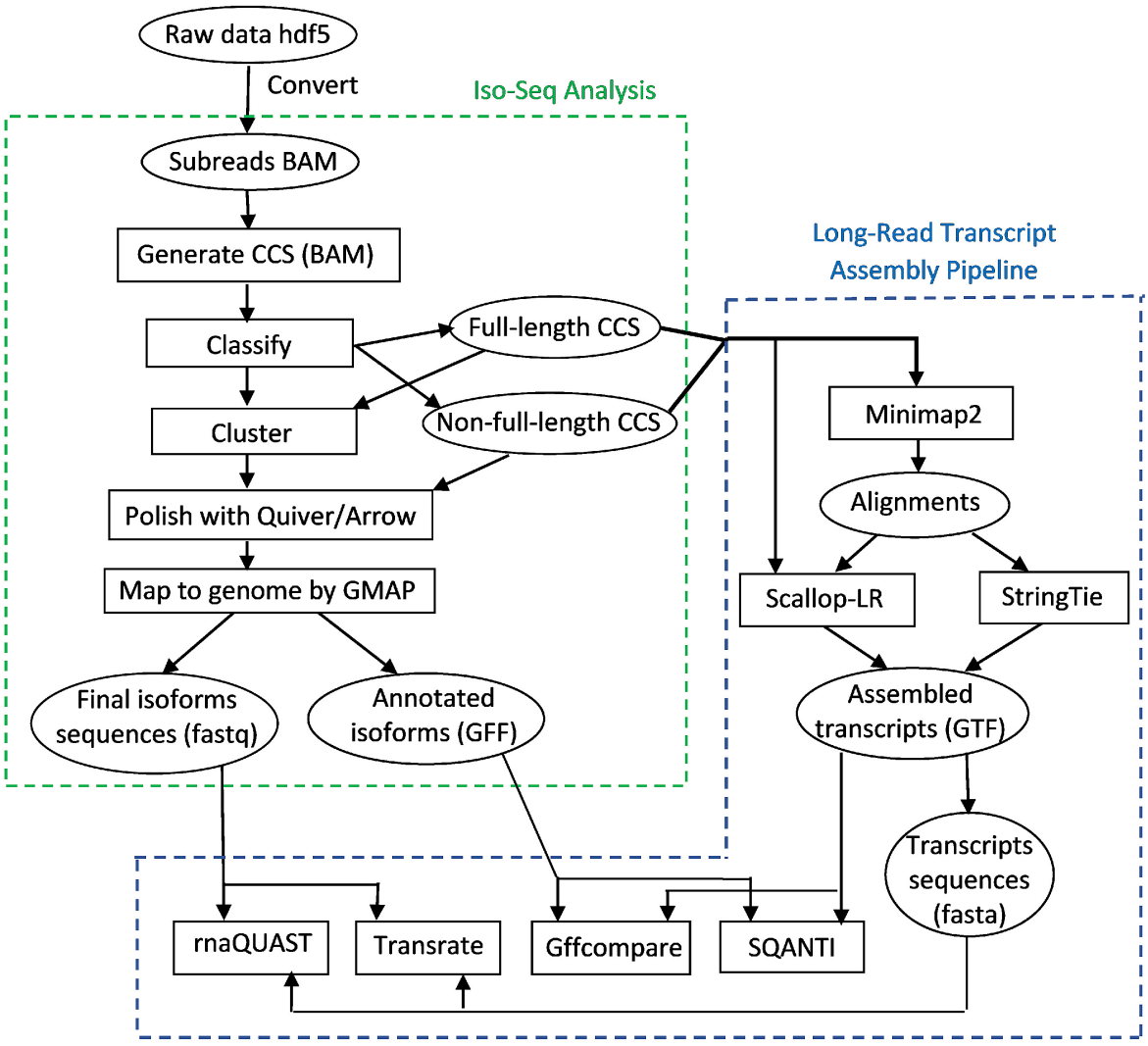


Figure 2.2: Workflow for analyzing the SRA PacBio datasets, combining the long-read transcript assembly pipeline (right) with the Iso-Seq Analysis pipeline (left).

The software versions and options used in this analysis workflow are summarized in Appendix Section 2.5.2. The code to reproduce the analysis is available at:
Scallop-LR: https://github.com/Kingsford-Group/scallop/tree/isoseq; long-read transcript assembly analysis: https://github.com/Kingsford-Group/lrassemblyanalysis.

2.3 Results

2.3.1 Scallop-LR and StringTie predict more known transcripts than Iso-Seq Analysis

From the Gffcompare results for the human data, Scallop-LR and StringTie consistently predict more known transcripts than Iso-Seq Analysis and thus consistently have higher sensitivity than Iso-Seq Analysis. Scallop-LR finds 2100–4000 more known transcripts than Iso-Seq Analysis, and the sensitivity of Scallop-LR is 1.33–1.71 times higher than that of Iso-Seq Analysis (Figures 2.3 and 2.4, Appendix Tables 2.1 and 2.2). StringTie finds 350–1960 more known transcripts than Iso-Seq Analysis, and the sensitivity of StringTie is 1.05–1.4 times higher than that of Iso-Seq Analysis. Scallop-LR and StringTie have higher sensitivity than Iso-Seq Analysis because Scallop-LR and StringTie do assembly but Iso-Seq Analysis does not. This supports the idea that the transcript assembly of long reads is needed. Assembly is likely useful because the success level of transcriptomic long-read sequencing depends on the completeness of cDNA synthesis, and also long reads may not cover those transcripts longer than a certain length limit [78].

In the human data, Scallop-LR also consistently assembles more known transcripts correctly than StringTie and thus consistently has higher sensitivity than StringTie. Scallop-LR finds 950–3770 more known transcripts than StringTie, and the sensitivity of Scallop-LR is 1.14–1.42 times higher than that of StringTie (Figures 2.3 and 2.4, Appendix Tables 2.1 and 2.2). Scallop-LR's higher sensitivity is likely due to its phasing path preservation and its transcript boundary identification in the splice graph based on the boundary information extracted from long reads.

Scallop-LR has higher precision than StringTie for the majority of the datasets. For the first 12 datasets in Figure 2.3 and Appendix Table 2.1, Scallop-LR has both higher sensitivity and higher precision than StringTie. Scallop-LR's higher precision is partially contributed by its post-assembly clustering. However, for the last six datasets in Figure 2.3 and Appendix Table 2.1, Scallop-LR has lower precision than StringTie. The last six datasets in Figure 2.3 (each has 11, 12, 24, or 27 movies) are significantly larger than the first 12 datasets (each has 7 or 8 movies). Scallop-LR's precision decreases in the six larger datasets as it assembles significantly more transcripts in total in these larger datasets (Appendix Table 2.2), while StringTie's precision does not seem to change much with the size of the sample. As the sequencing depth goes up in larger datasets, more lowly-expressed transcripts can be captured by RNA-seq reads. Thus, Scallop-LR is able to identify more lowly-expressed transcripts (Appendix Tables 2.2 and 2.5 show that Scallop-LR finds many more potential novel isoforms in these six much larger datasets), as its core algorithm can preserve all phasing paths (the Scallop paper illustrated the significant improvement of Scallop over other methods in assembling lowly-expressed transcripts). However, overall lowly-expressed transcripts are harder to assemble (as transcripts may not be fully covered by reads), which may lead to the relatively lower precision on these six larger datasets. As-

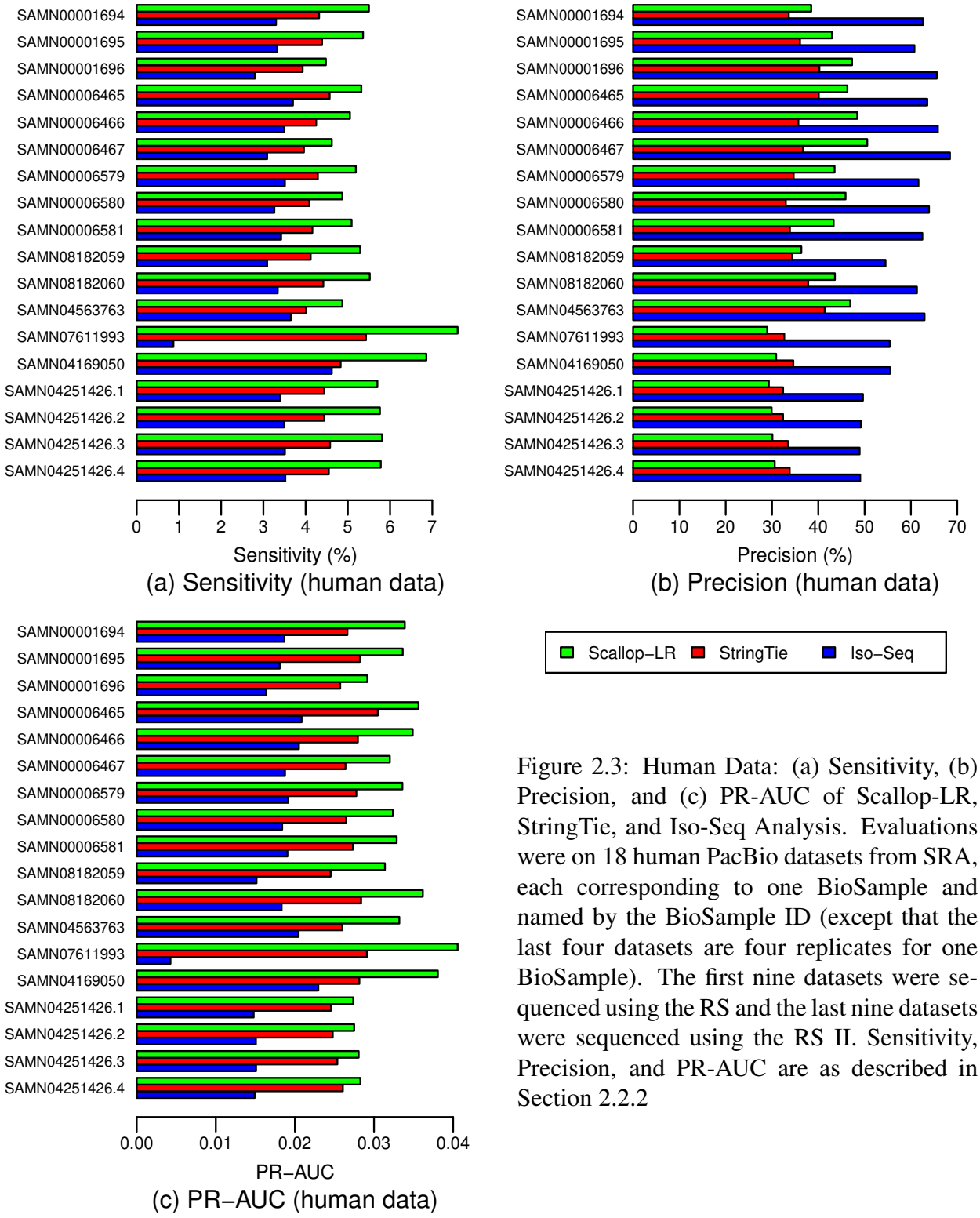
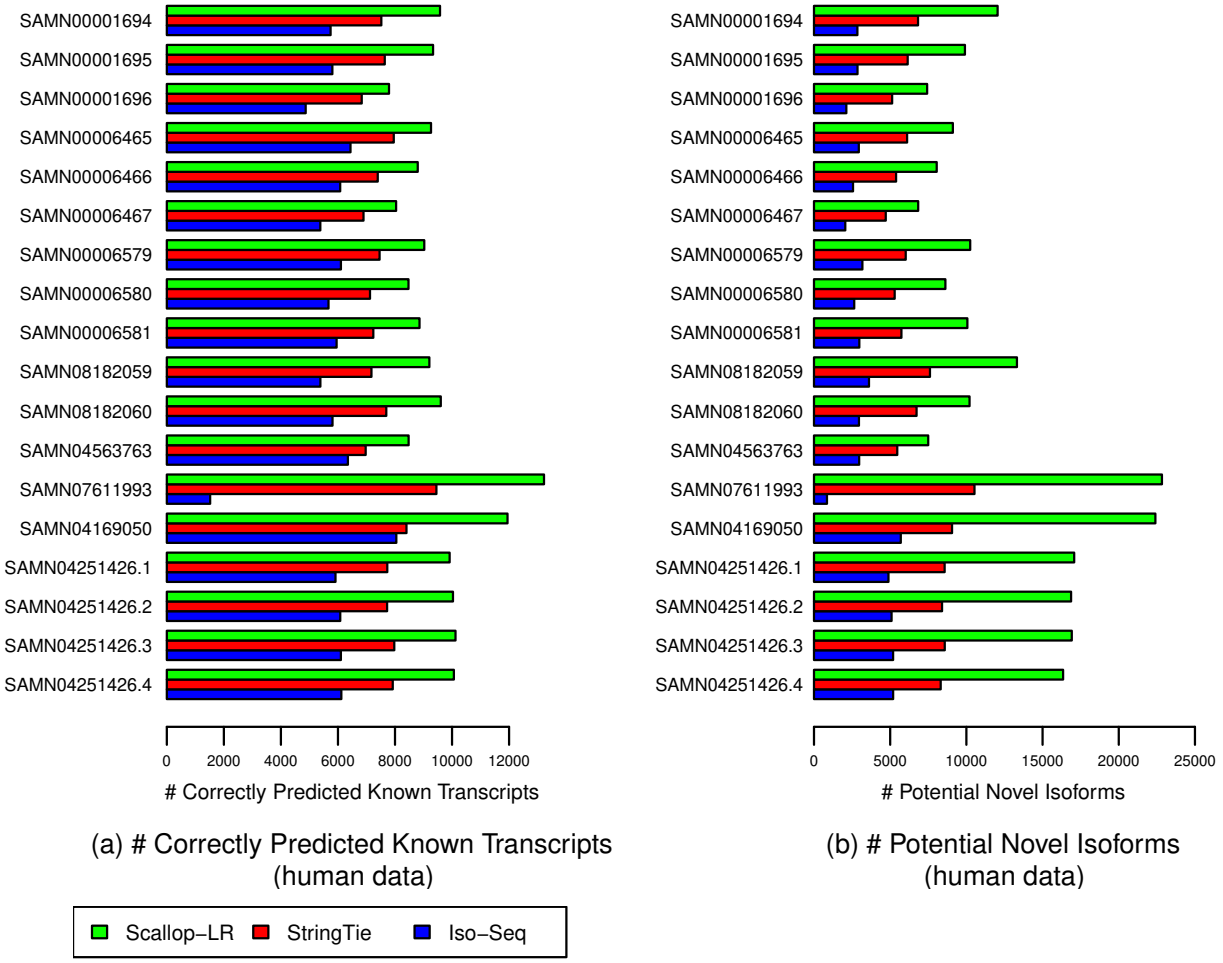


Figure 2.3: Human Data: (a) Sensitivity, (b) Precision, and (c) PR-AUC of Scallop-LR, StringTie, and Iso-Seq Analysis. Evaluations were on 18 human PacBio datasets from SRA, each corresponding to one BioSample and named by the BioSample ID (except that the last four datasets are four replicates for one BioSample). The first nine datasets were sequenced using the RS and the last nine datasets were sequenced using the RS II. Sensitivity, Precision, and PR-AUC are as described in Section 2.2.2



(a) # Correctly Predicted Known Transcripts (human data)

(b) # Potential Novel Isoforms (human data)

Figure 2.4: Human Data: (a) Correctly Predicted Known Transcripts, and (b) Potential Novel Isoforms of Scallop-LR, StringTie, and Iso-Seq Analysis. The same 18 human PacBio datasets as described in Figure 2.3 were evaluated. A Correctly Predicted Known Transcript has the exact intron-chain matching with a transcript in the reference annotation. A Potential Novel Isoform is a predicted transcript that shares at least one splice junction with a reference transcript.

sembling more potential novel isoforms would also lower the precision on these larger datasets as the precision is computed based on the predicted known transcripts.

When two assemblers have opposite trends on sensitivity and precision on a dataset (e.g. the last six datasets in Figure 2.3 and Appendix Table 2.1), we compare their sensitivity and precision on the same footing. That is, for the assembler with a higher sensitivity, we find the precision on its PR curve by matching the sensitivity of the other assembler, and this precision is called adjusted precision. Similarly, we find the sensitivity on its PR curve by matching the precision of the other assembler, and this sensitivity is called adjusted sensitivity. The adjusted sensitivity and precision are needed only when the datasets have opposite trends on sensitivity and precision between assemblers. These adjusted values are shown inside the parentheses on Appendix Table 2.1. Scallop-LR's adjusted sensitivity and adjusted precision are consistently higher than StringTie's sensitivity and precision, indicating that Scallop-LR has consistently better performance than StringTie.

On the other hand, Iso-Seq Analysis consistently has higher precision than Scallop-LR and StringTie (Figure 2.3, Appendix Table 2.1). Iso-Seq Analysis has higher precision partially because the full-length CCS reads are run through the ICE (Iterative Clustering and Error correction) algorithm and the isoforms are also polished with Quiver to achieve higher accuracy.

Scallop-LR consistently has higher PR-AUC than Iso-Seq Analysis and StringTie, indicating better overall performance of Scallop-LR. The PR-AUC of Scallop-LR is 1.62–2.07 times higher than that of Iso-Seq Analysis, and 1.1–1.4 times higher than that of StringTie (Figure 2.3, Appendix Table 2.1).

2.3.2 Scallop-LR and StringTie find more potential novel isoforms than Iso-Seq Analysis

Scallop-LR and StringTie find more potential novel isoforms (i.e. novel transcripts containing at least one annotated splice junction) than Iso-Seq Analysis in the human data. Scallop-LR also consistently finds more potential novel isoforms than StringTie in the human data. Scallop-LR finds 2.53–4.23 times more potential novel isoforms than Iso-Seq Analysis, and 1.37–2.47 times more potential novel isoforms than StringTie (Figure 2.4, Appendix Table 2.2). This is likely due to the same reasons that led to the higher sensitivity of Scallop-LR. This shows the potential benefit that long-read transcript assembly could offer in discovering novel isoforms.

2.3.3 Scallop-LR finds more novel isoforms in catalog than Iso-Seq Analysis

We use SQANTI to evaluate Scallop-LR and Iso-Seq Analysis (SQANTI does not work for the transcripts assembled by StringTie). Figure 2.5 and Appendix Table 2.5 show the SQANTI evaluation results for Scallop-LR and Iso-Seq Analysis on the 18 human datasets.

The NIC (transcripts containing either new combinations of known splice junctions or novel splice junctions with annotated donors and acceptors) results show that Scallop-LR finds more novel isoforms in catalog than Iso-Seq Analysis consistently. Scallop-LR finds 2.2–4.02 times more NIC than Iso-Seq Analysis (Figure 2.5, Appendix Table 2.5). This is an important indica-

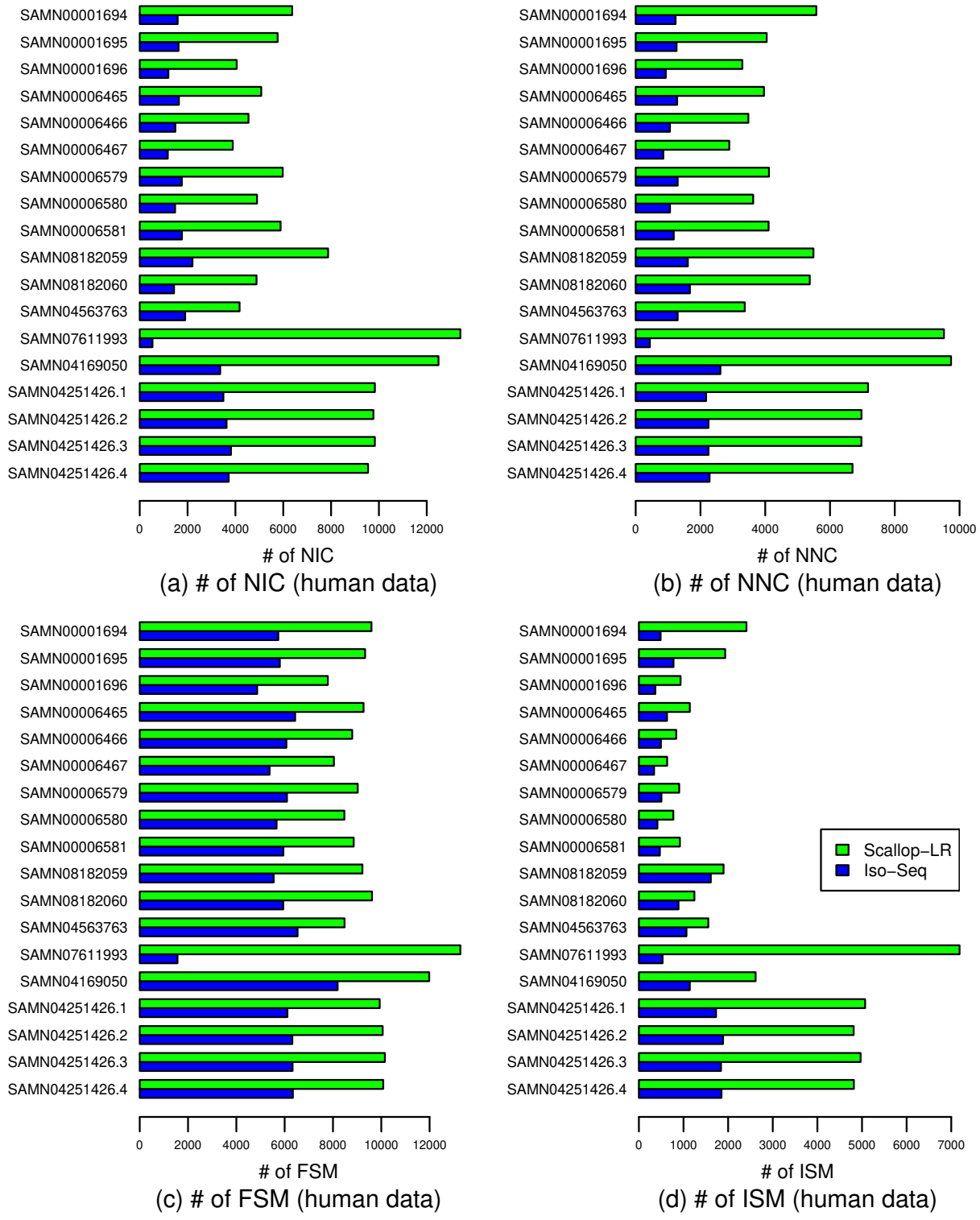


Figure 2.5: Human Data: Numbers of (a) NIC, (b) NNC, (c) FSM, and (d) ISM transcripts of Scallop-LR and Iso-Seq Analysis based on SQANTI evaluations. The same 18 human PacBio datasets as described in Figure 2.3 were evaluated. NIC, NNC, FSM, and ISM are as described in Section 2.2.2

tion of Scallop-LR's ability to find more new transcripts that are not yet annotated, as we conjecture that the novel isoforms in catalog may be more likely to be new transcripts than wrongly assembled transcripts since the novel splice junctions are formed from annotated donors and acceptors. This finding further supports the advantage of assembly of long reads.

The NNC (transcripts containing novel splice junctions with novel donors and/or acceptors) results indicate that Scallop-LR also finds more novel isoforms not in catalog than Iso-Seq Analysis consistently (Figure 2.5, Appendix Table 2.5). The novel isoforms not in catalog could be either new transcripts or wrongly assembled transcripts.

SQANTI's results on novel isoforms are roughly consistent with Gffcompare's results on novel isoforms. Comparing Appendix Table 2.5 with Appendix Table 2.2, we can see that the sums of NIC and NNC from SQANTI are similar to the numbers of potential novel isoforms reported by Gffcompare, except that for the last four datasets in Appendix Table 2.5, for Iso-Seq Analysis, the sums of NIC and NNC are notably larger than the corresponding numbers of potential novel isoforms in Appendix Table 2.2 (this may be because some NIC or NNC may not contain an annotated splice junction although they contain an annotated donor and/or acceptor).

The FSM (Full Splice Match) results from SQANTI support the trend we found from Gffcompare that Scallop-LR consistently predicts more known transcripts correctly than Iso-Seq Analysis. Comparing Appendix Table 2.5 with Appendix Table 2.2, we can see that the numbers of FSM from SQANTI are very close to the numbers of correctly predicted known transcripts from Gffcompare for these datasets.

The ISM (Incomplete Splice Match) results show that Scallop-LR also yields more partially matched transcripts than Iso-Seq Analysis (Figure 2.5, Appendix Table 2.5). The NNC and ISM results support the trend we found from Gffcompare that Iso-Seq Analysis has higher precision than Scallop-LR.

The mouse data exhibit the same trends as the human data as summarized above, which can be seen from Figure 2.6 and Appendix Table 2.6 and by comparing Appendix Table 2.6 with Appendix Table 2.4. In the mouse data, Scallop-LR finds significantly more novel isoforms in catalog (2.43–3.5 times more) than Iso-Seq Analysis consistently (Figure 2.6, Appendix Table 2.6). This further supports our finding on Scallop-LR's ability to discover more new transcripts that are not yet annotated.

2.3.4 Assessment of predicted transcripts that partially match known transcripts

In rnaQUAST, "isoforms" refer to reference transcripts from the gene annotation database, and "transcripts" refer to transcripts predicted by the tools being evaluated. Here, we inherit these terminologies. Figures 2.7, 2.8, and 2.9 show box-whisker plots of matched transcripts in matched fraction bins, assembled isoforms in assembled fraction bins, "mean isoform assembly" and "mean fraction of transcript matched" for Scallop-LR, StringTie, and Iso-Seq Analysis on the 18 human datasets based on rnaQUAST evaluations. Full results are shown in Appendix Tables 2.7–2.24.

Scallop-LR predicts more transcripts that have a high fraction of their bases matching reference transcripts than both Iso-Seq Analysis and StringTie. The metric "x-y% matched tran-

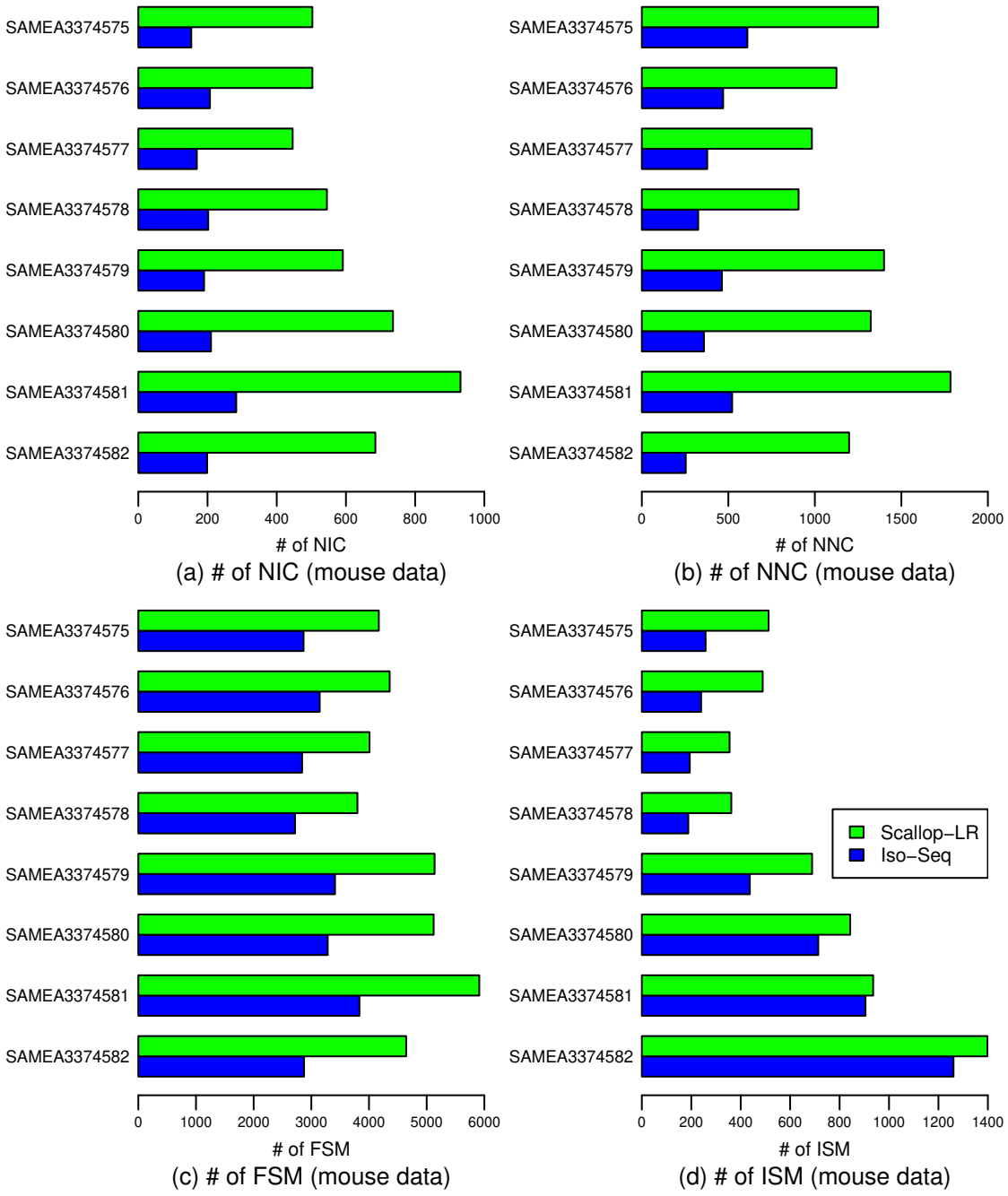


Figure 2.6: Mouse Data: Numbers of (a) NIC, (b) NNC, (c) FSM, and (d) ISM transcripts of Scallop-LR and Iso-Seq Analysis based on SQANTI evaluations. Evaluations were on eight mouse PacBio datasets from SRA, each corresponding to one BioSample and named by the BioSample ID. All eight datasets were sequenced using the RS. Metrics descriptions are the same as in Figure 2.5.

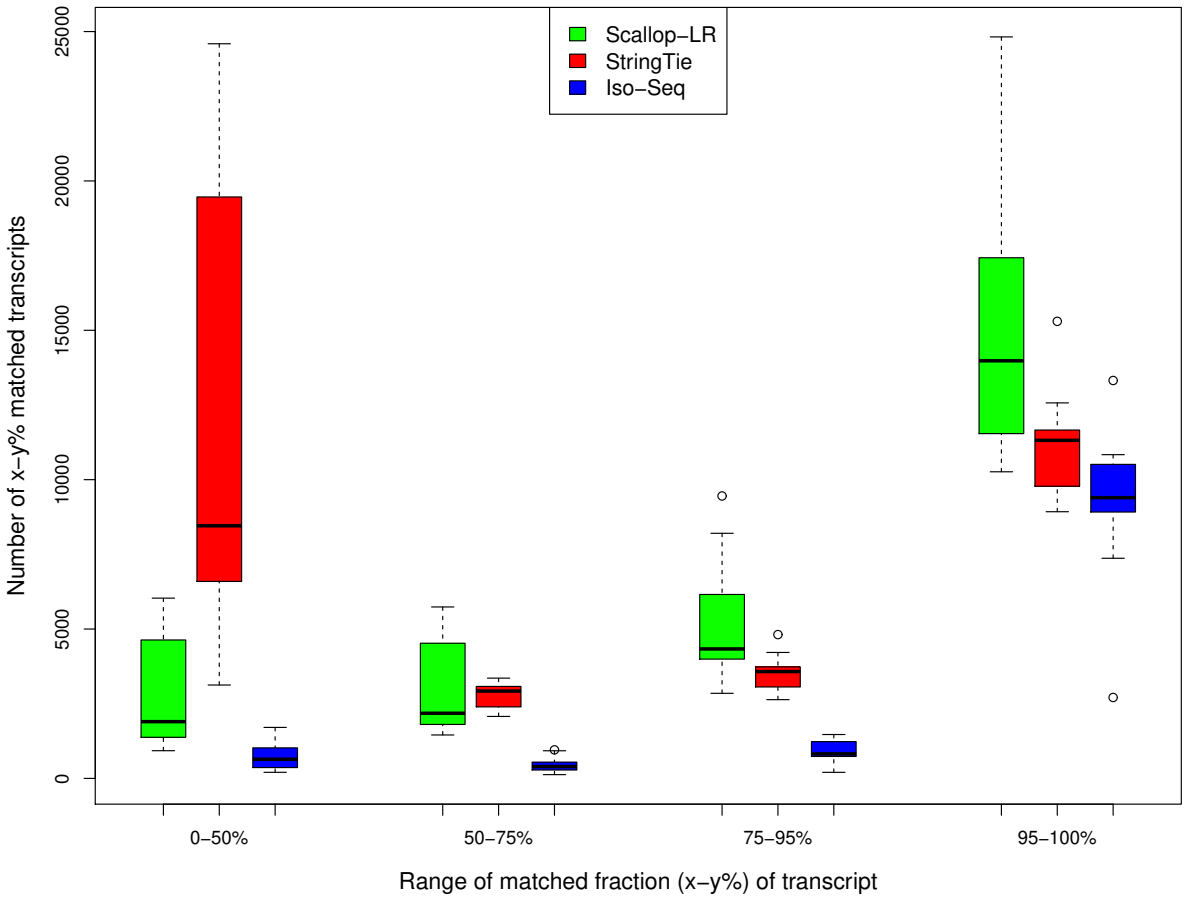


Figure 2.7: Human data: box-whisker plots of matched transcripts in four matched fraction bins for Scallop-LR, StringTie, and Iso-Seq Analysis, based on rnaQUAST evaluations. This is to compare numbers of x-y% matched transcripts. The same 18 human PacBio datasets as described in Figure 2.3 were evaluated. "Number of x-y% matched transcripts" is as described in Section 2.3.4. The four bins of matched fraction (x-y%) of transcript are 0–50%, 50–75%, 75–95%, and 95–100%.

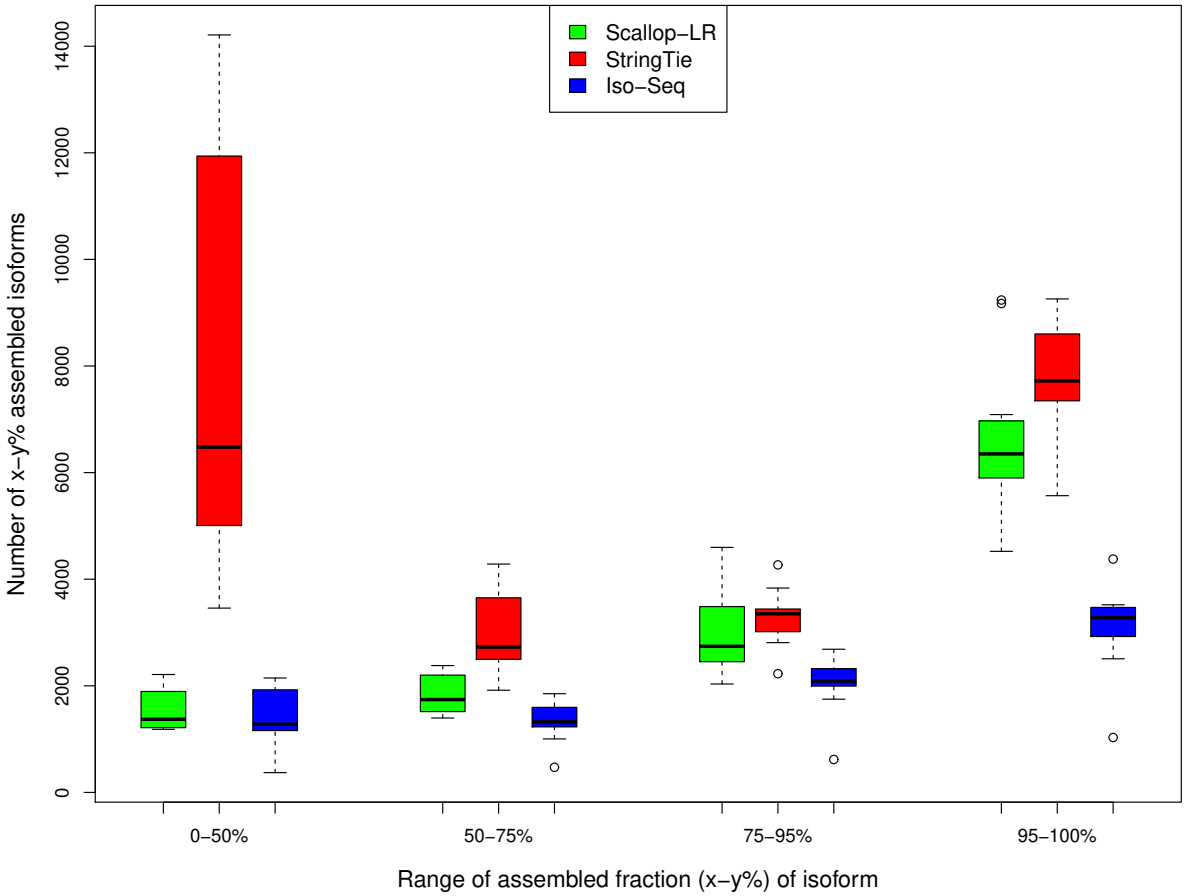


Figure 2.8: Human data: box-whisker plots of assembled isoforms in four assembled fraction bins for Scallop-LR, StringTie, and Iso-Seq Analysis, based on rnaQUAST evaluations. This is to compare numbers of x-y% assembled isoforms. The same 18 human PacBio datasets as described in Figure 2.3 were evaluated. "Number of x-y% assembled isoforms" is as described in Section 2.3.4. The four bins of assembled fraction (x-y%) of isoform are 0–50%, 50–75%, 75–95%, and 95–100%.

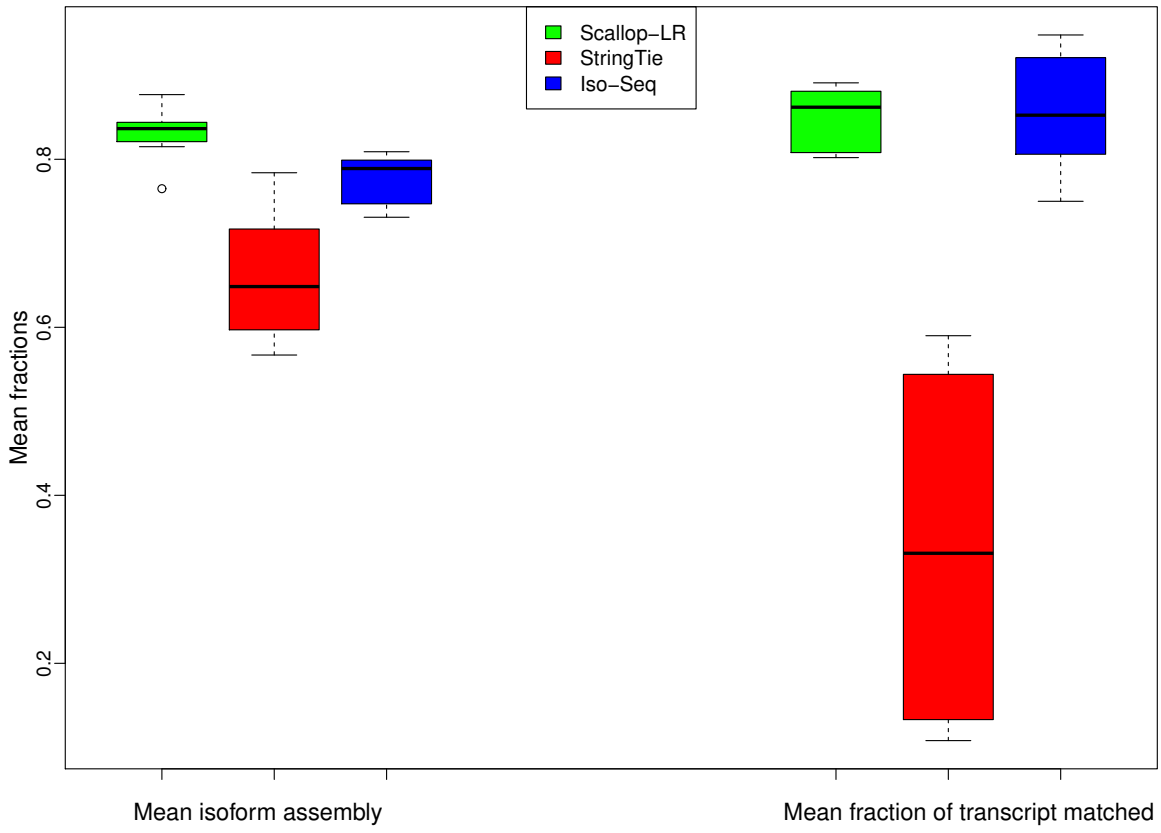


Figure 2.9: Human data: box-whisker plots of mean isoform assembly and mean fraction of transcript matched for Scallop-LR, StringTie, and Iso-Seq Analysis, based on rnaQUAST evaluations. The same 18 human PacBio datasets as described in Figure 2.3 were evaluated. "Mean isoform assembly" and "Mean fraction of transcript matched" are as described in Section 2.3.4.

scripts" is the number of transcripts that have at least x% and at most y% of their bases matching an isoform from the annotation database. We report this measure in four different bins to examine how well predicted transcripts match reference transcripts. From Appendix Tables 2.7–2.24, in the high % bins of the "x-y% matched transcripts" (75–95% and 95–100% matched), Scallop-LR predicts more x-y% matched transcripts than both Iso-Seq Analysis and StringTie (with one exception compared with StringTie). This trend is visualized in Figure 2.7 (75–95% and 95–100% matched bins). In the high % bins, StringTie mostly has more x-y% matched transcripts than Iso-Seq Analysis. These further support the advantage of transcript assembly on long reads.

On average, Scallop-LR transcripts match reference transcripts much better than StringTie transcripts. The metric "Mean fraction of transcript matched" is the average value of matched fractions, where the matched fraction of a transcript is computed as the number of its bases covering an isoform divided by the transcript length. This measure indicates on average how well predicted transcripts match reference transcripts. In Appendix Tables 2.7–2.24, Scallop-LR consistently has much higher values of "Mean fraction of transcript matched" than StringTie, indicating its better assembly quality than StringTie. Scallop-LR performs slightly better than Iso-Seq Analysis on this measure. These trends are visualized in Figure 2.9 (right: "Mean fraction of transcript matched").

There are more reference transcripts that have a high fraction of their bases being captured/covered by Scallop-LR transcripts than by Iso-Seq Analysis predicted transcripts. The metric "x-y% assembled isoforms" is the number of isoforms from the annotation database that have at least x% and at most y% of their bases captured by a single predicted transcript. We report this measure in four different bins to examine how well reference transcripts are captured/covered by predicted transcripts. From Appendix Tables 2.7–2.24, in the high % bins of the "x-y% assembled isoforms" (75–95% and 95–100% assembled), Scallop-LR consistently has more x-y% assembled isoforms than Iso-Seq Analysis. However, Scallop-LR mostly (with six exceptions in the 75–95% bin and two exceptions in the 95–100% bin) has fewer x-y% assembled isoforms than StringTie in the high % bins. These trends are visualized in Figure 2.8 (75–95% and 95–100% assembled bins).

However, on average, reference transcripts are better captured/covered by Scallop-LR transcripts than by StringTie transcripts and Iso-Seq Analysis transcripts. The metric "Mean isoform assembly" is the average value of assembled fractions, where the assembled fraction of an isoform is computed as the largest number of its bases captured by a single predicted transcript divided by its length. This measure shows on average how well reference transcripts are captured by predicted transcripts. In Appendix Tables 2.7–2.24, Scallop-LR consistently has higher values of "Mean isoform assembly" than both StringTie and Iso-Seq Analysis. This trend is visualized in Figure 2.9 (left: "Mean isoform assembly"). This trend is consistent with the higher sensitivity of Scallop-LR in the Gffcompare results.

Scallop-LR consistently has fewer unannotated, misassembled, and unaligned transcripts than StringTie (Appendix Tables 2.7–2.24). This further indicates Scallop-LR's better assembly quality than StringTie. Scallop-LR mostly (with three exceptions) produces fewer unannotated transcripts than Iso-Seq Analysis as well. An unannotated transcript reported by rnaQUAST denotes an assembled transcript mapped to intergenic space, and thus does not relate to the novel isoforms identified by Gffcompare or SQANTI.

There are a few notable findings regarding StringTie transcripts. First, StringTie consistently

has significantly more unannotated transcripts than both Scallop-LR and Iso-Seq Analysis (Appendix Tables 2.7–2.24). Second, in Figure 2.7, in the 0–50% matched bin, StringTie has significantly higher numbers of transcripts than Scallop-LR and Iso-Seq Analysis. This indicates that StringTie assembled many more lower-quality transcripts than Scallop-LR and Iso-Seq Analysis, consistent with StringTie predicting many more unannotated transcripts. Lastly, in Figure 2.8, in the 0–50% assembled bin, StringTie has significantly higher numbers of isoforms than Scallop-LR and Iso-Seq Analysis. This indicates that, compared with Scallop-LR and Iso-Seq Analysis, there are many more isoforms from the annotation which are just marginally covered by StringTie transcripts.

The mouse data exhibit trends partially similar to those of the human data for the rnaQUAST results, and the quality of StringTie transcripts in the mouse data is somewhat improved compared to that in the human data. The detailed discussions on the rnaQUAST results for the mouse data are in Appendix Section 2.5.3.

We also evaluated Scallop-LR and StringTie on a simulated human dataset from Liu et al. [57]. The results and discussions for the simulated dataset are in Appendix Section 2.5.4.

2.3.5 Scallop-LR and StringTie predict more known transcripts and potential novel isoforms than Iso-Seq Analysis in mouse data

From the Gffcompare evaluation for the mouse data (Figure 2.10, Appendix Tables 2.3 and 2.4), Scallop-LR and StringTie consistently predict more known transcripts (Scallop-LR predicts 1100–2200 more) correctly than Iso-Seq Analysis and thus consistently have higher sensitivity (Scallop-LR's is 1.43–1.72 times higher) than Iso-Seq Analysis. Scallop-LR and StringTie also find more potential novel isoforms (Scallop-LR finds 2.38–4.36 times more) than Iso-Seq Analysis (Appendix Table 2.4). Scallop-LR and StringTie consistently have higher PR-AUC than Iso-Seq Analysis (Figure 2.10, Appendix Table 2.3).

We also found some trends different from those in the human data. In the mouse data, Scallop-LR consistently has higher precision than StringTie, but consistently has lower sensitivity than StringTie (Figure 2.10, Appendix Table 2.3). Thus, for StringTie we computed the adjusted sensitivity by matching Scallop-LR's precision and the adjusted precision by matching Scallop-LR's sensitivity. These adjusted values are shown inside the parentheses on Appendix Table 2.3. Scallop-LR's sensitivity and precision are consistently higher than StringTie's adjusted sensitivity and adjusted precision, indicating that when comparing on the same footing, Scallop-LR does better on these measures than StringTie.

In the mouse data, the trend of PR-AUC between Scallop-LR and StringTie is mixed (Figure 2.10, Appendix Table 2.3). Scallop-LR also finds fewer potential novel isoforms than StringTie (Appendix Table 2.4).

Before this work, Scallop was never systematically evaluated on organisms besides human, for either short reads or long reads. In fact, Scallop's parameters were optimized by targeting the human transcriptome. The current annotated mouse transcriptome is relatively less complex than the annotated human transcriptome although they share many similarities. It may be possible that some of Scallop-LR's advantages (such as preserving phasing paths) become less significant in a relatively less complex transcriptome.

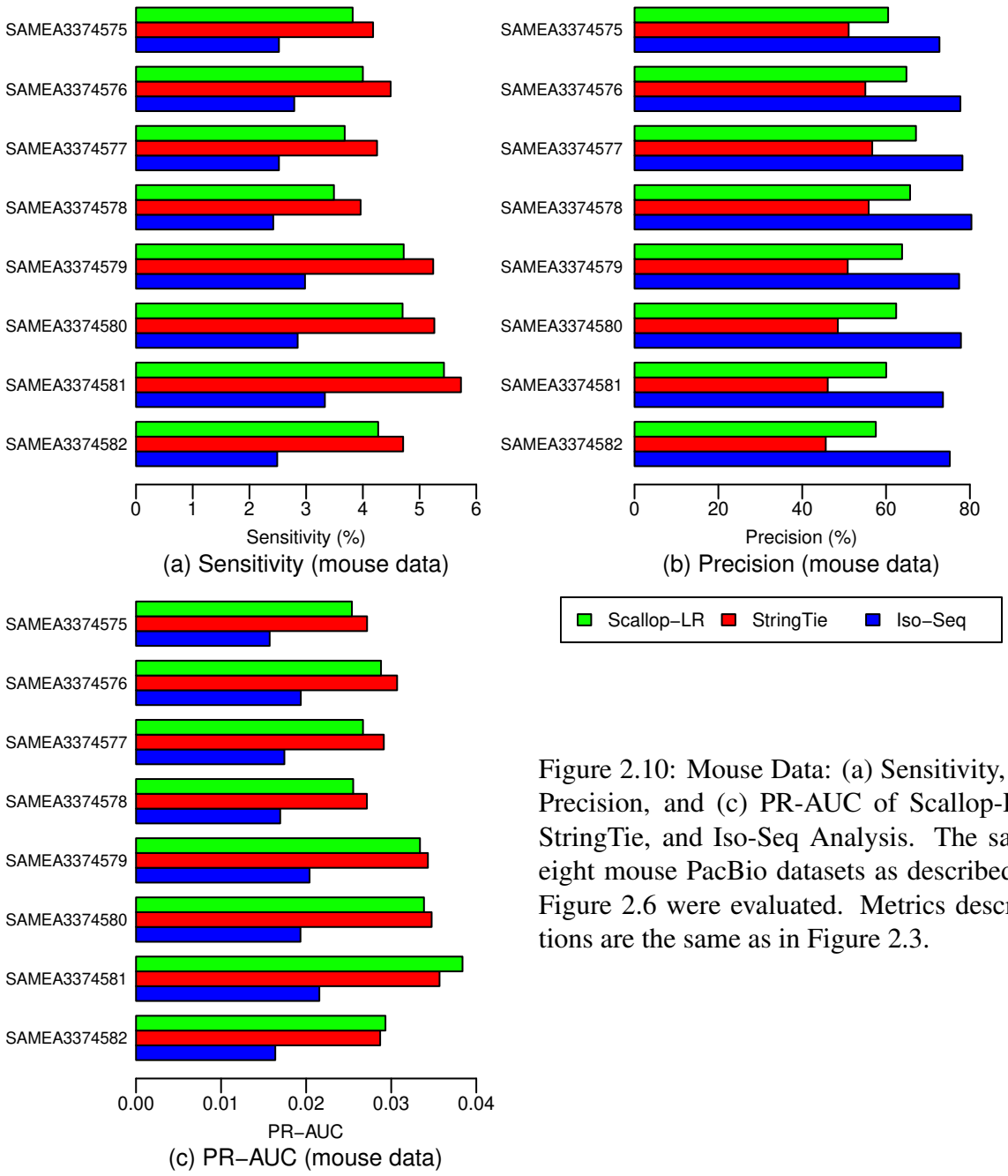


Figure 2.10: Mouse Data: (a) Sensitivity, (b) Precision, and (c) PR-AUC of Scallop-LR, StringTie, and Iso-Seq Analysis. The same eight mouse PacBio datasets as described in Figure 2.6 were evaluated. Metrics descriptions are the same as in Figure 2.3.

2.4 Discussion

The combined evaluations using Gffcompare, SQANTI, and rnaQUAST yield consistent observations that Scallop-LR not only correctly assembles more known transcripts but also finds more possible novel isoforms than Iso-Seq Analysis, which does not do assembly. Scallop-LR finding more NIC especially shows its ability to discover new transcripts. These observations further support the idea that transcript assembly of long reads is needed and demonstrate that long-read assembly by Scallop-LR can help reveal a more complete human transcriptome using long reads.

Two factors may limit the CCS read length: the read length of the platform and the cDNA template sizes. In many cases, the primary limiting factor for CCS read lengths is the cDNA template sizes [89]. When a cDNA is very long so that the continuous polymerase read is unable to get through at least two full passes of the template, the CCS read is not generated for that cDNA. Thus, the maximum possible CCS read length is limited by the read length of the platform. The read lengths of sequencing platforms have been increasing, however, there are limitations imposed by the cDNA synthesis methods.

cDNA synthesis can be incomplete with respect to the original mRNAs [89]. A CCS read represents the entire cDNA molecule, however, the CCS read could correspond to a partial transcript as a result of incomplete cDNAs [89]. The longer the transcripts are, the lower the fraction of CCS reads that can represent the entire splice structures of mRNAs is [89]. This is likely a reason that Scallop-LR is able to find more true transcripts through assembly: a fraction of CCS reads can be partial sequences of those long transcripts, and Scallop-LR is able to assemble them together to reconstruct the original transcripts.

Iso-Seq Analysis may also sacrifice some true transcripts in order to achieve a higher quality (i.e. less affected by the sequencing errors) in final isoforms. The "polish" step in Iso-Seq Analysis keeps only the isoforms with at least two full-length reads to support them. This increases the isoform quality and gives Iso-Seq Analysis a higher precision than Scallop-LR, but may cause Iso-Seq Analysis to miss those low-abundance, long transcripts with only one full-length read.

Although StringTie was designed for assembling short reads, it also exhibits the advantage of assembly of long reads compared to Iso-Seq Analysis. StringTie finds more known transcripts and potential novel isoforms than Iso-Seq Analysis. In the rnaQUAST results, StringTie produces large numbers of unannotated transcripts (in a range of 7600–113000 for the human datasets), significantly more than those of Scallop-LR and Iso-Seq Analysis (differing by orders of magnitude). Unannotated transcripts are the transcripts that do not have a fraction matching a reference transcript in the annotation database. StringTie also outputs large numbers of single-exon transcripts, significantly more than those of Scallop-LR and Iso-Seq Analysis (differing by orders of magnitude). We found that about 70% of the unannotated transcripts from StringTie are those single-exon transcripts. StringTie produces large numbers of single-exon transcripts most likely because StringTie discards the spliced read alignments that do not have the transcript strand information. There is a fraction of read alignments by Minimap2 which have no transcript strand information, since Minimap2 looks for the canonical splicing signal to infer the transcript strand and for some reads the transcript strands are undetermined by Minimap2. When those spliced alignments that do not have the transcript strand information are ignored by StringTie, the single-exon alignments that overlap those spliced alignments turn into single-exon transcripts by themselves, although they could have been represented by the spliced multi-exon transcripts

during the assembly if those spliced alignments they overlap were not ignored. Unlike StringTie, Scallop-LR attempts both strands if a read alignment has no transcript strand information.

Scallop-LR eliminates nearly redundant transcripts through post-assembly clustering. For reference-based assembly, clustering the transcripts with very similar splice positions into a single transcript could have a side effect that some true transcripts may also be eliminated by the clustering since some real transcripts may have very similar splice positions. Therefore, we investigated this effect by comparing the results of Scallop-LR without post-assembly clustering with the results of Scallop-LR with post-assembly clustering, and computing the percentages of correctly assembled known transcripts that are missing because of the clustering and the percentages of nearly redundant transcripts that are removed by the clustering (Appendix Table 2.35). For the 18 human datasets, we found that the percentages of correctly assembled known transcripts missing due to clustering are between 1.43% – 2.38% (this percentage < 2% for all datasets except for two), and the percentages of nearly redundant transcripts removed by clustering are between 9.22% – 15.52% (this percentage > 10% for all datasets except for four). These results indicate that the effect of missing correctly assembled known transcripts by the post-assembly clustering is relatively minor, while the post-assembly clustering substantially removes nearly redundant transcripts and significantly improves the precision. Decreasing the allowance for splice positions' differences (the parameter "--max_cluster_intron_distance"; the default is 10 bp) could further reduce the side effect of missing correctly assembled known transcripts due to the clustering.

We also compared the performance of Scallop-LR (v0.9.1) with the performance of the short-read assembler Scallop (v0.10.3) for the 18 human datasets using the Gffcompare evaluation (Appendix Table 2.34). We adjusted the parameters of Scallop so that it can also assemble long reads (by setting "--max_num_cigar 1000" and "--min_num_hits_in_bundle 1"). The precision of Scallop-LR increases compared with that of Scallop: on all 18 datasets Scallop-LR gives higher precision, and the average precision are 39.63% and 34.18% respectively for Scallop-LR and Scallop. The sensitivity of Scallop-LR also increases compared with that of Scallop (except for two datasets Scallop has slightly higher sensitivity than Scallop-LR, and for another two datasets there is a tie): the average numbers of correctly predicted known transcripts are 9543 and 9421 respectively for Scallop-LR and Scallop. These results show the benefits of the long-read-specific optimizations added in Scallop-LR.

A direction for future work is developing a hybrid transcript assembler that combines short and long reads. Recently, two *de novo* transcript assembly methods using hybrid sequencing were developed: IDP-denovo [28] and a new version of Trinity [32]. However, both Trinity and IDP-denovo do not assemble long reads; they assemble short reads and use long reads to extend, supplement, or improve the assembly of short reads. A reference-based hybrid transcript assembler that can assemble both short reads and long reads simultaneously, thus combining the advantages of short reads (low error rates, high throughput) and long reads (long read lengths), is an interesting direction for future work.

2.5 Appendix

2.5.1 Merging multiple SRA Runs from the same BioSample into one dataset

In most of the PacBio datasets in SRA, one BioSample has multiple SRA Runs. We merge multiple SRA Runs that belong to the same BioSample into one dataset. PacBio sequencing uses the template called SMRTbell that is a closed, single-stranded circular DNA created by ligating adaptors to both ends of a target double-stranded cDNA molecule. The sequencing is based on a SMRT Cell, a chip with consumable substrates comprising arrays of zero-mode waveguide (ZMW) nanostructures, and a SMRTbell diffuses into a sequencing unit ZMW on it. The real-time observation of a SMRT Cell is called a movie, and an SRA Run usually contains one movie and sometimes contains multiple movies. One BioSample has multiple SRA Runs because the experimenters used multiple movies (i.e. multiple SMRT Cells) to increase the coverage so that those low-abundance, long isoforms can be captured in Iso-Seq Analysis, since the "polish" step in Iso-Seq Analysis keeps only the isoforms with at least two full-length reads to support them. In most cases, the experimenters also used a size selection sequencing strategy, that is, isoforms that are in different size ranges are split into separate independent SMRTbell libraries for sequencing, so that larger isoforms are not detrimentally dominated by smaller isoform molecules during the sequencing. Thus, different SRA Runs are designated for different size ranges. Therefore, we use one BioSample instead of one SRA Run to represent one dataset in our analysis, and we merge multiple SRA Runs into that dataset.

2.5.2 Software versions and options used in the analysis workflow

The software versions and options used in the analysis workflow are summarized in the following:

Iso-Seq Analysis: Iso-Seq2 from SMRT Link v5.1.0.

Minimap2: v2.2. Options: *-ax splice*.

StringTie: v1.3.2d. Options: *-c 1.0*.

Scallop-LR: v0.9.1. Options: *-c <ccs_read_info > –min_num_hits_in_bundle 1*.

Gffcompare: v0.9.9c. Options: *-M -N -r <reference_annotation>*.

SQANTI: v1.2. Options: *-g*.

rnaQUAST: v.1.5.1. Options: *–transcripts <multiple_assemblies> –reference <reference_genome> –gene_db <gene_database> –gmap_index <gmap_index> –labels <labels> –no_plots –disable_infer_genes –disable_infer_transcripts –lower_threshold <lower_threshold> –upper_threshold <upper_threshold>*.

Transrate: v1.0.3. Options: *–assembly <assembly> –reference <reference_transcriptome>*.

2.5.3 Assessment of predicted transcripts that partially match known transcripts in mouse data

Figures 2.11, 2.12, and 2.13 show box-whisker plots of matched transcripts in matched fraction bins, assembled isoforms in assembled fraction bins, "mean isoform assembly" and "mean frac-

tion of transcript matched" for Scallop-LR, StringTie, and Iso-Seq Analysis on the eight mouse datasets based on rnaQUAST evaluations. Full results are shown in Tables 2.25–2.32.

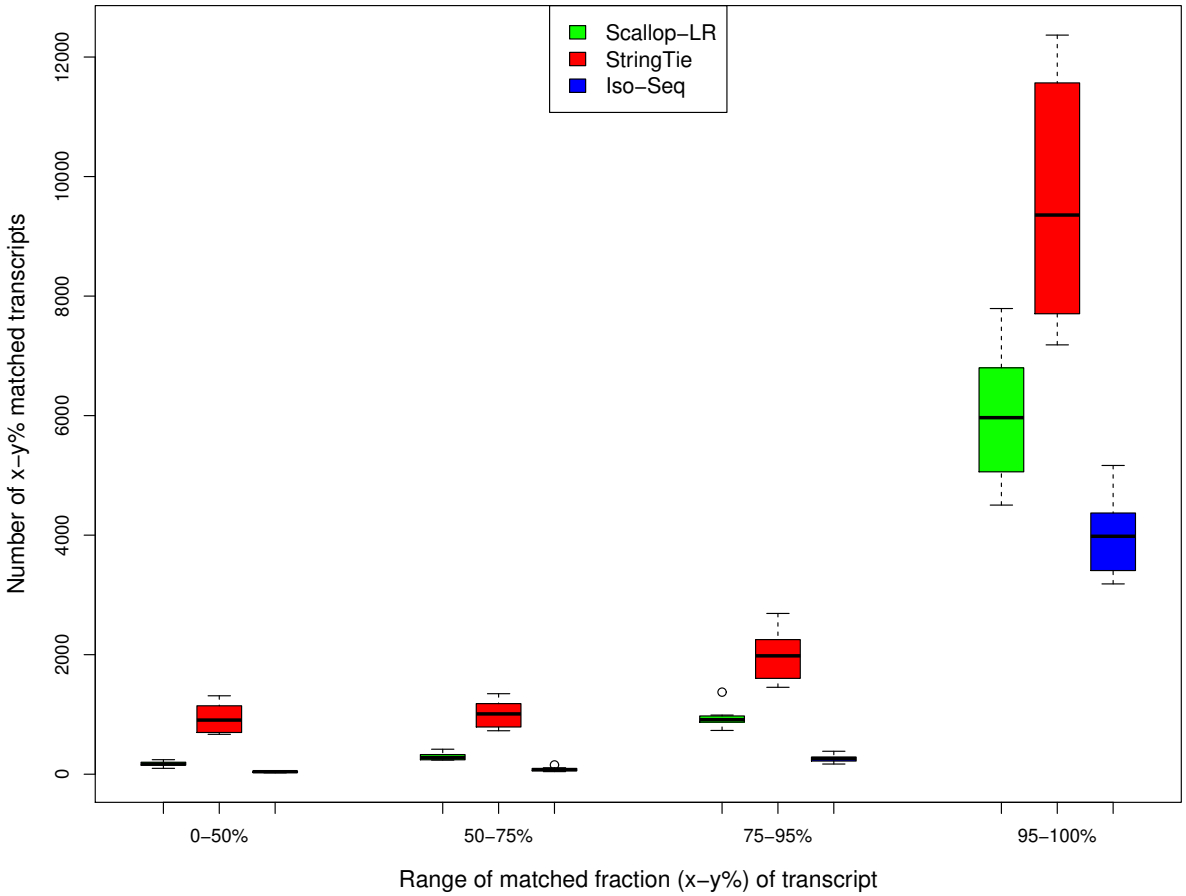


Figure 2.11: Mouse data: box-whisker plots of matched transcripts in four matched fraction bins for Scallop-LR, StringTie, and Iso-Seq Analysis, based on rnaQUAST evaluations. This figure is to compare numbers of x-y% matched transcripts. The same eight mouse PacBio datasets as described before were evaluated via rnaQUAST.

In the mouse data, Scallop-LR predicts more transcripts that have a high fraction of their bases matching reference transcripts than Iso-Seq Analysis. From Tables 2.25–2.32, in the high % bins of the "x-y% matched transcripts" (75-95% and 95-100% matched), Scallop-LR consistently has more x-y% matched transcripts than Iso-Seq Analysis. However, unlike in the human data, Scallop-LR consistently has fewer x-y% matched transcripts than StringTie in the high % bins. These trends are visualized in Figure 2.11 (75-95% and 95-100% matched bins).

However, on average, Scallop-LR transcripts match reference transcripts better than StringTie transcripts. In Tables 2.25–2.32, Scallop-LR consistently has much higher values of "Mean fraction of transcript matched" than StringTie. Scallop-LR has slightly lower values than Iso-Seq

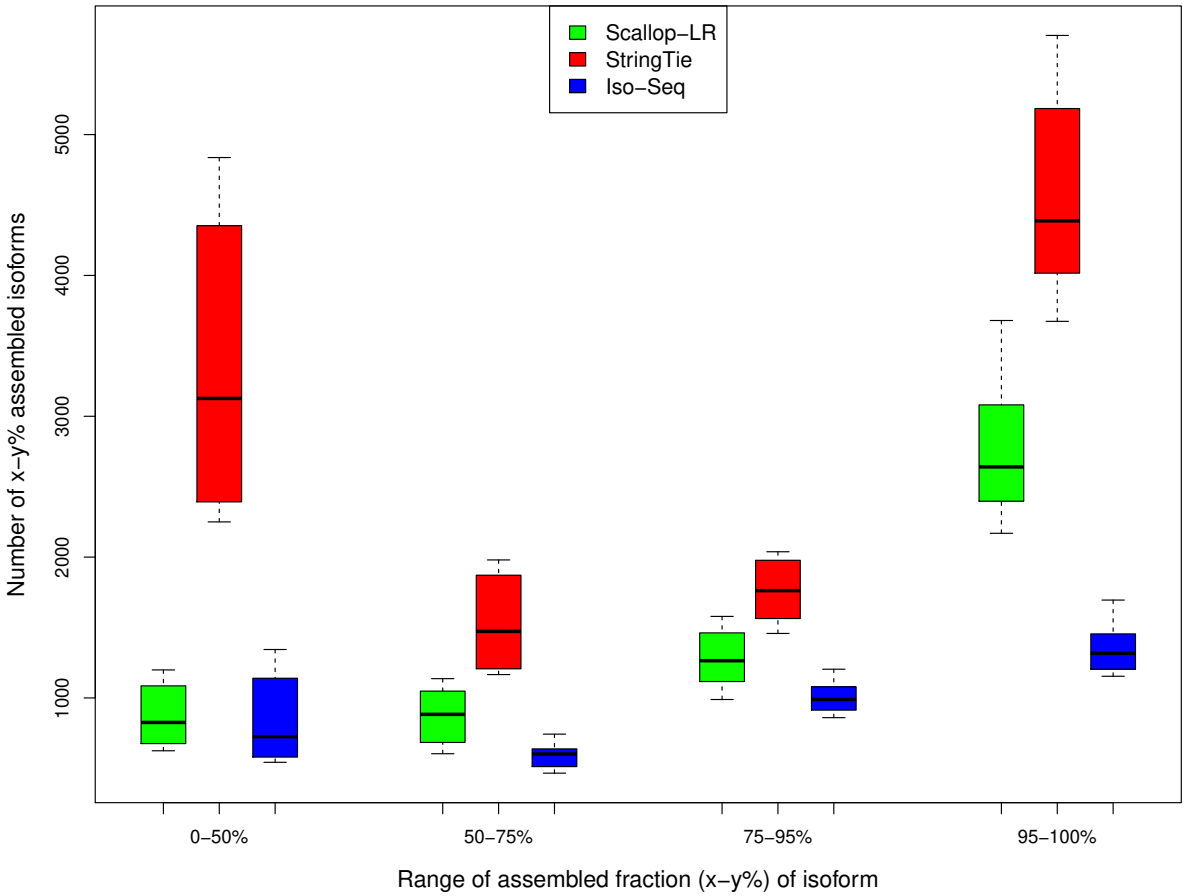


Figure 2.12: Mouse data: box-whisker plots of assembled isoforms in four assembled fraction bins for Scallop-LR, StringTie, and Iso-Seq Analysis, based on rnaQUAST evaluations. This figure is to compare numbers of x-y% assembled isoforms. The same eight mouse PacBio datasets as described before were evaluated via rnaQUAST.

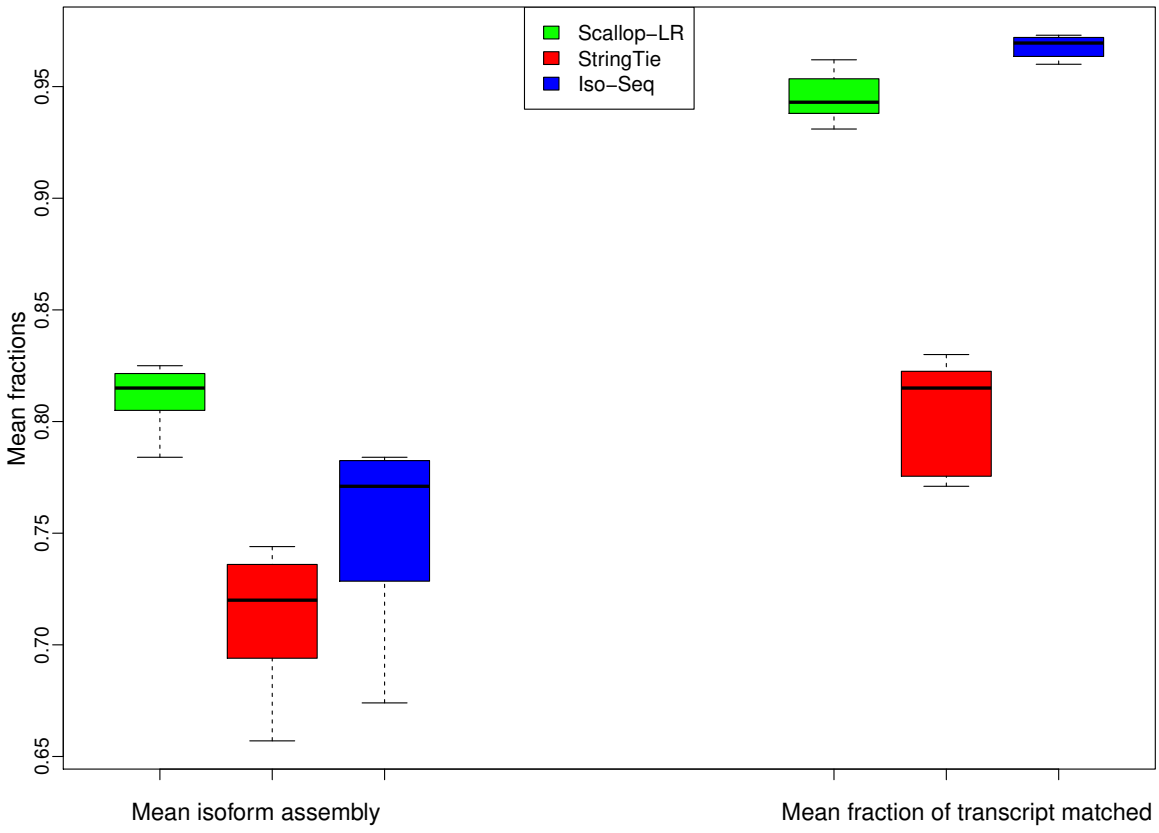


Figure 2.13: Mouse data: box-whisker plots of mean isoform assembly and mean fraction of transcript matched for Scallop-LR, StringTie, and Iso-Seq Analysis, based on rnaQUAST evaluations. The same eight mouse PacBio datasets as described before were evaluated via rnaQUAST.

Analysis though. These trends are visualized in Figure 2.13 (right: "Mean fraction of transcript matched").

In the mouse data, there are more reference transcripts that have a high fraction of their bases being captured/covered by Scallop-LR transcripts than by Iso-Seq Analysis predicted transcripts. From Tables 2.25–2.32, in the high % bins of the "x-y% assembled isoforms" (75-95% and 95-100% assembled), Scallop-LR consistently has more x-y% assembled isoforms than Iso-Seq Analysis. However, Scallop-LR consistently has fewer x-y% assembled isoforms than StringTie in the high % bins. These trends are visualized in Figure 2.12 (75-95% and 95-100% assembled bins).

However, on average, reference transcripts are better captured/covered by Scallop-LR transcripts than by StringTie transcripts and Iso-Seq Analysis transcripts. In Tables 2.25–2.32, Scallop-LR consistently has higher values of "Mean isoform assembly" than both StringTie and Iso-Seq Analysis. Iso-Seq Analysis consistently has higher values than StringTie. This trend is visualized in Figure 2.13 (left: "Mean isoform assembly").

The quality of StringTie transcripts in the mouse data is somewhat improved compared to that in the human data. As in the human data, StringTie consistently has significantly more unannotated transcripts than both Scallop-LR and Iso-Seq Analysis (Tables 2.25–2.32). However, in Figure 2.11, unlike Figure 2.7, in the 0-50% matched bin StringTie no longer has a very high number of transcripts. This indicates that StringTie performs better in the mouse data than in the human data. In Figure 2.12, though, in the 0-50% assembled bin StringTie still has significantly higher numbers of isoforms than both Scallop-LR and Iso-Seq Analysis, similar to Figure 2.8.

2.5.4 Evaluation of Scallop-LR and StringTie on simulated human data

We evaluated Scallop-LR and StringTie on a simulated human dataset [57]. The transcriptome that was used to generate the simulated long reads originated from the Ensembl annotation *Homo sapiens* GRCh38.94 and was a subset of the transcripts in this Ensembl annotation, by removing unfinished scaffolds, transcripts shorter than 200 bp, and annotations with an unknown reference, and randomly selecting alternative-splicing genes, single-splicing genes, and genes with small exons (< 31bp). The PacBio PBSIM tool was used to generate the simulated CCS reads from this transcriptome. The simulation was model-based using the CCS model, and three runs of simulations were performed by using three different sequencing depths 4X, 10X and 30X respectively. We merged the CCS reads generated with the three sequencing depths together to obtain this simulated human dataset. We used the transcripts in the transcriptome sequences that were used to generate the simulated CCS reads to extract the transcripts' records and their corresponding genes' records from the Ensembl annotation *Homo sapiens* GRCh38.94 to obtain an annotation GTF file. This extracted annotation GTF file serves as the "ground truth" and contains 7810 multi-exon transcripts.

In the Gffcompare evaluation, the extracted annotation GTF file corresponding to the transcriptome that was used to generate the simulated CCS reads serves as the reference annotation. Scallop-LR demonstrates both higher sensitivity and higher precision than StringTie (Table 2.36), consistent with the trends on the real human datasets. Note that since the simulated CCS reads do not contain the primer information, Scallop-LR's transcript boundary identification algorithm through extracting the boundary information from long reads is not used on the simulated data.

In the rnaQUAST evaluation, the extracted annotation GTF file corresponding to the transcriptome that was used to generate the simulated CCS reads is used to make the gene annotation database. Therefore, the metric "x-y% assembled isoforms" is computed relative to the initial set of expressed isoforms that was used to generate the simulated reads, rather than all known isoforms. StringTie has more "x-y% assembled isoforms" in the 95-100% bin than Scallop-LR (Table 2.37), consistent with the trend on the majority of the real human datasets. However, Scallop-LR has more "x-y% assembled isoforms" in the 75-95% bin than StringTie, which is different from the majority of the real human datasets. The results of the simulated data confirm that StringTie assembles more "95-100% assembled isoforms" than Scallop-LR, but Scallop-LR assembles more correctly predicted transcripts than StringTie (Table 2.36). This implies that while Scallop-LR outperforms StringTie in terms of the exact reference-matching transcripts (100% assembled, correct predictions), StringTie transcripts cover more reference transcripts on 95-99% of their bases than Scallop-LR transcripts.

We further performed the rnaQUAST evaluation on the simulated dataset by using the entire Ensembl annotation (*Homo sapiens* GRCh38.94) as the reference annotation. The resulting "95-100% assembled isoforms" are 3094 and 4613 for Scallop-LR and StringTie respectively. Compared with the corresponding results in Table 2.37 (which uses the "ground truth" as the reference annotation), StringTie assembles 365 more "95-100% assembled isoforms" when the entire Ensembl annotation is used as the reference. This implies that StringTie assembles 365 transcripts that are not in the ground truth but appear to be misassembled to match the reference transcriptome. For Scallop-LR, the number of "95-100% assembled isoforms" by using the entire Ensembl annotation as the reference is very close to the number of "95-100% assembled isoforms" by using the "ground truth" as the reference (the difference between the numbers of "95-100% assembled isoforms" by using these two references is -9). This result may suggest that, for the real datasets, the "95-100% assembled isoforms" of StringTie could be somewhat inflated (i.e. some assembled transcripts are not in the ground truth but are misassembled to match certain transcripts in the reference), as we do not know the ground truth for the real datasets and use all known isoforms as the reference when evaluating the real data. On the other hand, based on this evidence, it seems that Scallop-LR stays consistent on this measure when the entire Ensembl annotation or the "ground truth" is used as the reference.

2.5.5 Additional tables

Datasets	Sensitivity (%)			Precision (%)			PR-AUC		
	Scallop-LR	StringTie	Iso-Seq	Scallop-LR	StringTie	Iso-Seq	Scallop-LR	StringTie	Iso-Seq
SAMN00001694	5.50	4.32	3.30	38.47	33.63	62.64	0.03389	0.02664	0.01868
SAMN00001695	5.36	4.39	3.33	42.96	36.05	60.76	0.03363	0.02821	0.01808
SAMN00001696	4.48	3.93	2.80	47.29	40.24	65.59	0.02916	0.02573	0.01636
SAMN00006465	5.32	4.57	3.70	46.28	40.10	63.54	0.03563	0.03048	0.02085
SAMN00006466	5.05	4.25	3.49	48.42	35.70	65.82	0.03489	0.02796	0.02051
SAMN00006467	4.62	3.96	3.09	50.57	36.71	68.43	0.03200	0.02640	0.01875
SAMN00006579	5.19	4.29	3.51	43.52	34.68	61.63	0.03359	0.02777	0.01916
SAMN00006580	4.87	4.09	3.26	45.89	32.98	63.91	0.03239	0.02648	0.01839
SAMN00006581	5.09	4.16	3.42	43.31	33.87	62.47	0.03287	0.02733	0.01906
SAMN08182059	5.29	4.12	3.09	36.34	34.34	54.52	0.03138	0.02452	0.01515
SAMN08182060	5.52	4.42	3.34	43.59	37.82	61.30	0.03617	0.02837	0.01832
SAMN04563763	4.87	4.01	3.65	46.90	41.37	62.94	0.03320	0.02600	0.02047
SAMN07611993	7.60 (7.26)	5.43	0.87	28.97 (46.47)	32.65	55.42	0.04057	0.02910	0.00427
SAMN04169050	6.86 (6.61)	4.83	4.62	30.90 (51.74)	34.61	55.52	0.03807	0.02815	0.02296
SAMN04251426.1	5.70 (5.46)	4.44	3.40	29.32 (40.72)	32.40	49.64	0.02738	0.02457	0.01479
SAMN04251426.2	5.76 (5.56)	4.44	3.49	29.92 (41.18)	32.39	49.19	0.02749	0.02478	0.01507
SAMN04251426.3	5.81 (5.52)	4.58	3.51	30.09 (40.37)	33.44	48.93	0.02806	0.02537	0.01509
SAMN04251426.4	5.78 (5.47)	4.55	3.52	30.59 (40.91)	33.82	49.05	0.02828	0.02606	0.01492

Table 2.1: **Human Data: Sensitivity, Precision, and PR-AUC of Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the Gffcompare evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on human data. 18 human PacBio datasets were extracted from SRA, each corresponding to one BioSample and named by the BioSample ID (except that the last four datasets are four replicates for one BioSample). Multiple SRA Runs that belong to each BioSample were merged into a large dataset to perform the analyses. The first nine datasets were sequenced using the RS instrument and the last nine datasets were sequenced using the RS II instrument. Sensitivity is the ratio of the number of correctly predicted known transcripts over the total number of known transcripts, and precision is the ratio of the number of correctly predicted known transcripts over the total number of predicted transcripts. PR-AUC was calculated from the precision-recall curves we generated. The values within the parentheses are the adjusted sensitivity and adjusted precision. The adjusted sensitivity for Scallop-LR was calculated by matching the precision of StringTie, and the adjusted precision for Scallop-LR was calculated by matching the sensitivity of StringTie. The adjusted sensitivity and precision were only calculated for the last six datasets, since the last six datasets have opposite trends on sensitivity and precision comparing Scallop-LR and StringTie.

Datasets	# Potential Novel Isoforms			# Total Multi-Exon Transcripts			# Correctly Predicted Known Transcripts		
	Scallop-LR	StringTie	Iso-Seq	Scallop-LR	StringTie	Iso-Seq	Scallop-LR	StringTie	Iso-Seq
SAMN00001694	12050	6827	2847	24903	22370	9166	9580	7522	5742
SAMN00001695	9905	6149	2856	21731	21201	9554	9336	7642	5805
SAMN00001696	7425	5129	2122	16476	16983	7423	7791	6834	4869
SAMN00006465	9112	6111	2941	20019	19847	10136	9264	7958	6440
SAMN00006466	8054	5387	2561	18171	20715	9236	8798	7396	6079
SAMN00006467	6838	4710	2054	15900	18783	7865	8040	6896	5382
SAMN00006579	10250	6020	3175	20742	21508	9906	9027	7460	6105
SAMN00006580	8623	5295	2640	18467	21607	8870	8474	7125	5669
SAMN00006581	10064	5736	2974	20458	21376	9531	8861	7241	5954
SAMN08182059	13318	7610	3609	25332	20893	9876	9206	7175	5384
SAMN08182060	10207	6732	2951	22038	20348	9478	9606	7696	5810
SAMN04563763	7496	5464	2965	18078	16857	10087	8478	6973	6349
SAMN07611993	22834	10532	852	45657	28953	2741	13226	9453	1519
SAMN04169050	22403	9059	5696	38657	24280	14493	11946	8403	8047
SAMN04251426.1	17074	8572	4887	33824	23863	11913	9916	7731	5914
SAMN04251426.2	16871	8403	5090	33534	23849	12363	10034	7724	6081
SAMN04251426.3	16916	8580	5191	33637	23850	12472	10122	7976	6103
SAMN04251426.4	16347	8314	5194	32908	23423	12476	10065	7922	6119

Table 2.2: **Human Data: Correctly Predicted Known Transcripts, Total Multi-Exon Transcripts, and Potential Novel Isoforms of Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares additional Gffcompare evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on human data. The same 18 human PacBio datasets as described in Table 2.1 were evaluated. A Correctly Predicted Known Transcript is a transcript that has the exact intron-chain matching with a transcript in the reference annotation. A Potential Novel Isoform is a predicted transcript that shares at least one splice junction with a reference transcript. The # Total Multi-Exon Transcripts is the total number of predicted multi-exon transcripts.

Datasets	Sensitivity (%)			Precision (%)			PR-AUC		
	Scallop-LR	StringTie	Iso-Seq	Scallop-LR	StringTie	Iso-Seq	Scallop-LR	StringTie	Iso-Seq
SAMEA3374575	3.82	4.18 (3.70)	2.52	60.50	51.08 (57.55)	72.74	0.02538	0.02716	0.01571
SAMEA3374576	4.00	4.49 (3.87)	2.79	64.86	55.06 (62.48)	77.74	0.02880	0.03069	0.01937
SAMEA3374577	3.68	4.25 (3.61)	2.52	67.14	56.71 (65.44)	78.23	0.02668	0.02913	0.01744
SAMEA3374578	3.49	3.96 (3.38)	2.42	65.74	55.86 (64.21)	80.37	0.02553	0.02714	0.01695
SAMEA3374579	4.72	5.24 (4.48)	2.98	63.83	50.86 (58.13)	77.44	0.03338	0.03431	0.02040
SAMEA3374580	4.70	5.26 (4.52)	2.85	62.42	48.50 (58.28)	77.87	0.03385	0.03475	0.01933
SAMEA3374581	5.43	5.73 (4.93)	3.33	60.04	46.09 (50.63)	73.58	0.03838	0.03568	0.02155
SAMEA3374582	4.27	4.71 (4.05)	2.49	57.57	45.62 (53.16)	75.21	0.02932	0.02870	0.01637

Table 2.3: **Mouse Data: Sensitivity, Precision, and PR-AUC of Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the Gffcompare evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on mouse data. Eight mouse PacBio datasets were extracted from SRA, each corresponding to one BioSample and named by the BioSample ID. Multiple SRA Runs that belong to each BioSample were merged into a large dataset to perform the analyses. All eight datasets were sequenced using the RS instrument. Sensitivity, precision, and PR-AUC are as described in Table 2.1. The values within the parentheses are the adjusted sensitivity and adjusted precision. The adjusted sensitivity for StringTie was calculated by matching the precision of Scallop-LR, and the adjusted precision for StringTie was calculated by matching the sensitivity of Scallop-LR. The adjusted sensitivity and precision were calculated for all eight datasets, since all eight datasets have opposite trends on sensitivity and precision comparing Scallop-LR and StringTie.

Datasets	# Potential Novel Isoforms			# Total Multi-Exon Transcripts			# Correctly Predicted Known Transcripts		
	Scallop-LR	StringTie	Iso-Seq	Scallop-LR	StringTie	Iso-Seq	Scallop-LR	StringTie	Iso-Seq
SAMEA3374575	1973	2476	829	6879	8913	3768	4162	4553	2741
SAMEA3374576	1671	2206	699	6707	8870	3903	4350	4884	3034
SAMEA3374577	1468	1973	596	5962	8155	3505	4003	4625	2742
SAMEA3374578	1434	1958	520	5774	7723	3280	3796	4314	2636
SAMEA3374579	2003	2973	663	8048	11221	4193	5137	5707	3247
SAMEA3374580	2020	3237	543	8204	11801	3981	5121	5724	3100
SAMEA3374581	2714	3806	832	9842	13531	4920	5909	6237	3620
SAMEA3374582	1803	2934	414	8069	11233	3606	4645	5125	2712

Table 2.4: **Mouse Data: Correctly Predicted Known Transcripts, Total Multi-Exon Transcripts, and Potential Novel Isoforms of Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares additional Gffcompare evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on mouse data. The same eight mouse PacBio datasets as described in Table 2.3 were evaluated. The Correctly Predicted Known Transcript, Potential Novel Isoform, and # Total Multi-Exon Transcripts are as described in Table 2.2.

Datasets	# of FSM		# of NIC		# of NNC		# of ISM	
	Scallop-LR	Iso-Seq	Scallop-LR	Iso-Seq	Scallop-LR	Iso-Seq	Scallop-LR	Iso-Seq
SAMN00001694	9594	5736	6375	1586	5579	1232	2411	487
SAMN00001695	9329	5800	5765	1624	4045	1263	1934	776
SAMN00001696	7787	4861	4054	1198	3294	930	934	369
SAMN00006465	9266	6434	5080	1638	3962	1281	1141	630
SAMN00006466	8796	6071	4548	1488	3479	1057	837	493
SAMN00006467	8039	5375	3890	1173	2890	854	633	340
SAMN00006579	9026	6096	5979	1759	4115	1297	903	505
SAMN00006580	8472	5665	4903	1478	3628	1060	772	415
SAMN00006581	8859	5947	5889	1765	4104	1176	919	470
SAMN08182059	9220	5545	7874	2206	5488	1611	1898	1612
SAMN08182060	9616	5941	4882	1433	5375	1674	1244	888
SAMN04563763	8480	6535	4174	1899	3368	1297	1556	1066
SAMN07611993	13275	1564	13406	533	9518	438	7186	531
SAMN04169050	11984	8193	12490	3364	9737	2616	2616	1141
SAMN04251426.1	9934	6114	9831	3497	7174	2175	5070	1729
SAMN04251426.2	10058	6317	9771	3630	6972	2248	4813	1886
SAMN04251426.3	10150	6327	9830	3814	6970	2249	4969	1840
SAMN04251426.4	10081	6340	9545	3719	6695	2282	4818	1851

Table 2.5: **Human Data: Numbers of FSM, NIC, NNC, and ISM transcripts of Scallop-LR and Iso-Seq Analysis based on SQANTI evaluations.** This table compares the SQANTI evaluation results for Scallop-LR and Iso-Seq Analysis on human data. The same 18 human PacBio datasets as described in Table 2.1 were evaluated. FSM (Full Splice Match): the predicted transcript that matches a reference transcript at all splice junctions. ISM (Incomplete Splice Match): the predicted transcript that matches consecutive, but not all, splice junctions of a reference transcript. NIC (Novel in Catalog): the predicted transcript that contains new combinations of already annotated splice junctions or novel splice junctions formed from already annotated donors and acceptors. NNC (Novel Not in Catalog): the predicted transcript that contains novel splice junctions formed from novel donors or/and novel acceptors.

Datasets	# of FSM		# of NIC		# of NNC		# of ISM	
	Scallop-LR	Iso-Seq	Scallop-LR	Iso-Seq	Scallop-LR	Iso-Seq	Scallop-LR	Iso-Seq
SAMEA3374575	4170	2865	503	153	1366	610	513	258
SAMEA3374576	4358	3145	503	207	1125	470	489	240
SAMEA3374577	4009	2840	446	169	983	378	355	194
SAMEA3374578	3799	2718	545	202	906	326	362	188
SAMEA3374579	5137	3410	591	190	1400	463	689	437
SAMEA3374580	5121	3284	736	210	1323	359	843	714
SAMEA3374581	5912	3834	931	283	1784	522	936	905
SAMEA3374582	4644	2874	685	199	1198	254	1398	1261

Table 2.6: **Mouse Data: Numbers of FSM, NIC, NNC, and ISM transcripts of Scallop-LR and Iso-Seq Analysis based on SQANTI evaluations.** This table compares the SQANTI evaluation results for Scallop-LR and Iso-Seq Analysis on mouse data. The same eight mouse PacBio datasets as described in Table 2.3 were evaluated. Metrics descriptions are the same as in Table 2.5.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	25364	32230	11467
Aligned	25363	32224	11467
Uniquely aligned	25361	32028	11444
Unaligned	1	6	0
Misassemblies	2	4	4
0-50% assembled isoforms	1412	4156	1681
50-75% assembled isoforms	1827	2332	1358
75-95% assembled isoforms	2994	3012	1994
95-100% assembled isoforms	6052	6691	2507
Mean isoform assembly	0.834	0.729	0.74
0-50% matched transcripts	1917	4466	267
50-75% matched transcripts	2057	2394	231
75-95% matched transcripts	4612	3659	734
95-100% matched transcripts	16595	12570	9971
Unannotated	180	9118	260
Mean fraction of transcript matched	0.883	0.558	0.939

Table 2.7: **rnaQUAST evaluation results for human dataset SAMN08182059, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. "Isoforms" refer to reference transcripts from the gene annotation database, and "transcripts" refer to predicted transcripts. "# Transcripts" is the total number of predicted transcripts (including single-exon transcripts). "Aligned" is the number of transcripts which have at least one significant alignment to the reference genome. "Uniquely aligned" is the number of transcripts which have a single significant alignment. "Unaligned" is the number of transcripts without any significant alignments. "Misassemblies" are the transcripts that have discordant best-scored alignments (i.e. partial alignments that are mapped to different strands, different chromosomes, in reverse order, or too far away). "Unannotated" is the number of transcripts that do not cover any isoform from the annotation database. "x-y% assembled isoforms" is the number of isoforms from the annotation database that have at least x% and at most y% captured by a single predicted transcript. "x-y% matched transcripts" is the number of transcripts that have at least x% and at most y% matching an isoform from the annotation database. "Mean isoform assembly" is the average value of assembled fractions, where the assembled fraction of an isoform is computed as the largest number of its bases captured by a single predicted transcript divided by its length. "Mean fraction of transcript matched" is the average value of matched fractions, where the matched fraction of a transcript is computed as the number of its bases covering an isoform divided by the transcript length.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	22065	29664	10233
Aligned	22065	29660	10233
Uniquely aligned	22057	29480	10219
Unaligned	0	4	0
Misassemblies	3	4	2
0-50% assembled isoforms	1205	3659	1154
50-75% assembled isoforms	1547	1994	1236
75-95% assembled isoforms	2759	2810	2035
95-100% assembled isoforms	6760	7420	3064
Mean isoform assembly	0.855	0.757	0.794
0-50% matched transcripts	1677	4263	266
50-75% matched transcripts	1806	2147	186
75-95% matched transcripts	4035	3588	652
95-100% matched transcripts	14376	11914	9004
Unannotated	165	7726	123
Mean fraction of transcript matched	0.881	0.577	0.948

Table 2.8: **rnaQUAST evaluation results for human dataset SAMN08182060, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	18097	29003	11103
Aligned	18096	29002	11103
Uniquely aligned	18093	28788	11088
Unaligned	1	1	0
Misassemblies	2	3	0
0-50% assembled isoforms	2212	5762	1926
50-75% assembled isoforms	1792	2496	1411
75-95% assembled isoforms	2034	2227	1954
95-100% assembled isoforms	4523	5567	2669
Mean isoform assembly	0.765	0.657	0.731
0-50% matched transcripts	927	3126	362
50-75% matched transcripts	1657	2337	327
75-95% matched transcripts	4318	3769	1166
95-100% matched transcripts	11053	11733	9105
Unannotated	138	8017	143
Mean fraction of transcript matched	0.89	0.59	0.934

Table 2.9: **rnaQUAST evaluation results for human dataset SAMN04563763, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	45773	42040	3382
Aligned	45769	42028	3382
Uniquely aligned	45763	41813	3378
Unaligned	4	12	0
Misassemblies	5	9	3
0-50% assembled isoforms	1572	3951	371
50-75% assembled isoforms	2380	2458	471
75-95% assembled isoforms	4597	4269	617
95-100% assembled isoforms	9173	9143	1030
Mean isoform assembly	0.855	0.775	0.795
0-50% matched transcripts	5156	6578	204
50-75% matched transcripts	5739	3356	126
75-95% matched transcripts	9454	4814	203
95-100% matched transcripts	24821	15300	2709
Unannotated	594	11959	137
Mean fraction of transcript matched	0.832	0.544	0.883

Table 2.10: **rnaQUAST evaluation results for human dataset SAMN07611993, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	38737	33225	15752
Aligned	38734	33217	15752
Uniquely aligned	38727	33099	15750
Unaligned	3	8	0
Misassemblies	6	7	0
0-50% assembled isoforms	1203	3458	1387
50-75% assembled isoforms	1691	1917	1596
75-95% assembled isoforms	3377	3015	2686
95-100% assembled isoforms	9238	8800	4378
Mean isoform assembly	0.877	0.784	0.809
0-50% matched transcripts	6034	6593	580
50-75% matched transcripts	5428	3086	506
75-95% matched transcripts	8205	4216	1232
95-100% matched transcripts	18839	11650	13320
Unannotated	221	7662	114
Mean fraction of transcript matched	0.802	0.558	0.939

Table 2.11: **rnaQUAST evaluation results for human dataset SAMN04169050, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	33883	41154	15342
Aligned	33882	41147	15342
Uniquely aligned	33881	40883	15322
Unaligned	1	7	0
Misassemblies	1	19	10
0-50% assembled isoforms	1972	5067	1999
50-75% assembled isoforms	2334	2666	1812
75-95% assembled isoforms	3649	3429	2323
95-100% assembled isoforms	6903	7347	3350
Mean isoform assembly	0.815	0.714	0.747
0-50% matched transcripts	4722	7576	1554
50-75% matched transcripts	4542	3059	865
75-95% matched transcripts	6157	3734	1385
95-100% matched transcripts	17879	11661	10297
Unannotated	581	15084	1228
Mean fraction of transcript matched	0.807	0.442	0.806

Table 2.12: **rnaQUAST evaluation results for human dataset SAMN04251426.1, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	33588	41008	16119
Aligned	33587	41004	16119
Uniquely aligned	33578	40814	16104
Unaligned	1	4	0
Misassemblies	6	11	7
0-50% assembled isoforms	1903	5087	2051
50-75% assembled isoforms	2201	2609	1827
75-95% assembled isoforms	3602	3276	2418
95-100% assembled isoforms	7051	7377	3477
Mean isoform assembly	0.821	0.713	0.748
0-50% matched transcripts	4834	7801	1615
50-75% matched transcripts	4525	3048	925
75-95% matched transcripts	6174	3653	1393
95-100% matched transcripts	17429	11337	10787
Unannotated	616	15143	1389
Mean fraction of transcript matched	0.802	0.435	0.8

Table 2.13: **rnaQUAST evaluation results for human dataset SAMN04251426.2, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table‘2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	33699	41038	16328
Aligned	33698	41036	16328
Uniquely aligned	33691	40815	16306
Unaligned	1	2	0
Misassemblies	6	16	4
0-50% assembled isoforms	1894	5007	2146
50-75% assembled isoforms	2236	2616	1834
75-95% assembled isoforms	3486	3416	2417
95-100% assembled isoforms	7089	7384	3471
Mean isoform assembly	0.821	0.717	0.744
0-50% matched transcripts	4595	7693	1707
50-75% matched transcripts	4538	3008	955
75-95% matched transcripts	6329	3757	1469
95-100% matched transcripts	17653	11587	10747
Unannotated	576	14965	1441
Mean fraction of transcript matched	0.808	0.441	0.794

Table 2.14: **rnaQUAST evaluation results for human dataset SAMN04251426.3, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	32952	40754	16179
Aligned	32951	40749	16179
Uniquely aligned	32943	40528	16157
Unaligned	1	5	0
Misassemblies	7	11	7
0-50% assembled isoforms	1940	5087	2109
50-75% assembled isoforms	2246	2729	1852
75-95% assembled isoforms	3563	3361	2430
95-100% assembled isoforms	6970	7220	3414
Mean isoform assembly	0.818	0.711	0.743
0-50% matched transcripts	4634	7569	1647
50-75% matched transcripts	4322	3079	910
75-95% matched transcripts	6116	3565	1440
95-100% matched transcripts	17260	11484	10837
Unannotated	611	15037	1334
Mean fraction of transcript matched	0.806	0.438	0.803

Table 2.15: **rnaQUAST evaluation results for human dataset SAMN04251426.4, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	24956	81557	10813
Aligned	24956	81554	10813
Uniquely aligned	24948	81117	10802
Unaligned	0	3	0
Misassemblies	7	9	0
0-50% assembled isoforms	1547	8224	1070
50-75% assembled isoforms	2016	3162	1263
75-95% assembled isoforms	2967	3329	2025
95-100% assembled isoforms	6470	7697	3107
Mean isoform assembly	0.83	0.64	0.802
0-50% matched transcripts	2015	11463	466
50-75% matched transcripts	2582	2822	398
75-95% matched transcripts	5034	3726	762
95-100% matched transcripts	15137	11144	8916
Unannotated	180	52379	270
Mean fraction of transcript matched	0.868	0.219	0.915

Table 2.16: **rnaQUAST evaluation results for human dataset SAMN00001694, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	21741	107273	12057
Aligned	21741	107269	12057
Uniquely aligned	21736	106575	12039
Unaligned	0	4	0
Misassemblies	3	15	1
0-50% assembled isoforms	1469	10422	1269
50-75% assembled isoforms	1808	3535	1356
75-95% assembled isoforms	2723	3400	2179
95-100% assembled isoforms	6144	8079	3138
Mean isoform assembly	0.832	0.607	0.786
0-50% matched transcripts	1498	15153	606
50-75% matched transcripts	2146	2924	367
75-95% matched transcripts	4350	3529	829
95-100% matched transcripts	13582	10679	9484
Unannotated	161	74942	770
Mean fraction of transcript matched	0.879	0.165	0.872

Table 2.17: **rnaQUAST evaluation results for human dataset SAMN00001695, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	16487	65425	8855
Aligned	16487	65418	8855
Uniquely aligned	16482	65094	8847
Unaligned	0	7	0
Misassemblies	3	4	1
0-50% assembled isoforms	1183	7195	859
50-75% assembled isoforms	1482	2726	1004
75-95% assembled isoforms	2364	2904	1749
95-100% assembled isoforms	5115	6335	2604
Mean isoform assembly	0.834	0.633	0.807
0-50% matched transcripts	1066	9110	350
50-75% matched transcripts	1454	2074	240
75-95% matched transcripts	3011	2636	573
95-100% matched transcripts	10875	9927	7369
Unannotated	77	41656	321
Mean fraction of transcript matched	0.891	0.227	0.911

Table 2.18: **rnaQUAST evaluation results for human dataset SAMN00001696, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	20038	82621	12557
Aligned	20038	82617	12557
Uniquely aligned	20033	82068	12546
Unaligned	0	4	0
Misassemblies	4	11	1
0-50% assembled isoforms	1333	8262	1180
50-75% assembled isoforms	1630	2921	1348
75-95% assembled isoforms	2517	3083	2176
95-100% assembled isoforms	6315	7746	3520
Mean isoform assembly	0.842	0.636	0.804
0-50% matched transcripts	1375	11130	426
50-75% matched transcripts	1856	2375	397
75-95% matched transcripts	3993	3258	854
95-100% matched transcripts	12688	11297	10512
Unannotated	122	54521	367
Mean fraction of transcript matched	0.882	0.207	0.921

Table 2.19: **rnaQUAST evaluation results for human dataset SAMN00006465, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	18192	140280	12335
Aligned	18190	140274	12335
Uniquely aligned	18187	139647	12307
Unaligned	2	6	0
Misassemblies	1	18	1
0-50% assembled isoforms	1315	13372	1414
50-75% assembled isoforms	1502	3942	1229
75-95% assembled isoforms	2481	3485	2072
95-100% assembled isoforms	5899	8561	3280
Mean isoform assembly	0.84	0.574	0.782
0-50% matched transcripts	1300	20842	701
50-75% matched transcripts	1649	3109	391
75-95% matched transcripts	3547	3061	749
95-100% matched transcripts	11541	9780	9300
Unannotated	151	103450	1193
Mean fraction of transcript matched	0.879	0.122	0.833

Table 2.20: **rnaQUAST evaluation results for human dataset SAMN00006466, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	15914	134501	10487
Aligned	15914	134492	10487
Uniquely aligned	15912	133856	10458
Unaligned	0	9	0
Misassemblies	2	17	0
0-50% assembled isoforms	1199	13214	1160
50-75% assembled isoforms	1396	3910	1081
75-95% assembled isoforms	2322	3441	1846
95-100% assembled isoforms	5296	8069	2925
Mean isoform assembly	0.839	0.567	0.79
0-50% matched transcripts	1202	20694	674
50-75% matched transcripts	1463	2767	281
75-95% matched transcripts	2848	2744	574
95-100% matched transcripts	10264	8927	7765
Unannotated	135	99314	1192
Mean fraction of transcript matched	0.878	0.117	0.813

Table 2.21: **rnaQUAST evaluation results for human dataset SAMN00006467, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	20769	124624	13147
Aligned	20769	124621	13147
Uniquely aligned	20768	123959	13128
Unaligned	0	3	0
Misassemblies	1	25	4
0-50% assembled isoforms	1214	12066	1231
50-75% assembled isoforms	1516	3755	1299
75-95% assembled isoforms	2494	3507	2106
95-100% assembled isoforms	6387	8960	3508
Mean isoform assembly	0.85	0.597	0.798
0-50% matched transcripts	1882	20058	848
50-75% matched transcripts	2384	2921	545
75-95% matched transcripts	4223	3167	895
95-100% matched transcripts	12105	9394	9655
Unannotated	174	89030	1199
Mean fraction of transcript matched	0.855	0.133	0.829

Table 2.22: **rnaQUAST evaluation results for human dataset SAMN00006579, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	18480	152741	12817
Aligned	18480	152739	12817
Uniquely aligned	18477	152257	12795
Unaligned	0	2	0
Misassemblies	1	17	5
0-50% assembled isoforms	1246	14212	1290
50-75% assembled isoforms	1497	4284	1192
75-95% assembled isoforms	2438	3834	2001
95-100% assembled isoforms	5809	9259	3276
Mean isoform assembly	0.842	0.578	0.788
0-50% matched transcripts	1688	24592	1023
50-75% matched transcripts	1898	3141	427
75-95% matched transcripts	3507	2930	739
95-100% matched transcripts	11154	8948	8562
Unannotated	230	113096	2059
Mean fraction of transcript matched	0.856	0.108	0.75

Table 2.23: **rnaQUAST evaluation results for human dataset SAMN00006580, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	20476	125563	12512
Aligned	20476	125556	12512
Uniquely aligned	20475	125049	12495
Unaligned	0	7	0
Misassemblies	1	17	2
0-50% assembled isoforms	1279	11940	1195
50-75% assembled isoforms	1582	3652	1243
75-95% assembled isoforms	2452	3340	2128
95-100% assembled isoforms	6273	8602	3409
Mean isoform assembly	0.844	0.592	0.799
0-50% matched transcripts	2036	19464	878
50-75% matched transcripts	2213	2901	417
75-95% matched transcripts	4104	2967	812
95-100% matched transcripts	11946	9341	9311
Unannotated	176	90846	1091
Mean fraction of transcript matched	0.852	0.13	0.832

Table 2.24: **rnaQUAST evaluation results for human dataset SAMN00006581, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a human dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	6890	12610	4017
Aligned	6890	12610	4017
Uniquely aligned	6886	12528	4013
Unaligned	0	0	0
Misassemblies	4	1	0
0-50% assembled isoforms	703	2605	606
50-75% assembled isoforms	761	1258	520
75-95% assembled isoforms	1194	1597	1003
95-100% assembled isoforms	2451	4101	1248
Mean isoform assembly	0.818	0.729	0.779
0-50% matched transcripts	243	765	41
50-75% matched transcripts	369	899	155
75-95% matched transcripts	920	1738	294
95-100% matched transcripts	5323	8108	3510
Unannotated	31	1098	16
Mean fraction of transcript matched	0.931	0.819	0.96

Table 2.25: **rnaQUAST evaluation results for mouse dataset SAMEA3374575, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a mouse dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	6720	12207	4164
Aligned	6720	12206	4164
Uniquely aligned	6715	12104	4163
Unaligned	0	1	0
Misassemblies	5	0	1
0-50% assembled isoforms	699	2406	616
50-75% assembled isoforms	706	1245	585
75-95% assembled isoforms	1155	1632	975
95-100% assembled isoforms	2563	4194	1385
Mean isoform assembly	0.825	0.743	0.783
0-50% matched transcripts	176	700	48
50-75% matched transcripts	275	815	71
75-95% matched transcripts	899	1657	279
95-100% matched transcripts	5334	7967	3749
Unannotated	31	1067	16
Mean fraction of transcript matched	0.94	0.817	0.969

Table 2.26: **rnaQUAST evaluation results for mouse dataset SAMEA3374576, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a mouse dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	5972	11247	3699
Aligned	5972	11247	3699
Uniquely aligned	5967	11193	3695
Unaligned	0	0	0
Misassemblies	4	1	1
0-50% assembled isoforms	653	2250	554
50-75% assembled isoforms	662	1169	505
75-95% assembled isoforms	1078	1531	908
95-100% assembled isoforms	2343	3931	1222
Mean isoform assembly	0.823	0.744	0.784
0-50% matched transcripts	180	667	60
50-75% matched transcripts	236	765	83
75-95% matched transcripts	732	1454	230
95-100% matched transcripts	4796	7442	3305
Unannotated	23	918	20
Mean fraction of transcript matched	0.942	0.826	0.963

Table 2.27: **rnaQUAST evaluation results for mouse dataset SAMEA3374577, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a mouse dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	5783	10979	3492
Aligned	5783	10979	3492
Uniquely aligned	5778	10925	3491
Unaligned	0	0	0
Misassemblies	5	0	0
0-50% assembled isoforms	625	2378	543
50-75% assembled isoforms	604	1166	466
75-95% assembled isoforms	989	1458	860
95-100% assembled isoforms	2169	3674	1154
Mean isoform assembly	0.82	0.726	0.782
0-50% matched transcripts	160	696	23
50-75% matched transcripts	239	727	57
75-95% matched transcripts	842	1550	210
95-100% matched transcripts	4503	7184	3184
Unannotated	34	822	18
Mean fraction of transcript matched	0.936	0.83	0.973

Table 2.28: **rnaQUAST evaluation results for mouse dataset SAMEA3374578, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a mouse dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	8052	16520	4609
Aligned	8052	16520	4609
Uniquely aligned	8052	16414	4600
Unaligned	0	0	0
Misassemblies	0	0	0
0-50% assembled isoforms	949	3648	835
50-75% assembled isoforms	1006	1687	650
75-95% assembled isoforms	1433	1994	1111
95-100% assembled isoforms	3104	5197	1502
Mean isoform assembly	0.812	0.714	0.763
0-50% matched transcripts	155	1046	26
50-75% matched transcripts	288	1119	82
75-95% matched transcripts	989	2223	268
95-100% matched transcripts	6598	10605	4213
Unannotated	22	1526	19
Mean fraction of transcript matched	0.952	0.813	0.972

Table 2.29: **rnaQUAST evaluation results for mouse dataset SAMEA3374579, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a mouse dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	8206	18482	4679
Aligned	8206	18480	4679
Uniquely aligned	8206	18301	4670
Unaligned	0	2	0
Misassemblies	0	0	1
0-50% assembled isoforms	1057	4217	1028
50-75% assembled isoforms	1016	1811	627
75-95% assembled isoforms	1491	1961	1047
95-100% assembled isoforms	3058	5173	1408
Mean isoform assembly	0.802	0.692	0.729
0-50% matched transcripts	149	1199	42
50-75% matched transcripts	281	1196	57
75-95% matched transcripts	960	2229	257
95-100% matched transcripts	6794	11398	4297
Unannotated	22	2457	22
Mean fraction of transcript matched	0.955	0.771	0.972

Table 2.30: **rnaQUAST evaluation results for mouse dataset SAMEA3374580, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a mouse dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	9851	20349	5765
Aligned	9851	20348	5765
Uniquely aligned	9848	20157	5759
Unaligned	0	1	0
Misassemblies	2	2	0
0-50% assembled isoforms	1199	4489	1252
50-75% assembled isoforms	1137	1932	743
75-95% assembled isoforms	1579	2038	1204
95-100% assembled isoforms	3680	5704	1695
Mean isoform assembly	0.808	0.696	0.728
0-50% matched transcripts	221	1312	57
50-75% matched transcripts	418	1347	108
75-95% matched transcripts	1373	2690	384
95-100% matched transcripts	7792	12367	5166
Unannotated	44	2629	50
Mean fraction of transcript matched	0.944	0.772	0.964

Table 2.31: **rnaQUAST evaluation results for mouse dataset SAMEA3374581, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a mouse dataset. Metrics descriptions are the same as in Table 2.7.

Metrics	Scallop-LR	StringTie	Iso-Seq
# Transcripts	8071	18712	4748
Aligned	8071	18712	4748
Uniquely aligned	8069	18543	4740
Unaligned	0	0	0
Misassemblies	1	1	0
0-50% assembled isoforms	1115	4837	1344
50-75% assembled isoforms	1081	1980	619
75-95% assembled isoforms	1335	1892	919
95-100% assembled isoforms	2718	4581	1187
Mean isoform assembly	0.784	0.657	0.674
0-50% matched transcripts	98	1091	36
50-75% matched transcripts	243	1166	43
75-95% matched transcripts	905	2275	170
95-100% matched transcripts	6806	11737	4444
Unannotated	17	2439	55
Mean fraction of transcript matched	0.962	0.779	0.97

Table 2.32: **rnaQUAST evaluation results for mouse dataset SAMEA3374582, comparing Scallop-LR, StringTie, and Iso-Seq Analysis.** This table compares the rnaQUAST evaluation results for Scallop-LR, StringTie, and Iso-Seq Analysis on a mouse dataset. Metrics descriptions are the same as in Table 2.7.

Dataset	BioSample	SRA Study	Organism	Year	Instrument
1	SAMN00001694	ERP015321	*Homo sapiens*	2016	RS
2	SAMN00001695	ERP015321	*Homo sapiens*	2016	RS
3	SAMN00001696	ERP015321	*Homo sapiens*	2016	RS
4	SAMN00006465	ERP015321	*Homo sapiens*	2016	RS
5	SAMN00006466	ERP015321	*Homo sapiens*	2016	RS
6	SAMN00006467	ERP015321	*Homo sapiens*	2016	RS
7	SAMN00006579	ERP015321	*Homo sapiens*	2016	RS
8	SAMN00006580	ERP015321	*Homo sapiens*	2016	RS
9	SAMN00006581	ERP015321	*Homo sapiens*	2016	RS
10	SAMN08182059	SRP126849	*Homo sapiens*	2017	RS II
11	SAMN08182060	SRP126849	*Homo sapiens*	2017	RS II
12	SAMN04563763	SRP071928	*Homo sapiens*	2016	RS II
13	SAMN07611993	SRP098984	*Homo sapiens*	2018	RS II
14	SAMN04169050	SRP068953	*Homo sapiens*	2016	RS II
15	SAMN04251426.1	SRP065930	*Homo sapiens*	2016	RS II
16	SAMN04251426.2	SRP065930	*Homo sapiens*	2016	RS II
17	SAMN04251426.3	SRP065930	*Homo sapiens*	2016	RS II
18	SAMN04251426.4	SRP065930	*Homo sapiens*	2016	RS II
19	SAMEA3374575	ERP010189	*Mus musculus*	2015	RS
20	SAMEA3374576	ERP010189	*Mus musculus*	2015	RS
21	SAMEA3374577	ERP010189	*Mus musculus*	2015	RS
22	SAMEA3374578	ERP010189	*Mus musculus*	2015	RS
23	SAMEA3374579	ERP010189	*Mus musculus*	2015	RS
24	SAMEA3374580	ERP010189	*Mus musculus*	2015	RS
25	SAMEA3374581	ERP010189	*Mus musculus*	2015	RS
26	SAMEA3374582	ERP010189	*Mus musculus*	2015	RS

Table 2.33: **SRA information for the 26 datasets used in this study.** This table summarizes the 26 datasets used in this study. 18 datasets are human and eight datasets are mouse. The data were downloaded from the corresponding SRA Study. The multiple SRA Runs for each BioSample under the corresponding SRA Study were extracted, processed, and then merged into a large dataset.

Datasets	Sensitivity (%)		Precision (%)		# Total Multi-Exon Transcripts		# Correctly Predicted Known Transcripts	
	Scallop-LR	Scallop	Scallop-LR	Scallop	Scallop-LR	Scallop	Scallop-LR	Scallop
SAMN00001694	5.50	5.38	38.47	34.57	24903	27070	9580	9357
SAMN00001695	5.36	5.29	42.96	36.82	21731	25013	9336	9211
SAMN00001696	4.48	4.48	47.29	41.01	16476	19030	7791	7805
SAMN00006465	5.32	5.36	46.28	40.69	20019	22950	9264	9339
SAMN00006466	5.05	5.05	48.42	38.91	18171	22608	8798	8797
SAMN00006467	4.62	4.58	50.57	39.95	15900	19953	8040	7972
SAMN00006579	5.19	5.13	43.52	35.87	20742	24908	9027	8934
SAMN00006580	4.87	4.81	45.89	35.43	18467	23638	8474	8376
SAMN00006581	5.09	5.07	43.31	35.03	20458	25194	8861	8826
SAMN08182059	5.29	5.13	36.34	31.37	25332	28454	9206	8927
SAMN08182060	5.52	5.41	43.59	36.01	22038	26129	9606	9410
SAMN04563763	4.87	4.92	46.90	41.09	18078	20850	8478	8567
SAMN07611993	7.60	7.53	28.97	27.02	45657	48531	13226	13113
SAMN04169050	6.86	6.63	30.90	25.72	38657	44848	11946	11536
SAMN04251426.1	5.70	5.57	29.32	28.25	33824	34343	9916	9701
SAMN04251426.2	5.76	5.66	29.92	28.83	33534	34155	10034	9846
SAMN04251426.3	5.81	5.75	30.09	29.42	33637	33995	10122	10001
SAMN04251426.4	5.78	5.67	30.59	29.32	32908	33647	10065	9864

Table 2.34: **Performance comparison of Scallop-LR vs. Scallop on human data.** This table compares the performance of Scallop-LR (v0.9.1) with the performance of Scallop (v0.10.3) using the Gffcompare evaluation. The same 18 human PacBio datasets as described in Table 2.1 were evaluated. The parameter settings (options) used for Scallop (v0.10.3) are "–max_num_cigar 1000" and "–min_num_hits_in_bundle 1".

Datasets	# Total Multi-Exon Transcripts		# Correctly Predicted Known Transcripts		% of Correctly Assembled Known Transcripts Missing Due to Clustering	% of Nearly Redundant Transcripts Removed by Clustering
	Scallop-LR with Clustering	Scallop-LR without Clustering	Scallop-LR with Clustering	Scallop-LR without Clustering		
SAMN00001694	24903	26853	9580	9770	1.94	10.3
SAMN00001695	21731	23508	9336	9479	1.51	11.65
SAMN00001696	16476	17695	7791	7937	1.84	11.0
SAMN00006465	20019	21504	9264	9452	1.99	10.76
SAMN00006466	18171	19513	8798	8955	1.75	11.22
SAMN00006467	15900	17007	8040	8157	1.43	11.19
SAMN00006579	20742	22229	9027	9201	1.89	10.08
SAMN00006580	18467	19745	8474	8622	1.72	10.16
SAMN00006581	20458	21994	8861	9033	1.9	10.52
SAMN08182059	25332	27415	9206	9340	1.43	10.78
SAMN08182060	22038	24449	9606	9793	1.91	15.17
SAMN04563763	18078	19493	8478	8634	1.81	11.59
SAMN07611993	45657	50082	13226	13549	2.38	11.23
SAMN04169050	38657	43806	11946	12188	1.99	15.52
SAMN04251426.1	33824	36534	9916	10083	1.66	9.61
SAMN04251426.2	33534	36225	10034	10234	1.95	9.58
SAMN04251426.3	33637	36190	10122	10283	1.57	9.23
SAMN04251426.4	32908	35450	10065	10287	2.16	9.22

Table 2.35: **Comparison of Scallop-LR with clustering vs. Scallop-LR without clustering on human data.** This table compares the results of Scallop-LR without post-assembly clustering with the results of Scallop-LR with post-assembly clustering (using the default value for parameter "–max_cluster_intron_distance") by using the Gffcompare evaluation. The same 18 human PacBio datasets as described in Table 2.1 were evaluated. The percentages are computed as the following:

"% of Correctly Assembled Known Transcripts Missing Due to Clustering" = 100 × (("# matching transcripts without clustering" – "# matching transcripts with clustering") / "# matching transcripts without clustering").

"% of Nearly Redundant Transcripts Removed by Clustering" = 100 × (("# non-matching transcripts without clustering" – "# non-matching transcripts with clustering") / "# non-matching transcripts without clustering").

Where "# non-matching transcripts" = "# Total Multi-Exon Transcripts" – "# matching transcripts"; "# matching transcripts" = "# Correctly Predicted Known Transcripts".

Dataset	Sensitivity (%)		Precision (%)		# Total Multi-Exon Transcripts		# Correctly Predicted Known Transcripts	
	Scallop-LR	StringTie	Scallop-LR	StringTie	Scallop-LR	StringTie	Scallop-LR	StringTie
Simulated Human CCS Reads	53.05	45.03	66.54	63.51	6226	5538	4143	3517

Table 2.36: **Simulated Human Data: Sensitivity, Precision, Correctly Predicted Known Transcripts, and Total Multi-Exon Transcripts of Scallop-LR and StringTie.** This table compares the Gffcompare evaluation results for Scallop-LR and StringTie on a simulated human dataset [57]. The transcriptome that was used to generate the simulated long reads originated from the Ensembl annotation *Homo sapiens* GRCh38.94 and was a subset of the transcripts in this Ensembl annotation, by removing unfinished scaffolds, transcripts shorter than 200 bp, and annotations with an unknown reference, and randomly selecting alternative-splicing genes, single-splicing genes, and genes with small exons (< 31bp). The PacBio PBSIM tool was used to generate the simulated CCS reads from this transcriptome. The simulation was model-based using the CCS model, and three runs of simulations were performed by using three different sequencing depths 4X, 10X and 30X respectively. We merged the CCS reads generated with the three sequencing depths together to obtain this simulated human dataset. We used the transcripts in the transcriptome sequences that were used to generate the simulated CCS reads to extract the transcripts' records and their corresponding genes' records from the Ensembl annotation *Homo sapiens* GRCh38.94 to obtain an annotation GTF file. This extracted annotation GTF file serves as the reference in Gffcompare and the "ground truth", and it contains 7810 multi-exon transcripts.

Metrics	Scallop-LR	StringTie
# Transcripts	6228	7246
Aligned	6228	7246
Uniquely aligned	6225	7120
Unaligned	0	0
Misassemblies	0	0
0-50% assembled isoforms	90	41
50-75% assembled isoforms	171	48
75-95% assembled isoforms	253	170
95-100% assembled isoforms	3103	4248
Mean isoform assembly	0.956	0.984
0-50% matched transcripts	107	41
50-75% matched transcripts	418	217
75-95% matched transcripts	1668	1039
95-100% matched transcripts	4027	4979
Unannotated	5	967
Mean fraction of transcript matched	0.924	0.81

Table 2.37: **rnaQUAST evaluation results for a simulated human dataset, comparing Scallop-LR and StringTie.** This table compares the rnaQUAST evaluation results for Scallop-LR and StringTie on a simulated human dataset. The same simulated human dataset as described in Table 2.36 was evaluated. The same extracted annotation GTF file as described in Table 2.36 was used to generate the gene annotation database, which was used by rnaQUAST. Metrics descriptions are the same as in Table 2.7.

Chapter 3

Representative set selection of RNA-seq samples using a hierarchical approach

A version of this chapter was published in *ISMB/Bioinformatics* [101] and is joint work with Carl Kingsford.

3.1 Background

A vast number of RNA-seq short-read samples are publicly available at large sequence databases (e.g. NIH's Sequence Read Archive [51], known as SRA). However, most bioinformatics tools for RNA-seq analyses are evaluated on a limited number of samples; this evaluation may be insufficient, as the tools may not be adequately evaluated by samples with a variety of cell/tissue types and disease conditions. To ensure general applicability, an RNA-seq analysis tool should be validated on varying cell/tissue types and experiments. On the other hand, using all available RNA-seq samples to evaluate RNA-seq analysis tools is infeasible, and many samples in databases are similar to each other. This leads to a need to select a representative subset from available RNA-seq samples that effectively summarizes a large collection of RNA-seq samples to capture various essential transcriptional phenomena. Moreover, bioinformatics tools have algorithm parameters to be optimized, and automatic learning of optimal parameters can be made more robust using representative samples. Thus, our objective is to develop a computational method of selecting a representative subset from a large collection of RNA-seq short-read samples for a given organism (e.g. human), such that RNA-seq analysis tools can be effectively evaluated on this subset. Bioinformatics tools such as transcript assemblers, read mappers, and expression abundance estimators would benefit from a good selection of RNA-seq samples in their evaluation and parameter optimization.

Various representative set selection methods that solve the problem of finding a subset of data points (representatives) to efficiently describe the original collection of data have been developed in the fields of computer vision, signal/image processing, information retrieval, and machine learning [9, 21, 25, 26, 27, 29, 72, 107]. These methods include clustering-based approaches [21, 27], sparse modeling-based algorithms [26, 107], Rank Revealing QR algorithms [9], etc. The representative subset selected by these approaches usually follows the density distribution of

the original set. However, for RNA-seq analyses, it is important to have the representative set approximately evenly span the space of the original data in order to represent rare cell types and conditions.

In the field of RNA-seq analysis, Hie et al. developed a geometric sketching algorithm [35] for single-cell RNA-seq, which summarizes the transcriptomic heterogeneity within a data set using a representative subset of cells to accelerate single-cell analysis. Using a covering algorithm that approximates the original data space as a union of equal-sized boxes, geometric sketching focuses on even coverage of the transcriptional space spanned by the original set, such that rare cell types can be sufficiently sampled and represented. While maintaining a similar density distribution to that of the original set is useful for video/photo summarizations, for RNA-seq analyses, even coverage of the transcriptional space is more important in order to represent rare cell types.

A Python package, apricot [84], has been developed for selecting representative subsets using submodular optimization. Based on the "diminishing returns" property, apricot maximizes a monotone submodular function's value to find a representative subset. Apricot uses a greedy algorithm that can find a subset whose objective value is guaranteed to be within a constant factor of the optimal subset. Using facility location, apricot maximizes the sum of similarities between each sample and its closest representative sample; as a result, the representative set that is selected by apricot approximately evenly spans the space of the original data, like geometric sketching. While geometric sketching requires knowing the samples' gene expression vectors, apricot can work with the similarity matrix between the samples directly.

A main challenge in representative set selection for RNA-seq samples is that the number of RNA-seq samples in large databases is huge and each RNA-seq sample takes up substantial disk space; therefore, it is impractical to download all RNA-seq sequences of all the samples available at a large database like the SRA due to limited disk space. To perform representative set selection directly, we need to obtain gene expression vectors of, or distances between, all available SRA samples; however, SRA streaming is also not feasible due to issues with paired-end reads.

Given this challenge, one might attempt to select a representative set without looking at the sequences of each RNA-seq sample and relying instead on each sample's metadata, for example, using NCBI's BioSample attributes [7] that affect gene expression levels to predict gene expression distances between RNA-seq samples. However, for most large RNA-seq collections, including the SRA, the metadata is highly incomplete and most samples do not have the needed metadata values for predicting their gene expression distances.

Thus, we use a sequence-based approach for representative set selection of RNA-seq samples. We randomly sample a small subset of reads from each RNA-seq sample to download, such that the subsets of reads from all available RNA-seq samples at the SRA take a reasonable disk space. We count k-mers in the subset of reads of each sample and compute the similarity between k-mer distributions of samples. This approach selects a representative set based on k-mer similarities and thus sequence similarities among RNA-seq samples. Since the number of publicly available RNA-seq samples in the SRA is large (N=196523 for human) and the number of k-mers in each sample is large ($\sim$2000000 k-mers), computing the 196523×196523 similarity matrix with k-mers has memory and runtime challenges even using a chunking method for matrix computation [54].

To tackle this challenge, we developed a novel method called "hierarchical representative set

selection." The hierarchical representative set selection is a divide-and-conquer-like algorithm that hierarchically selects representative samples through multiple levels. At each level, samples are divided into smaller chunks, and representative set selection is performed on each chunk with a weighting scheme. The representative samples selected from every chunk are merged into the next level, and the process repeats until the size of the similarity matrix of the merged samples is feasible for the computing resources.

Our results show that hierarchical representative set selection achieves summarization quality close to that of direct representative set selection using apricot (5.37% average difference in the measure of how well a selected subset represents the full set), while substantially reducing runtime and memory requirements of computing the full similarity matrix (up to 8.4X runtime reduction and 5.35X memory reduction for 10000 and 12000 samples respectively that could be practically run with direct subset selection), thus making selecting representative samples from the entire SRA RNA-seq samples feasible (the estimated runtime reduction is 90X and memory reduction is 41.4X for the SRA full set of 196523 human samples). We demonstrate that the representative subset selected by our hierarchical representative set selection method from all human RNA-seq samples in the SRA better represents the transcriptomic heterogeneity among those samples than that by random sampling, and thus can be used for more comprehensive and complete evaluation of bioinformatics tools.

3.2 Methods

3.2.1 Problem formulation

Let set R be a large set of RNA-seq samples (such as all the RNA-seq samples in the NIH SRA database for a given organism), let $d(i, j)$ be a distance or dissimilarity measure between samples i and j in R, and let $d(x, S)$ be the distance between a data point x and its closest data point in a set S. A reasonable formulation is to find a representative subset $\tilde{R} \subseteq R$, such that

$$\max_{r \in R} d(r, \tilde{R})$$

is as small as possible.

This is equivalent to minimizing the classical Hausdorff distance which is defined as: $d_H(X, S) = \max_{x \in X}\{\min_{s \in S} d(x, s)\}$ where X is the full set and S is its representative subset [35]. The Hausdorff distance can be used to evaluate how well a selected subset represents the original full set (a smaller value is better) [35]. However, the classical Hausdorff distance is highly sensitive to extreme outliers [37, 91]. Thus, in practice, a more robust measure, the partial Hausdorff distance, is used to evaluate the representative subset [35, 37]. The partial Hausdorff distance is defined as: $d_{HK}(X, S) = K^{th}_{x \in X}\{\min_{s \in S} d(x, s)\}$ where $K^{th}_{x \in X}$ is the K^{th} largest value (counting from the minimum), and a parameter $q = 1 - K/|X|$ is used to determine K (when $q = 0$, $d_{HK} = d_H$; when q is small enough, d_{HK} is very close to d_H but is robust to extreme outliers) [35, 37]. d_{HK} is at the $((1-q) \times 100)$-th percentile of the distances from every sample x in the full set to its closest representative sample.

3.2.2 K-mer similarity-based approach

The similarity between k-mer distributions of RNA-seq samples reflects the similarity between their sequences and is a reasonable approach for computing $d(i, j)$. Thus, by counting the k-mers of RNA-seq reads, we can select a representative set based on the k-mer similarities and therefore sequence similarities among samples. Downloading all reads of all SRA samples is infeasible, so we download a small subset of reads from each RNA-seq sample. To represent a full RNA-seq sample, the sampled small subset of reads are random with respect to the genome coordinates. For approximately 88% of SRA RNA-seq samples, the reads are stored from the sequencer without alignment. Reads coming from Illumina sequencers without alignment are random with respect to the genome coordinates. Thus, a range of reads downloaded from unaligned samples by *fastq-dump* are random. We download 10,000 reads to represent each unaligned RNA-seq sample (we skip the first 5000 reads in the sample when downloading, since the beginning of sequencing may contain some technical variation of signal introduced in the sequencing process).

Choosing a proper k-mer size is important, as smaller k-mers give less information about sequence similarities, while larger k-mers may result in fewer matches due to sequencing errors. To select an optimal k-mer size, we plot the number of distinct k-mers with the varying k-mer size for a range of typical read lengths of Illumina (Figs. 3.6–3.10). In the linearly increasing part of the curve, short k-mers match randomly; from the beginning of the horizontal part of the curve, k-mers start to reveal the genome structure. Thus, we want the smallest k-mer size in the horizontal part of the curve, so that the k-mer matching moves from being random to being representative of the read content and is still resilient to sequencing errors. Among the optimal k-mer sizes we obtained for long, medium, and short read-lengths, we choose a compromise 17 as the optimal k-mer size.

Jellyfish [61] was used to count k-mers in the subset of reads. We use canonical k-mers (i.e. the lexicographically smaller of a k-mer and its reverse complement), so all samples are compared based on the common k-mer sequences regardless of the sequenced strand. We use the cosine similarity as the similarity of k-mer distributions between samples. Cosine similarity is the cosine of the angle between two vectors and is commonly used in document clustering and information retrieval; our k-mer vectors have some similar aspects to TFIDF (Term Frequency–Inverse Document Frequency) vectors (e.g. large vocabulary, word counts, high dimension, and high sparsity). Cosine similarity is also a good measure for k-mer-based metagenome comparisons [19].

3.2.3 Hierarchical representative set selection algorithm

To handle the memory and runtime challenges, hierarchical representative set selection (as shown in Algorithm 1) breaks the whole representative set selection into multiple levels of progressive sub-selections, like divide-and-conquer. At each level, the "full set" of samples are divided into smaller equal-size chunks using a seeded-chunking method (see Algorithm 2), such that chunks are well separated with closer samples going into the same chunk if possible. The similarity matrix for each chunk is computed, and weights that determine the size of the representative set for every chunk are computed based on the density/sparseness of each chunk (samples in a denser chunk are more similar to each other). Representative set selection is then performed on

each chunk (for example, using apricot's facility location approach). The representative samples selected from every chunk are merged into a new set, which becomes the "full set" of the next level. This process iterates until the size of the similarity matrix of the merged set is feasible for the computing resources. Lastly, the similarity matrix of the final merged set is computed, and representative set selection is performed on it to get the final representative set of a desired size.

Algorithm 1: Hierarchical Representative Set Selection

Input: Full set R of N samples (k-mer counts). m: chunk size. Q: average representative-set size for each chunk. L: max size of similarity matrix (L×L) feasible for computing resources. n: user-desired size of final representative set.

Output: n representative samples.

1. Divide: divide R into $l = \lceil N/m \rceil$ chunks using a seeded-chunking method (see Algorithm 2), each has m samples.
2. Compute similarity matrix ($m \times m$) for each chunk i.
3. Compute weight w_i for each chunk i (see "Weighting scheme").
   Set the representative-set size $RSS_i = w_i Q$
4. For each chunk i, perform representative set selection on its $m \times m$ similarity matrix to get RSS_i representative samples.
5. Merge: sequentially merge RSS_i representative samples from every chunk into a set R' of $N' = \sum_i RSS_i$ samples.
6. $R \leftarrow R'$, repeat steps 1–5 until $N' \leq L$
7. Compute similarity matrix ($N' \times N'$); store it for any user-desired smaller-size representative set selection.
8. Perform representative set selection on the $N' \times N'$ similarity matrix to get n representative samples.

* Once step 7 is completed, step 8 can be independently performed repeatedly for different n.

Computing the similarity matrix with k-mers is the runtime and memory bottleneck, while apricot's runtime is relatively negligible in comparison. For direct representative set selection using apricot, the computational cost for the similarity matrix of the full set is $O(N^2)$. Using the hierarchical selection (considering one iteration of divide-and-merge with l chunks, chunk size m, and the final merged set size N'), the computational cost is reduced to $O(lm^2) + O(N'^2) = O(N^2/l) + O(N'^2)$. The seeded-chunking has an added computational cost $O(Nl)$. So the total computational cost of the hierarchical selection is $O(N^2/l) + O(N'^2) + O(Nl)$, where $l \ll N$ and $N' \ll N$. With multiple iterations, the computational cost is further reduced. Since $m \ll N$, the memory requirement for computing the similarity matrix is greatly reduced.

Fig. 3.1 illustrates applying the hierarchical representative set selection to 196523 human RNA-seq samples in the SRA, using two levels (two iterations) of divide-and-merge. The first level has $l_1 = 197$ chunks; the second level has $l_2 = 40$ chunks. ∼200 representative samples are selected from each chunk of 1000 samples.

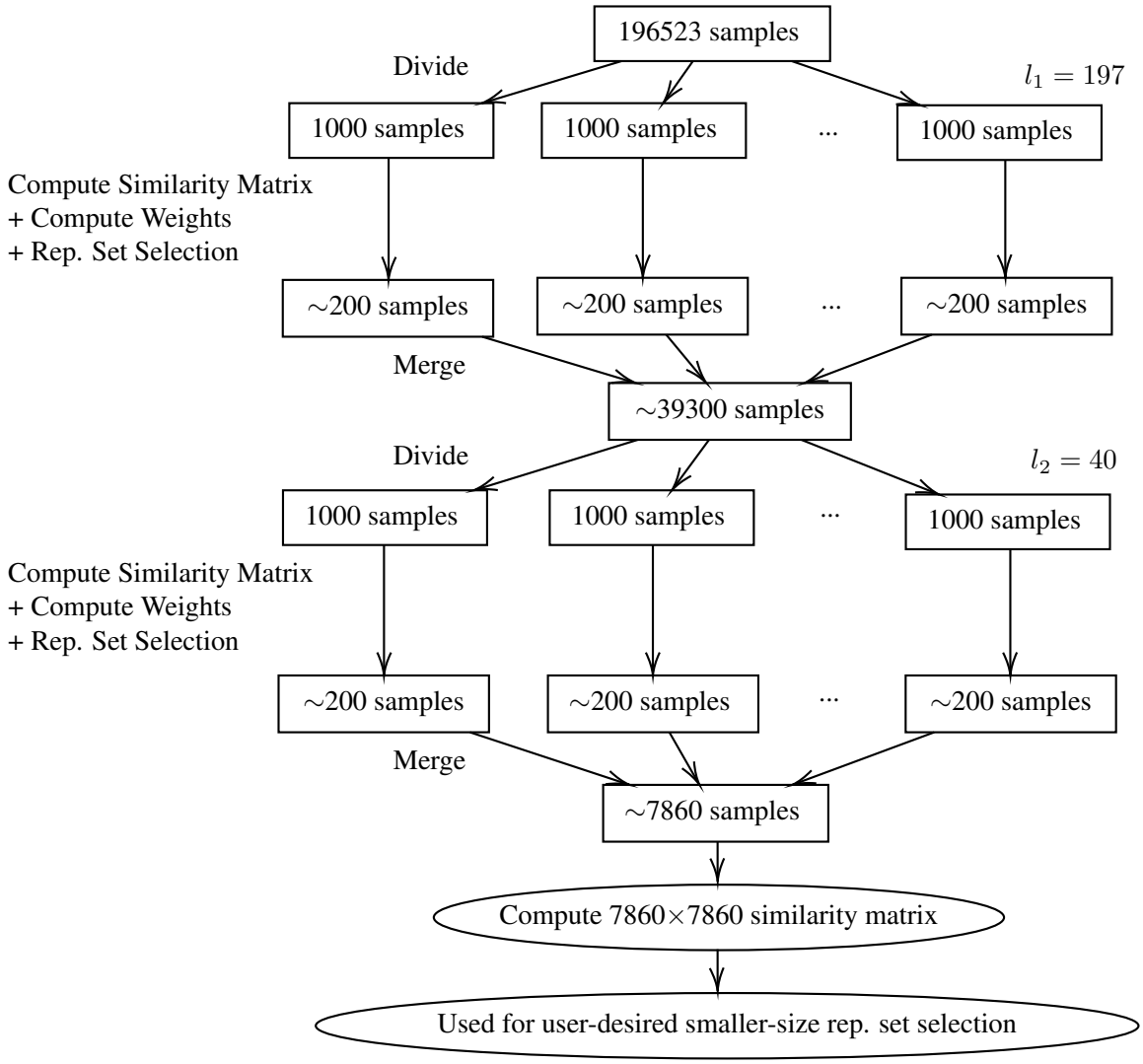


Figure 3.1: Illustration of the hierarchical representative set selection for 196523 human RNA-seq samples in the SRA.

Seeded-chunking method

The ideal situation for the hierarchical representative set selection is that chunks have no overlaps. Here, the "overlap" of chunks means data points and their close neighbors are in different chunks (e.g. different chunks occupy the same dense cluster). When chunks have no overlaps, the union of the representative sets selected from every chunk would be similar to the representative set selected directly from the original set. When chunks have overlapping regions with similar densities, unnecessarily more representative data points may be selected in the overlapping regions from different chunks. The subsequent representative set selection on the merged set can alleviate this effect, but the over-use of the quota (i.e. a region's proportion of the desired number of representative data points) in the overlapping regions may cause other regions to have less quota. Thus, more separated chunks lead to more accurate hierarchical selection.

A sequential-chunking method that divides chunks sequentially along the SRA accession list can cause many overlaps, even complete overlaps between chunks. To overcome the introduction of overlaps with sequential chunking, we developed a novel seeded-chunking method (as shown in Algorithm 2). The seeded-chunking first uses the "farthest point sampling" algorithm [10, 71] to find l seeds for l chunks, such that the seeds are farthest away from each other. That is, starting from a randomly selected seed, the seeds are chosen one at a time, such that each new seed has the largest distance to the set of already selected seeds (i.e. the largest minimum distance to the already selected seeds). The entire set of RNA-seq samples available at a large database cannot fit into memory at once, but if loading each sample one at a time when computing its distance to a seed, each sample would be repeatedly loaded for $l - 1$ times. Thus, we randomly select a subset X from the full set and perform the "farthest point sampling" on X to find l seeds. Each sample in the full set is assigned to its closest seed. Since chunks have equal sizes, the sample is assigned to its closest seed among all currently non-full chunks (i.e. their current size $< m$).

The seeded-chunking can generate well-separated chunks, with an added computational cost: $O(Nl)$ of computing similarities. Since $l \ll N$, this cost is fairly small compared to computing the full similarity matrix ($O(N^2)$). We benchmarked the seeded-chunking vs. sequential chunking on various sizes of full sets (Table 3.8). In all cases, the seeded-chunking outperforms the sequential chunking; as the full-set size increases, the seeded-chunking shows more advantages over the sequential chunking in the partial Hausdorff distance.

Weighting scheme

Denser chunks (in which samples are more similar to each other) should have fewer representative samples than sparser chunks, since we want representative data points to approximately evenly span the space of the original data, so that rare cell/tissue types can be sufficiently represented. Since chunks have equal sizes, denser chunks occupy smaller spaces than sparser chunks. Thus, we propose a novel weighting scheme ("mean^2-weighting scheme") to assign the representative-set size to each chunk based on their average density/sparseness.

The mean^2-weighting scheme is as follows. Let μ_i be the mean of distances between samples in chunk i; z_i is the size of chunk i (note that when N/m is not an integer, not all chunks are full); Q, l, and m are as defined in Algorithm 1. Suppose the *l*th chunk is non-full: let $\alpha_l = z_l/m$. The

Algorithm 2: Seeded-Chunking Method

Input: Full set R of N samples. m: chunk size. l: number of chunks. J: size of a randomly selected subset used for selecting seeds.

Output: l chunks.

1. Randomly select a subset X of J samples from R.
2. Perform the "farthest point sampling" on X to find a set S of l seeds:

   Initialization: Seed $s_1 =$ a randomly selected sample from X.
   $S = \{s_1\}$. Distance from $x \in X$ to S:
   $d_S(x) = distance(x, s_1)$

   For $i = 2, \ldots, l$, repeat steps (a)–(c):

   (a) Find the farthest sample away from the already selected seeds' set S:
   $s_i = \arg\max d_S(x)$, $x \in X$

   (b) Add s_i as a new seed into S.

   (c) Update the distance from $x \in X$ to S:
   $d_S(x) \leftarrow \min\{d_S(x), distance(x, s_i)\}$
3. Compute distances between each sample in R and l seeds.
4. Assign each sample in R to its closest seed:

   (i) Find all currently non-full chunks (i.e. size $< m$).

   (ii) Assign the sample to its closest seed among all non-full chunks.

weight w_i and the representative-set size RSS_i for chunk i are defined to be:

$$w_i = \mu_i^2 / weighted\_mean\{\text{all } \mu_j^2\} \qquad \text{where } i, j = 1, \ldots, l \tag{3.1}$$

$$RSS_i = w_i Q \qquad \text{for } i = 1, \ldots, l-1 \tag{3.2}$$

$$RSS_l = w_l \alpha_l Q \tag{3.3}$$

where the $weighted\_mean\{all\ \mu_j^2\}$ uses $weight = 1$ for $j = 1, \ldots, l-1$ and $weight = \alpha_l$ for $j = l$. The following relationship holds:

$$\sum_i RSS_i = (l-1)Q + \alpha_l Q.$$

With the seeded-chunking, there could be multiple non-full chunks, but the same weighting method applies. The mean^2-weighting scheme is a heuristic to adjust the representative-set size for each chunk according to their average density/sparseness.

The code for the hierarchical representative set selection is available at https://github.com/Kingsford-Group/hierrepsetselection; the code for computing k-mer similarities is available at https://github.com/Kingsford-Group/jellyfishsim. We have two GitHub repositories since the 2nd repository contains the code that can also be used for other k-mer similarity applications; other applications that need to compute k-mer similarities between samples can use the 2nd repository only.

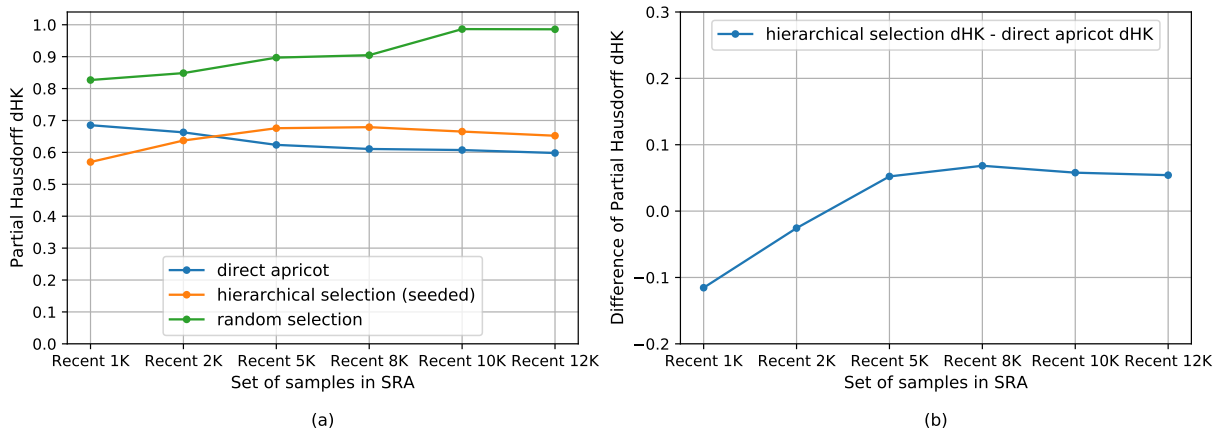


Figure 3.2: Using the most recent 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets. rep_set_size/N=0.1. (a) Partial Hausdorff distances d_{HK} of direct apricot, hierarchical selection, and random selection. (b) Partial Hausdorff distances' difference: hierarchical selection d_{HK} − direct apricot d_{HK}.

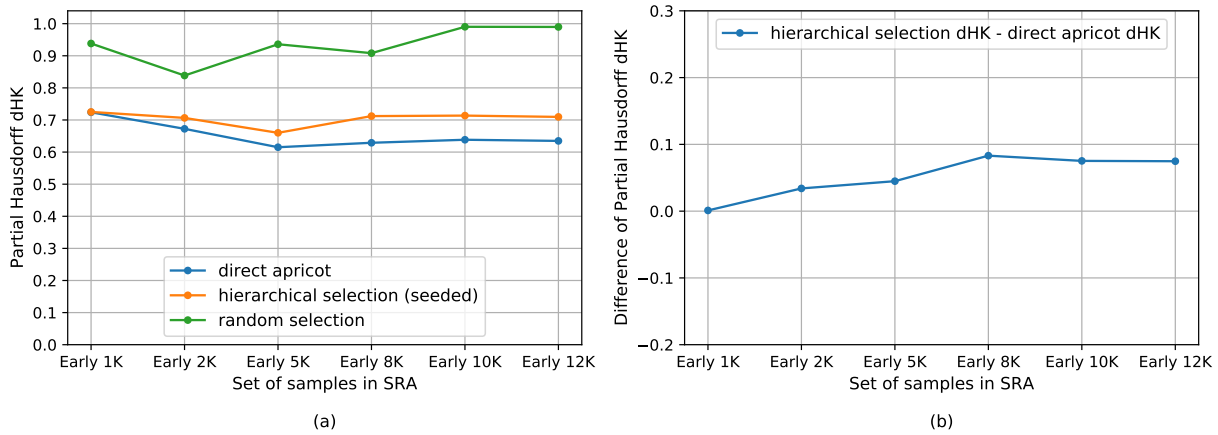


Figure 3.3: Using the early-time 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets. rep_set_size/N=0.1. (a) Partial Hausdorff distances d_{HK} of direct apricot, hierarchical selection, and random selection. (b) Partial Hausdorff distances' difference: hierarchical selection d_{HK} − direct apricot d_{HK}.

3.3 Results

3.3.1 Hierarchical selection achieves summarization quality close to that of direct representative set selection

Excluding SRA samples with no public access permission, aligned samples, and samples with no valid 17-mers (read-lengths $<$ the k-mer size 17, or reads that contain many N's in the middle), we obtained 196523 human bulk RNA-seq (Illumina) samples as the SRA entire set. Each sample corresponds to an SRA Experiment.

In this particular implementation of the hierarchical representative set selection, we use apricot's facility location approach as the base level to perform representative set selection on each chunk and on the merged set. We use cosine distance as the distance measure. In this context, we define the term "summarization quality" as a measure of how well a selected subset represents the full set (i.e. "representativeness"), evaluated by d_{HK}.

We refer to the direct representative set selection using apricot as "direct apricot", which includes two parts: (a) computing the similarity matrix of the full set; (b) applying apricot's facility location approach to the full similarity matrix. The main computational cost of direct apricot comes from part (a).

The motivation of the hierarchical representative set selection is to reduce the runtime and memory requirement of direct representative set selection, while not sacrificing too much summarization quality. Thus, we compared the performance between direct apricot, the hierarchical selection, and random sampling, using the most recent 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets (Fig. 3.2, Table 3.4). We also compared the three methods using the early-time 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets (in the earliest quarter of the SRA time span, i.e. the 4th quarter of the SRA accession list) (Fig. 3.3, Table 3.5), and using the mid-time 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets (around the middle of the SRA time span) (Fig. 3.4, Table 3.6). For all these cases: we use 1 iteration, $rep\_set\_size/N = 0.1$, $m = N/l$, $Q = m/5$. We set $l = 10$, except for $N = 1000$, $l = 5$.

The hierarchical selection achieves summarization quality close to that of direct apricot. For the recent 1000 and 2000 samples, the hierarchical selection performs better than direct apricot; for the recent 5000, 8000, 10000, and 12000 samples, the hierarchical selection is modestly less accurate than direct apricot but has representativeness close to that of direct apricot (Fig. 3.2). As N increases while keeping the same rep_set_size/N, the d_{HK} difference of the hierarchical selection minus direct apricot initially increases from negative values, then levels out, and then slightly decreases when N becomes very large, indicating that the d_{HK} difference between the hierarchical selection and direct apricot does not get larger when N further increases (Fig. 3.2). For the early and middle sets of samples, the trend of d_{HK} of the hierarchical selection is more similar to that of direct apricot, so their d_{HK} difference curves are flatter (Figs. 3.3 and 3.4). For the early sets, the d_{HK} difference increases initially, and then slightly decreases and levels out when N becomes very large (Fig. 3.3). For the middle sets, the d_{HK} difference also eventually decreases when N becomes very large (Fig. 3.4). Overall, the average sacrifice in representativeness (% increase in d_{HK}) of the hierarchical selection vs. direct apricot is 5.37%. These demonstrate that the hierarchical selection achieves summarization quality close to that of direct

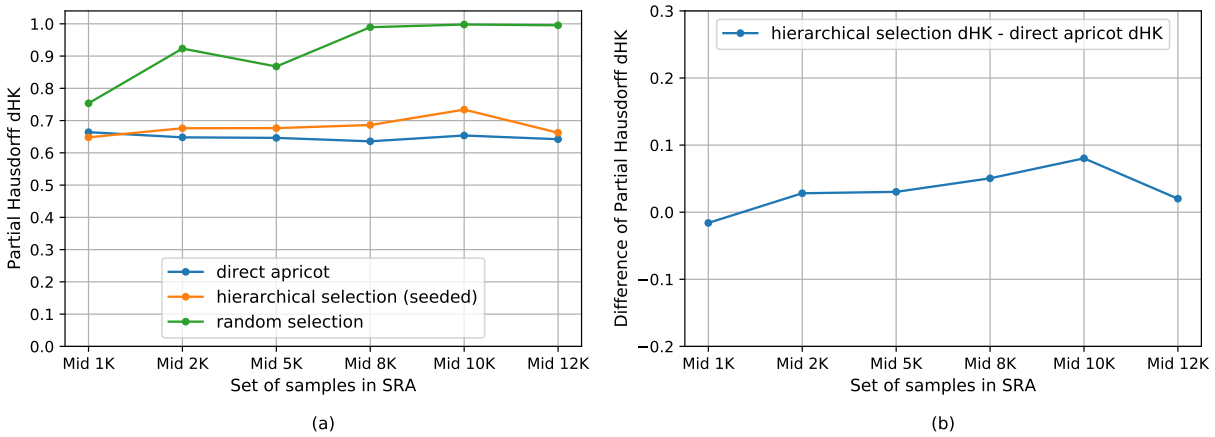


Figure 3.4: Using the mid-time 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets. rep_set_size/N=0.1. (a) Partial Hausdorff distances d_{HK} of direct apricot, hierarchical selection, and random selection. (b) Partial Hausdorff distances' difference: hierarchical selection d_{HK} − direct apricot d_{HK}.

apricot, which is partially contributed by the seeded-chunking and mean^2-weighting.

The hierarchical selection substantially outperforms random sampling. Random sampling has substantially larger d_{HK} values than the hierarchical selection for all the recent, early, and middle sets of samples, and this trend is consistent (Figs. 3.2, 3.3, and 3.4). When the full set becomes very large, random sampling's d_{HK} approaches 1.0 that is the maximum cosine distance. Random sampling follows the density distribution of the original set, so the rare cell/tissue types are not sufficiently represented, which yields large d_{HK}.

3.3.2 Hierarchical selection substantially reduces runtime and memory of direct representative set selection

The hierarchical selection substantially reduces the runtime and memory usage of direct apricot. For the recent, early, and middle sets of samples, the hierarchical selection and direct apricot were all run using 85 cores for parallelism; their runtime and memory are reported in Tables 3.4, 3.5, and 3.6. As the full-set size increases, the real time and user+system time reductions generally increase (Tables 3.1, 3.2, and 3.3). The user+system time reduction can reach 8.4X and the real time reduction can reach 7.47X when $N = 10000$. Real time reductions are less than user+system time reductions, since in direct apricot the full similarity matrix computation is fully parallel, while in the hierarchical selection although the similarity matrix computation for each chunk is fully parallel, chunks are processed sequentially to reduce the memory usage. Runtime reductions for most recent samples are generally greater than those for early- and mid-time samples, as longer reads generate more k-mers. The memory reduction also generally increases as the full-set size increases (Tables 3.1, 3.2, and 3.3). The memory reduction can reach 5.35X when $N = 12000$.

For the entire set of SRA RNA-seq samples ($N = 196523$), we use two levels of divide-and-merge (Fig. 3.1), with $m = 1000$, $Q = m/5 = 200$. At the 1st-level, $l_1 = 197$, the subset-size

Table 3.1: Runtime and memory reduction with the hierarchical selection over direct apricot using the most recent 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets.

Reduction	Recent 1000 (select 100, $l = 5$)	Recent 2000 (select 200, $l = 10$)	Recent 5000 (select 500, $l = 10$)	Recent 8000 (select 800, $l = 10$)	Recent 10000 (select 1000, $l = 10$)	Recent 12000 (select 1200, $l = 10$)
Real time Reduction	1.47 X	2.72 X	6.04 X	6.53 X	7.02 X	7.17 X
User+Sys time Reduction	2.99 X	5.76 X	8.33 X	8.17 X	8.4 X	8.19 X
Memory Reduction	2.13 X	2.9 X	4.47 X	4.59 X	4.54 X	5.22 X

Table 3.2: Runtime and memory reduction with the hierarchical selection over direct apricot using the early-time 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets.

Reduction	Early 1000 (select 100, $l = 5$)	Early 2000 (select 200, $l = 10$)	Early 5000 (select 500, $l = 10$)	Early 8000 (select 800, $l = 10$)	Early 10000 (select 1000, $l = 10$)	Early 12000 (select 1200, $l = 10$)
Real time Reduction	1.53 X	2.14 X	4.62 X	4.99 X	7.47 X	6.35 X
User+Sys time Reduction	3.05 X	4.89 X	5.88 X	6.2 X	8.31 X	7.06 X
Memory Reduction	2.2 X	2.83 X	4.33 X	4.37 X	4.68 X	5.0 X

$J_1 = 2000$ (used for the seeded-chunking). At the 2nd-level, $l_2 = 40$, $J_2 = 1000$.

The hierarchical representative set selection makes selecting representative samples from the entire set of SRA RNA-seq samples feasible. When $N = 10000$, direct apricot by computing the full $N \times N$ similarity matrix takes 4.1 hours (using 85 cores) and 118.906 GB memory (Table 3.4). With the $O(N^2)$ time complexity and $O(N)$ space complexity, for the entire SRA set ($N = 196523$), the estimated runtime of direct apricot is 66 days (using 85 cores) and the estimated memory usage is 2336.776 GB, which is infeasible. The hierarchical selection on the entire SRA set ($N = 196523$) takes 17.6 hours (using 85 cores) and 56.430 GB memory, which makes this task fully feasible.

3.3.3 Hierarchical selection outperforms random sampling for the entire set of SRA RNA-seq samples

The hierarchical representative set selection outperforms random sampling for the entire set of SRA RNA-seq samples. We compare the hierarchical selection with random sampling by selecting different sizes of representative sets from the entire SRA set (Fig. 3.5, Table 3.7). The hierarchical selection outperforms random selection in all these cases (Fig. 3.5); when selecting 7000 representative samples, the hierarchical selection outperforms random selection substantially, with a similar level of difference to those of smaller full sets. The d_{HK} values of the hierarchical selection are larger than that of selecting 1000 from the recent 10000 samples, mainly because the ratios $rep\_set\_size/full\_set\_size$ are much smaller here, one magnitude smaller than the ratio 0.1 in selecting 1000 from 10000. As the size of the representative set increases, the representativeness of the hierarchical selection initially barely changes but increases when going from selecting 5000 to selecting 7000 representative samples. Random sampling's d_{HK} are almost 1.0 (the maximum cosine distance) and do not change as the size of the representative set increases, indicating its poor performance on the entire SRA set.

Table 3.3: Runtime and memory reduction with the hierarchical selection over direct apricot using the mid-time 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets.

Reduction	**Mid 1000 (select 100, $l = 5$)**	**Mid 2000 (select 200, $l = 10$)**	**Mid 5000 (select 500, $l = 10$)**	**Mid 8000 (select 800, $l = 10$)**	**Mid 10000 (select 1000, $l = 10$)**	**Mid 12000 (select 1200, $l = 10$)**
Real time Reduction	0.94 X	2.24 X	4.92 X	5.66 X	6.92 X	6.45 X
User+Sys time Reduction	3.13 X	5.5 X	7.12 X	7.1 X	7.79 X	7.19 X
Memory Reduction	2.33 X	2.78 X	4.58 X	4.5 X	4.55 X	5.35 X

The final representative sets of different sizes are available at the GitHub repository: https://github.com/Kingsford-Group/hierrepsetselection.

3.4 Discussion

Our results show that the hierarchical representative set selection is a close estimate to the direct representative set selection, while substantially reducing the runtime and memory usage of the direct selection, which makes subset selections feasible on big data such as the entire available RNA-seq samples in the SRA.

The chunk-size m needs to be large enough to avoid chunk overlaps in the seeded-chunking (Table 3.10). If the chunk-size is too small, a large dense blob containing many more samples than the chunk-size could have multiple chunks overlapping there. Chunks need to have equal sizes (except for 1 or 2 chunks when N/m is not an integer) to fit to the resource capacity, so when the chunk of the closest seed is full, the data point has to be assigned to its closest non-full seed. So although the seeds are spread out, if the chunk-size is not large enough, there could still be chunk overlaps in the seeded-chunking.

The choices of the parameters m (which yields l), Q, and the number of iterations (levels) depend on the full-set size N and the memory of available computing resources. These parameters can affect the selection accuracy, runtime, and memory usage of the hierarchical selection. In all the results here, we use $Q = m/5$, which means $rep\_set\_size/chunk\_size = \sim 0.2$ for each chunk. Increasing this ratio could increase the selection accuracy at each chunk, however, the subsequent merged set would be larger, causing more chunks at the next level and thus increasing the runtime. Decreasing m (and thus increasing l) could reduce the runtime, however, smaller chunks have more overlaps which decrease the overall selection accuracy (Table 3.10). Thus, the overall design of the hierarchy with parameters' choices involves trade-offs between selection accuracy, runtime, and the resource capacity.

The mean^2-weighting uses the average distance between samples of a chunk to indicate the chunk overall density. This is a heuristic. In a case that a chunk has several dense clusters that are far apart, causing a bigger average distance than the distances within clusters, the chunk may be assigned an unnecessarily larger weight. This may be partially addressed by performing clustering on each chunk and using the weighted-mean of average distances of all clusters as the chunk density. However, an accurate clustering incurs more computational cost. In our observation, most chunks do not have a distinctly clustered structure, and rather have a mixture of a few clusters and many roughly uniformly distributed data points. Thus, the mean^2-weighting

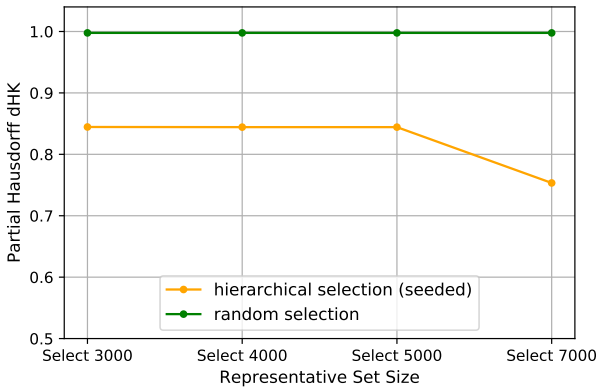


Figure 3.5: Selecting different sizes of representative sets from the SRA entire set (N=196523 human RNA-seq samples): partial Hausdorff distances d_{HK} of hierarchical selection and random selection. For d_{HK}, $q = 0.0001$, so d_{HK} is the 21st-largest distance.

is a viable trade-off between runtime and selection accuracy.

In addition to the partial Hausdorff distance, a useful evaluation for a representative set could be using the representative set as the training set to train classifiers to compare the classification accuracy. The classifiers could take the gene expression vectors or transcript abundance vectors as input features. We could compare the classification accuracy of the models trained by using different representative sets. This is a direction for future work.

Additional discussion can be found in Appendix Section 3.5.2.

3.5 Appendix

3.5.1 Additional tables and figures

In Tables 3.4, 3.5, and 3.6, for N=1000, 2000, 5000, d_{HK} is the 3rd largest distance; for N=8000, 10000, d_{HK} is the 4th largest distance; for N=12000, d_{HK} is the 5th largest distance. The results are from single runs.

The SRA accession list of the full set obtained using the Entrez API is in the reverse order of accession numbers.

Table 3.4: Partial Hausdorff distance, classical Hausdorff distance, runtime, and memory usage of direct apricot, hierarchical selection, and random selection, using the most recent 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets.

Set of samples	**Method**	**Hausdorff d_H**	**Partial Hausdorff d_{HK}**	**Runtime (seconds)**			**Memory (GB)**
				Real	**User**	**Sys**	
Recent 1000	direct apricot	0.688321022	0.685257115	174.58	11070.16	111.42	17.131
(select 100,	hierarchical (seeded)	0.578381536	0.569809642	118.91	3449.28	292.08	up to 8.061
$l = 5$)	random selection	0.837951394	0.826812255	0.424	2.282	5.205	negligible
Recent 2000	direct apricot	0.664517495	0.662815581	655.18	42882.64	286.79	29.223
(select 200,	hierarchical (seeded)	0.650712434	0.637185175	240.77	6906.76	586.80	up to 10.077
$l = 10$)	random selection	0.894947772	0.848467165	0.421	2.157	5.323	negligible
Recent 5000	direct apricot	0.624796548	0.62353618	4334.92	306588.88	1323.23	67.515
(select 500,	hierarchical (seeded)	0.696114667	0.675768259	717.42	35803.41	1164.84	up to 15.115
$l = 10$)	random selection	0.973292492	0.896965855	0.423	2.241	5.242	negligible
Recent 8000	direct apricot	0.610919893	0.610735987	10095.85	701276.54	1513.85	101.776
(select 800,	hierarchical (seeded)	0.760030601	0.679049314	1545.26	84481.00	1588.09	up to 22.169
$l = 10$)	random selection	0.991018754	0.904692245	0.536	2.347	5.129	negligible
Recent 10000	direct apricot	0.607682609	0.607369442	14768.33	1047257.38	2273.86	118.906
(select 1000,	hierarchical (seeded)	0.70951932	0.665272021	2103.18	123182.85	1749.27	up to 26.200
$l = 10$)	random selection	0.994547626	0.986377775	0.439	2.420	5.079	negligible
Recent 12000	direct apricot	0.598587653	0.598052637	20902.48	1449102.48	2202.62	142.083
(select 1200,	hierarchical (seeded)	0.770540655	0.652204188	2914.35	175006.16	2103.21	up to 27.207
$l = 10$)	random selection	0.994469360	0.985777326	0.49	2.57	4.97	negligible

Table 3.5: Partial Hausdorff distance, classical Hausdorff distance, runtime, and memory usage of direct apricot, hierarchical selection, and random selection, using the early-time 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets.

Set of samples	**Method**	**Hausdorff d_H**	**Partial Hausdorff d_{HK}**	**Runtime (seconds)**			**Memory (GB)**
				Real	**User**	**Sys**	
Early 1000	direct apricot	0.725003411	0.723976221	153.22	7701.56	125.76	11.084
(select 100,	hierarchical (seeded)	0.745518016	0.725044029	100.12	2351.73	215.52	up to 5.038
$l = 5$)	random selection	0.950139546	0.938242439	0.43	2.27	5.21	negligible
Early 2000	direct apricot	0.67307199	0.672427473	401.49	22979.13	218.19	17.131
(select 200,	hierarchical (seeded)	0.760020571	0.706453874	187.64	4360.50	383.16	up to 6.046
$l = 10$)	random selection	0.872261407	0.838447692	0.65	2.43	5.07	negligible
Early 5000	direct apricot	0.61634404	0.615099659	2298.45	133723.63	565.12	39.300
(select 500,	hierarchical (seeded)	0.666160711	0.659981919	497.60	21561.05	1262.98	up to 9.069
$l = 10$)	random selection	0.938632619	0.935902009	0.45	2.37	5.14	negligible
Early 8000	direct apricot	0.630098995	0.628989605	5719.26	382221.70	874.59	70.538
(select 800,	hierarchical (seeded)	0.834517734	0.71208656	1145.94	60263.98	1509.99	up to 16.123
$l = 10$)	random selection	0.9916713	0.908157434	0.78	1.55	3.35	negligible
Early 10000	direct apricot	0.639209628	0.6383828	12315.44	800933.84	1446.46	89.684
(select 1000,	hierarchical (seeded)	0.837916798	0.713645489	1648.26	95117.21	1381.86	up to 19.146
$l = 10$)	random selection	0.99772421	0.990244824	1.38	2.18	2.97	negligible
Early 12000	direct apricot	0.635004017	0.634814991	14554.59	984029.09	1247.92	110.845
(select 1200,	hierarchical (seeded)	0.908677303	0.709577599	2290.33	137885.59	1681.81	up to 22.169
$l = 10$)	random selection	0.999260054	0.989801358	0.53	2.34	5.25	negligible

Table 3.6: Partial Hausdorff distance, classical Hausdorff distance, runtime, and memory usage of direct apricot, hierarchical selection, and random selection, using the mid-time 1000, 2000, 5000, 8000, 10000, 12000 samples in the SRA as the full sets.

Set of samples	**Method**	**Hausdorff d_H**	**Partial Hausdorff d_{HK}**	**Runtime (seconds)**			**Memory (GB)**
				Real	**User**	**Sys**	
Mid 1000	direct apricot	0.666185918	0.664052819	150.21	9130.84	126.60	14.108
(select 100,	hierarchical (seeded)	0.685598613	0.648069325	160.45	2705.68	252.78	up to 6.046
$l = 5$)	random selection	0.968587397	0.753610206	0.43	2.45	5.03	negligible
Mid 2000	direct apricot	0.651423605	0.648054888	545.46	34587.00	248.80	25.192
(select 200,	hierarchical (seeded)	0.732927883	0.676261249	243.70	5827.03	508.06	up to 9.069
$l = 10$)	random selection	0.967519071	0.923284417	0.43	2.15	5.24	negligible
Mid 5000	direct apricot	0.646540816	0.646398158	3084.32	209037.81	892.53	55.422
(select 500,	hierarchical (seeded)	0.719052575	0.676710291	626.75	28509.77	983.47	up to 12.092
$l = 10$)	random selection	0.961274547	0.867699133	0.45	2.34	5.17	negligible
Mid 8000	direct apricot	0.636313583	0.635731787	7142.40	464515.34	865.10	81.622
(select 800,	hierarchical (seeded)	0.943464491	0.686230795	1262.73	64248.73	1301.38	up to 18.138
$l = 10$)	random selection	0.998728362	0.989508253	2.43	3.14	2.91	negligible
Mid 10000	direct apricot	0.654537313	0.653742747	12582.81	780770.97	1574.01	100.768
(select 1000,	hierarchical (seeded)	0.807148615	0.734005	1818.98	98855.02	1548.71	up to 22.169
$l = 10$)	random selection	0.999886845	0.997812734	0.45	2.42	4.99	negligible
Mid 12000	direct apricot	0.642698074	0.642257142	16318.41	1063199.56	1357.40	123.945
(select 1200,	hierarchical (seeded)	0.758132270	0.662355963	2529.71	146293.35	1731.62	up to 23.177
$l = 10$)	random selection	0.999041296	0.995426952	0.49	2.35	5.20	negligible

Table 3.7: Selecting different sizes of representative sets from the SRA entire set (N=196523 human RNA-seq samples): partial Hausdorff distance and classical Hausdorff distance of hierarchical selection and random selection.

Metric	**Select 3000**		**Select 4000**		**Select 5000**		**Select 7000**	
	hierarchical (seeded)	random selection	hierarchical (seeded)	random selection	hierarchical (seeded)	random selection	hierarchical (seeded)	random selection
Hausdorff d_H	0.945978504	0.998763361	0.945978504	0.998763361	0.945978504	0.998763361	0.875009817	0.998763361
Partial Hausdorff d_{HK}	0.844519904	0.997721045	0.844274154	0.997718649	0.844274154	0.997718649	0.753391502	0.997718289
Representative -set-size/ Full-set-size	0.0153		0.0204		0.0254		0.0356	

Table 3.8: Performance comparison of hierarchical selection using seeded-chunking method vs. using sequential chunking method: partial Hausdorff distance and classical Hausdorff distance, using the most recent 1000, 2000, 5000, 8000, 10000 samples in the SRA as the full sets.

Set of samples	**Method**	**Hausdorff d_H**	**Partial Hausdorff d_{HK}**
Recent 1000	hierarchical (seeded)	0.578381536	0.569809642
(select 100, $l = 5$)	hierarchical (sequential)	0.612622717	0.578381536
Recent 2000	hierarchical (seeded)	0.650712434	0.637185175
(select 200, $l = 10$)	hierarchical (sequential)	0.656343899	0.643755957
Recent 5000	hierarchical (seeded)	0.696114667	0.675768259
(select 500, $l = 10$)	hierarchical (sequential)	0.72711968	0.708127649
Recent 8000	hierarchical (seeded)	0.760030601	0.679049314
(select 800, $l = 10$)	hierarchical (sequential)	0.770009257	0.754661415
Recent 10000	hierarchical (seeded)	0.70951932	0.665272021
(select 1000, $l = 10$)	hierarchical (sequential)	0.790246213	0.752879052

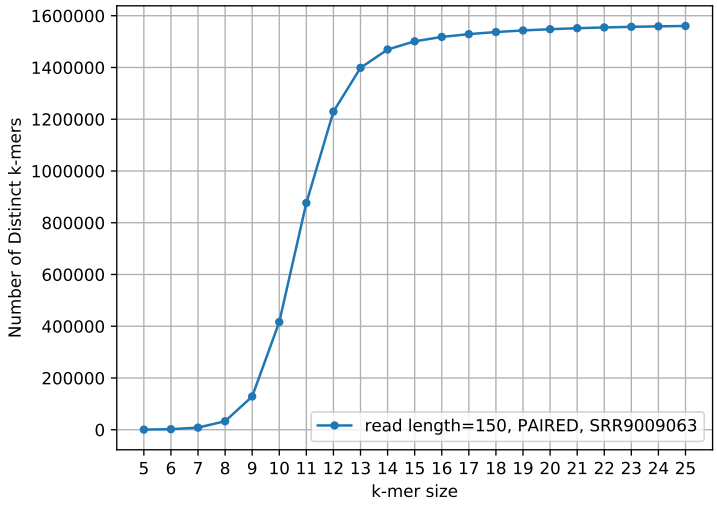


Figure 3.6: The number of distinct k-mers vs. k-mer size: read-length=150, paired-end reads, from SRR9009063 (10000 random reads).

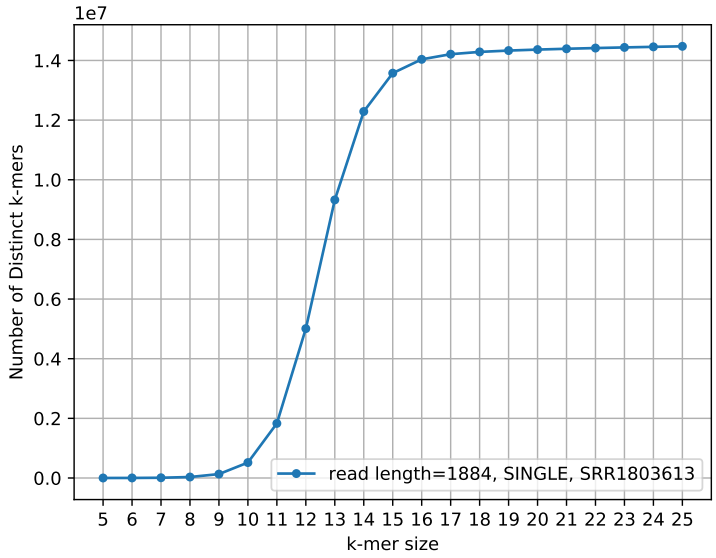


Figure 3.7: The number of distinct k-mers vs. k-mer size: read-length=1884, single-end reads, from SRR1803613 (10000 random reads).

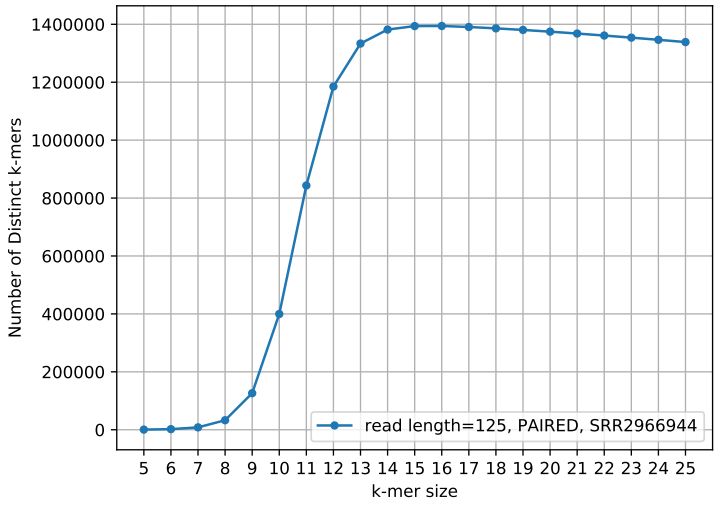


Figure 3.8: The number of distinct k-mers vs. k-mer size: read-length=125, paired-end reads, from SRR2966944 (10000 random reads).

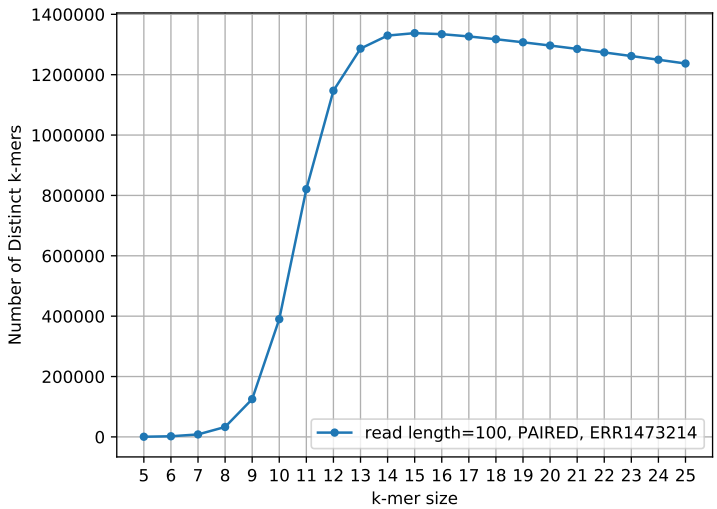


Figure 3.9: The number of distinct k-mers vs. k-mer size: read-length=100, paired-end reads, from ERR1473214 (10000 random reads).

Table 3.9: Hardware specifications of the system on which the experiments were run.

Attribute	**Value**
Architecture	x86_64
CPU op-mode(s)	32-bit, 64-bit
Byte Order	Little Endian
CPU(s)	88
On-line CPU(s) list	0-87
Thread(s) per core	2
Core(s) per socket	22
Socket(s)	2
NUMA node(s)	2
Vendor ID	GenuineIntel
CPU family	6
Model	79
Model name	Intel(R) Xeon(R) CPU E5-2699A v4 @ 2.40GHz
Stepping	1
CPU MHz	2871.660
CPU max MHz	3600.0000
CPU min MHz	1200.0000
BogoMIPS	4793.95
Virtualization	VT-x
Mem Total	1056631884 kB

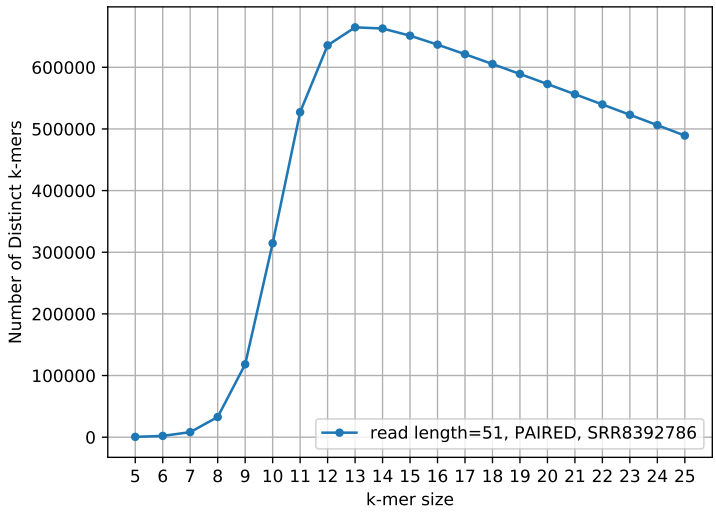


Figure 3.10: The number of distinct k-mers vs. k-mer size: read-length=51, paired-end reads, from SRR8392786 (10000 random reads).

In Figs. 3.8, 3.9, and 3.10, the horizontal part of the curve bends down as k-mer size further increases, especially for shorter read-lengths, since when read-lengths are short, using larger k-mers would reduce the number of distinct k-mers.

Table 3.10 shows the impact of the chunk size m on the selection accuracy of hierarchical selection, illustrating that chunk size m needs to be large enough to avoid chunk overlaps in the seeded-chunking. When $m = 100$, hierarchical selection performs worse than direct apricot. When $m = 150$, hierarchical selection performs almost as equally as (slightly better than) direct apricot. When $m = 200$, hierarchical selection performs better than direct apricot. Smaller

Table 3.10: Performance comparison of using different chunk size m in hierarchical selection, using the most recent 1000 samples in the SRA as the full set. Partial Hausdorff distance and classical Hausdorff distance.

Metric	**hierarchical (seeded)** $m = 100$	**hierarchical (seeded)** $m = 150$	**hierarchical (seeded)** $m = 200$	**direct apricot**
Hausdorff d_H	0.744303019	0.687825966	0.578381536	0.688321022
Partial Hausdorff d_{HK}	0.725728616	0.684475004	0.569809642	0.685257115

chunks have more overlaps which decrease the selection accuracy.

3.5.2 Additional discussion

We compared the performance between direct apricot and hierarchical selection by using the most recent, early-time, and mid-time RNA-seq samples in the SRA as the full sets. A more robust comparison would be computing confidence intervals for partial Hausdorff distances for each N value and compare the confidence intervals between different methods. Computing confidence intervals would involve running direct apricot and hierarchical selection on a substantial number of randomly selected subsets (used as the "full sets") across the SRA spectrum for each N value. Each randomly selected subset starts from a randomly selected position on the SRA accession list and contains N consecutive samples starting from that position; the subset should not contain randomly selected N samples, since we use the subset as the full set (we found that randomly selected N samples have a much smaller number of samples that are similar to each other and thus are not suitable to be used as a "full set"). This is a direction for future analysis.

We download 10,000 reads from each RNA-seq sample to represent that sample. We chose this number since prior researchers found that 10,000 reads are sufficient to tell what genome this sample belongs to, and we do not have extra disk space to hold more reads from all RNA-seq samples in the SRA. We envision that downloading more reads (such as 100,000 reads) would represent each sample better, and thus the selected representative samples based on the subsets of reads would better represent the SRA full set. In order to evaluate the effect of downloading different numbers of reads, one would need to download all reads (rather than the subset of reads) from each of the RNA-seq samples in the full set. Currently, the partial Hausdorff distance is computed based on the k-mer similarities using the subset of reads; this is effective for assessing the representative set selection algorithm itself. However, to assess the effect of downloading different numbers of reads, the partial Hausdorff distance would need to be computed using all reads from each of the RNA-seq samples in the full set. In this assessment, although the representative set is selected based on the subset of reads (10K or 100K reads), in order to compare which representative set represents the RNA-seq full set better, one would need to use all reads from each sample to compute the partial Hausdorff distance. The comparison between the d_{HK} value computed using 10K reads and the d_{HK} value computed using 100K reads would not be meaningful since they do not share the common ground. Thus, in this specific case, only the comparison between the d_{HK} values computed using all reads of each sample can tell which

representative set represents the RNA-seq full set better. Let us show this proposed evaluation process in more detail. First, one would select the representative set using 10K reads from each sample; one would then compute the d_{HK} based on the k-mer similarities using all reads from each of the selected representative samples and all reads from each sample in the full set. Second, one would select the representative set using 100K reads from each sample; one would then compute the d_{HK} based on the k-mer similarities using all reads from each of the selected representative samples and all reads from each sample in the full set. Then, the d_{HK} values of these two cases can be compared to assess the effect of downloading 10K reads vs. downloading 100K reads. This is a direction for future analysis when extra disk space becomes available for downloading all reads from each of the RNA-seq samples in the full set.

Bioinformatics tools should be validated and optimized on varying cell/tissue types and experiments. Thus, our method takes in all available RNA-seq samples regardless of their tissue/cell types, treatments, or experiments to select a representative set. However, our method can also be adapted for selecting representative samples for specific tissues/cell types. This only needs to be done at the full set construction stage. Our current SRA full set contains all RNA-seq samples (excluding those non-public, aligned, or with no valid 17-mers). From the full SRA accession list, one can extract those RNA-seq samples of specific tissues/cell types based on their BioSample Attributes. These extracted samples of specific tissues/cell types can be used as the full set, and then our hierarchical representative set selection can be performed to get representative samples for specific tissues/cell types.

When we download the subset of reads for each RNA-seq sample, we skip the first 5000 reads, since the beginning of sequencing may contain some technical variation of signal introduced in the sequencing process, as suggested by prior researchers. We also filter out the technical reads, remove those tags' sequences, and filter out reads that are all N's. When we do k-mer counting using Jellyfish, k-mers with N's in the middle are skipped; however, bases with low quality scores are still counted into k-mers, as Jellyfish does not read or consider the quality scores. However, using k-mers instead of the original sequences to compute the similarity between samples is more resilient to sequencing errors. Our choice of optimal k-mer size 17 is to ensure that the k-mer matches move from random to a representative of reads' content and are still resilient to sequencing errors. One potential improvement that could be made in Jellyfish would be taking into account the base call quality scores for each read to filter out those k-mers containing many low-quality bases (below a quality score threshold). We may also filter out or trim those reads with many low-quality bases (below a quality score threshold) before doing k-mer counting. These are directions for future work.

The hierarchical selection is more accurate than direct apricot for the most recent 1000 and 2000 samples and the mid-time 1000 samples. For the early-time 1000 samples, the hierarchical selection is nearly as accurate as direct apricot. It seems that for smaller N's, the hierarchical selection may have some advantage. A possible reason for the hierarchical selection being more accurate than direct apricot in these cases is that the chunks generated by the seeded-chunking method may have minimal or no overlaps. When chunks have no overlaps, the union of the representative sets selected from every chunk would be similar to the representative set selected directly from the original full set. Given that chunks have no overlaps, since the hierarchical selection performs the 2nd-round of representative set selection – selecting representative samples from the merged set (in the one-level hierarchy), it could be possible that the final representa-

tive set is better than the representative set selected directly from the original full set, since the direct selection only performs one round of selection. This is also supported by Table 3.10: for the same "Recent 1000" samples, when chunks are large enough to avoid overlaps ($m = 200$; $m = 150$), the hierarchical selection is more accurate than direct apricot; when the chunk size is small ($m = 100$) such that there are chunk overlaps, the hierarchical selection is less accurate than direct apricot. The selection accuracy loss of the hierarchical selection mainly comes from chunk overlaps. In our observation, smaller full sets (e.g. $N = 1000$) are more likely to have some clustered structures than large full sets, and thus would be more likely to generate chunks with no overlaps (when the chunk size is large enough).

When we get the SRA full set, we exclude those non-public samples as we do not have access to them; we are testing on what is publicly available. It is also more convenient for tools' developers to use publicly available representative samples to evaluate bioinformatics tools. It is possible that there might be a difference in sample distribution if those non-public samples were in the full set. By the same reasoning, some non-public samples might not be sufficiently represented by the representative set selected from publicly available samples only.

Chapter 4

De novo error correction for RNA-seq long reads using deep learning

A version of this chapter is in preparation for submission to a journal/conference and is joint work with Carl Kingsford.

4.1 Background

Third-generation sequencing has become increasingly important in transcriptome analyses such as isoform identification, study of alternative splicing, characterization of complex transcriptional events, and study of transcription initiation. Due to its low cost, portability, extended read-lengths, and improved throughput [108], Nanopore sequencing has become a compelling choice for long-read RNA-seq. However, Nanopore long reads have high error rates (per-base error rates vary and are up to $\sim$14% [56, 81]), which can affect transcriptome studies. Hence, computational methods are needed for error correcting Nanopore long reads.

Error correction methods for genomic long reads that use only long reads (i.e. self-correction) have been developed, such as Canu's error correction module [43], LoRMA [83], MECAT [114], and CONSENT [65], which are designed for or applicable to Nanopore genomic reads to reduce their error rates. There are also hybrid correction methods for Nanopore genomic long reads [42, 60, 82] that require using short reads, and signal-based Nanopore genomic error correction tools [59, 106] that require raw electric signals (measurement of current) [77]. However, Lima et al. [56] and Sahlin et al. [81] found that applying genomic error correctors to RNA-seq long reads has problematic effects—genomic correctors split reads in low coverage regions; they decrease the number of detected genes, the number of isoforms, and the number of detected splice sites, and change the isoform landscape by removing/adding exons. Therefore, genomic error correction tools are not suitable for transcriptomic long reads [81], and error correction methods that are designed for RNA-seq long reads are needed.

Two reference-based error correction methods that are specifically designed for transcriptomic long reads, TranscriptClean [113] and FLAIR [94], were designed to correct base errors and/or splice sites for RNA-seq long reads, but require using a reference genome. However, for many non-model organisms, *de novo* (reference-free) error correction is required when a high-

quality reference is unavailable. Recently, two *de novo* error correction methods specifically designed for Nanopore RNA-seq long reads, isONcorrect [81] and RATTLE [22], were developed. Both isONcorrect and RATTLE are graph-based self-correction methods that use partial order alignment (POA) for multiple sequence alignment (MSA) and consensus generation. Both methods use clustering to group reads from the same gene/isoform and correct reads based on each cluster. isONcorrect divides a read into intervals, and performs POA on the supporting reads' segments of each interval to create a consensus; it identifies "trusted variants" with high enough support, and corrects the read segment by using the trusted variant with surrounding bases closest to those in the read [81]. RATTLE performs POA on whole reads rather than segments in a cluster to generate a consensus; it changes the read base to the consensus if the consensus frequency is above a threshold and the read base's error probability is high enough compared to the consensus [22]. The two tools achieve similar reductions in the overall error rates [81].

The error profiles of ONT have not been thoroughly reported in the literature yet. Nevertheless, prior literature found that high error rates of ONT are partially due to the base-calling errors in homopolymer regions—the errors (mainly deletions) in homopolymer regions are substantially higher than in non-homopolymer regions [23, 116]. Prior literature also found that, in ONT reads, "A→T" and "T→A" substitutions are much less likely to occur than other substitutions [50]. Transitions (A↔G, T↔C) are more frequent than transversions (A↔C, A↔T, G↔C, G↔T) in ONT reads [23]. The chance of errors is higher near the ends of a read, and low-GC-content species have lower error rates than high-GC-content species [23]. These findings suggest that the error profile information (i.e. factors associated with error occurrences or error types) could be applied systematically to improve error correction. Although there are some specific considerations that are informed by error profiles (e.g. setting thresholds for consensus frequency, choosing trusted context, etc.), RATTLE and isONcorrect have not systematically taken into account the error profile information.

Thus, our objective is to develop an error-profile-aware, generalizable correction method, which not only uses supporting reads to generate the consensus but also learns the error profile information systematically. Here, we present a novel error correction method, called deepCorrRNA, for self-correcting RNA-seq long reads *de novo* that combines the graph-based MSA-POA and an error-profile-aware deep neural network. In this method, POA is performed to obtain the consensus; our neural network model takes in both the observed read and the consensus sequence from the POA, such that the model learns the relationship between the observed read, the consensus sequence, and the true read. The model also learns context dependencies along the whole read. Our model also uses the error profile associated information and the MSA-matrix associated information (see descriptions in "Method overview" in Section 4.2.1), so that the predicted correction is inferred from the combination of both the MSA-POA result (which addresses the support from other reads) and the error profile related information (which traditional graph-based self-correction methods did not systematically include). The model is trained by using an organism with a good reference genome, and the model can be used to perform *de novo* error correction for organisms without a good reference.

We evaluate our method, deepCorrRNA, on five ONT RNA-seq (cDNA) data sets (one human, three mouse, and one Drosophila data sets) using NanoSim [116], and compare the error correction results with the state-of-the-art *de novo* error corrector for ONT RNA-seq reads, isONcorrect. Our results show that deepCorrRNA achieves similar reductions in the total error rates

to isONcorrect. For all five data sets, deepCorrRNA performs better than isONcorrect on reducing the substitution and insertion rates, while isONcorrect performs better than deepCorrRNA on reducing the deletion rate. In three out of the five data sets, deepCorrRNA has slightly lower post-correction total error rates than isONcorrect, while in two out of the five data sets, deepCorrRNA has slightly higher post-correction total error rates than isONcorrect; nevertheless, their values are similar. While the model of deepCorrRNA is trained with the human data, its correction performance on the mouse and Drosophila data demonstrates deepCorrRNA's transferability and *de novo* error correction capability, which can improve transcriptome analysis for non-model organisms. With the comparable reductions of the total error rates to the state-of-the-art ONT-specific corrector isONcorrect, deepCorrRNA presents a generalizable method.

4.2 Methods

4.2.1 Method overview

Fig. 4.1 illustrates an overview of the deepCorrRNA method. In this method, we first cluster the reads such that the reads that came from the same isoform approximately go into the same cluster. We then perform graph-based MSA-POA on the cluster—the set of supporting reads for the read of interest, to get the consensus sequence and the MSA alignment matrix. Our neural network model takes both the observed read sequence and the consensus sequence from the POA as inputs in parallel, and the two sequences pass through RNNs simultaneously; the extracted sequential features from parallel RNNs are fed into a series of fully connected layers to output the probability for each class, and the model is trained against the true read to minimize the loss. Here, the "true read" is the sequence on the reference genome that the read is aligned to. Thus, the model learns the relationship between the observed read, the consensus sequence, and the true read. Since the model is trained along the whole read through RNNs, it also learns the context dependencies along the entire read for each base. The MSA alignment matrix associated information, such as the occurrence frequencies and the average quality scores for every base in the consensus sequence, are also fed into the RNN simultaneously, such that the level of support and the quality of each consensus base are taken into account when predicting the correct base. The model also takes in the error profile associated information, such as the base position, the base quality, the read length, surrounding k-mers, GC content, etc. Thus, the predicted correction is inferred from the combination of both the MSA-POA result and the error profile related information.

4.2.2 Features and classes of the model

A series of factors may be related to an error occurred at a particular position in a read, such as: (1) Error position in the read. The chance of errors could differ at the beginning, middle, and end of the sequence. The quality of ONT reads drops at both ends (around the first 10 bases and last 20 bases) due to less stable signals [23]. (2) Base at the error position. (3) Base quality score at the error position. Base quality scores in FASTQ encode the error probability for each base. (4) Read length. Longer reads may have a higher error rate. (5) Motif surrounding the

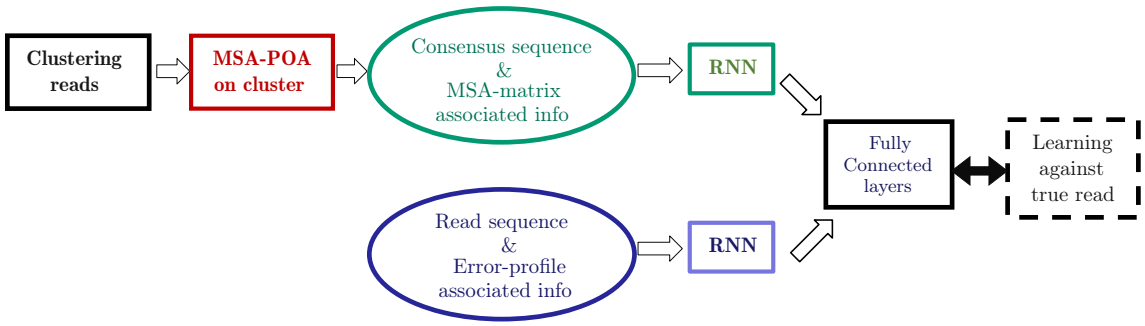


Figure 4.1: Method overview of deepCorrRNA.

error position [50]. In ONT reads, deletions in homopolymer regions are the source of about half of all sequencing errors. The indels in homopolymer regions are mainly deletions, although there are some long insertions [23]. (6) Platform used. (7) Chemistry used in sequencing. (8) Protocol kit used for sequencing. (9) Basecaller used. (10) Organism. The error rates differ across different species. Low-GC-content species have lower error rates than high-GC-content species in ONT reads [23]. (11) Substitution, insertion, and deletion rates. (12) Substitution-rate confusion matrix for every possible base substitution [50]. In ONT reads, transitions are more frequent than transversions [23]. (13) Length distributions of deletions, insertions, and substitutions [116]. (14) The set of reads overlapping with the read of interest; their base quality scores, mapped positions, mapped bases, and gaps. (15) Translocation speed and signal-to-noise ratio in raw electrical signals [23].

Therefore, we design the following features for the network model:

1. Base position in the read. The position uses the aligned coordinate of the read, including deletions, with the first base position $= 0$. The aligned coordinate is based on the pair-wise alignment between the read and the consensus sequence. We use relative position (*absolute-position* / *sequence-length*).
2. Base character at this position. They are the four regular base letters or deletion '-'.
3. Base quality score at this position. For regular base letters: use base quality scores in the FASTQ file; for deletions: use the lowest score (0) in the score scale.
4. Read length. This is the actual read length, not including deletions.
5. Sequences surrounding this base position: (a) k-bases before this position; (b) k-bases after this position. These left and right k-mers are the actual sequences in the read, not including deletions.
6. GC content rate of the read.
7. Consensus base at this position. They are the four regular base letters or deletion '-'.
8. Occurrence frequency of this consensus base in the MSA-matrix column.
9. Average base quality score of this consensus base across its occurrences in the MSA-matrix column. For deletion '-', we use the mean of average base quality scores of those regular

base-letter columns.

10. Number of supporting reads in MSA-POA. This is the total number of reads in the MSA-POA set.
11. Strand of the read on cDNA.

For the surrounding *k*-mers, we use $k = 5$ by default, since Delahaye et al. [23] found that considering 5-mers before or after the error position yields useful results when studying harmful *k*-mers (*k*-mers that are frequently associated with a type of error).

The model is a 10-class classifier. The output prediction is a sequence of class labels corresponding to every base position on the read. Each of the 10 classes represents one type of error on the read with respect to the true read (including no error). The following are the 10 classes:

1. Insertion: this read base is an insertion to the true read.
2. Substituted A: this read base substituted a base 'A' on the true read.
3. Substituted C: this read base substituted a base 'C' on the true read.
4. Substituted G: this read base substituted a base 'G' on the true read.
5. Substituted T: this read base substituted a base 'T' on the true read.
6. Deleted A: this read position deleted a base 'A' from the true read.
7. Deleted C: this read position deleted a base 'C' from the true read.
8. Deleted G: this read position deleted a base 'G' from the true read.
9. Deleted T: this read position deleted a base 'T' from the true read.
10. No error: this read base is the same as the true read base.

4.2.3 Neural network model architecture

Fig. 4.2 illustrates the neural network model architecture. The model has five input branches, each first passing through a masking layer (the details about padding and masking are discussed in Section 4.2.8). The two time-distributed LSTM (i.e. hierarchical LSTM) blocks are for extracting the features of the two surrounding *k*-mers for each base in the read, as these sequences of surrounding *k*-mers are two-levels of sequential features. The outputs from the two hierarchical LSTMs are concatenated with the read base-related features (read base character, position, and quality), and then go into the left Bi-LSTM branch. The read and the consensus sequence pass through two parallel branches of Bi-LSTMs—the left branch of Bi-LSTM is for extracting the read base-related sequential features, and the right branch of Bi-LSTM is for extracting the consensus base-related sequential features (consensus base character, occurrence frequency, and average quality). The extracted features from the two Bi-LSTMs are concatenated together along with the remaining read-level non-sequential features (read length, GC content, number of supporting reads, and strand), and then go through multiple fully connected layers to get the output prediction. The fully connected layers are time distributed, so that the model outputs predicted class probabilities for every base position in the read. The input to each Bi-LSTM or LSTM is batch normalized to stabilize and accelerate the training. To prevent overfitting, a dropout layer with a dropout rate 0.2 is added after each of the first two fully connected layers, and the two Bi-LSTMs also use dropout with a rate 0.1. Our experiments found that these

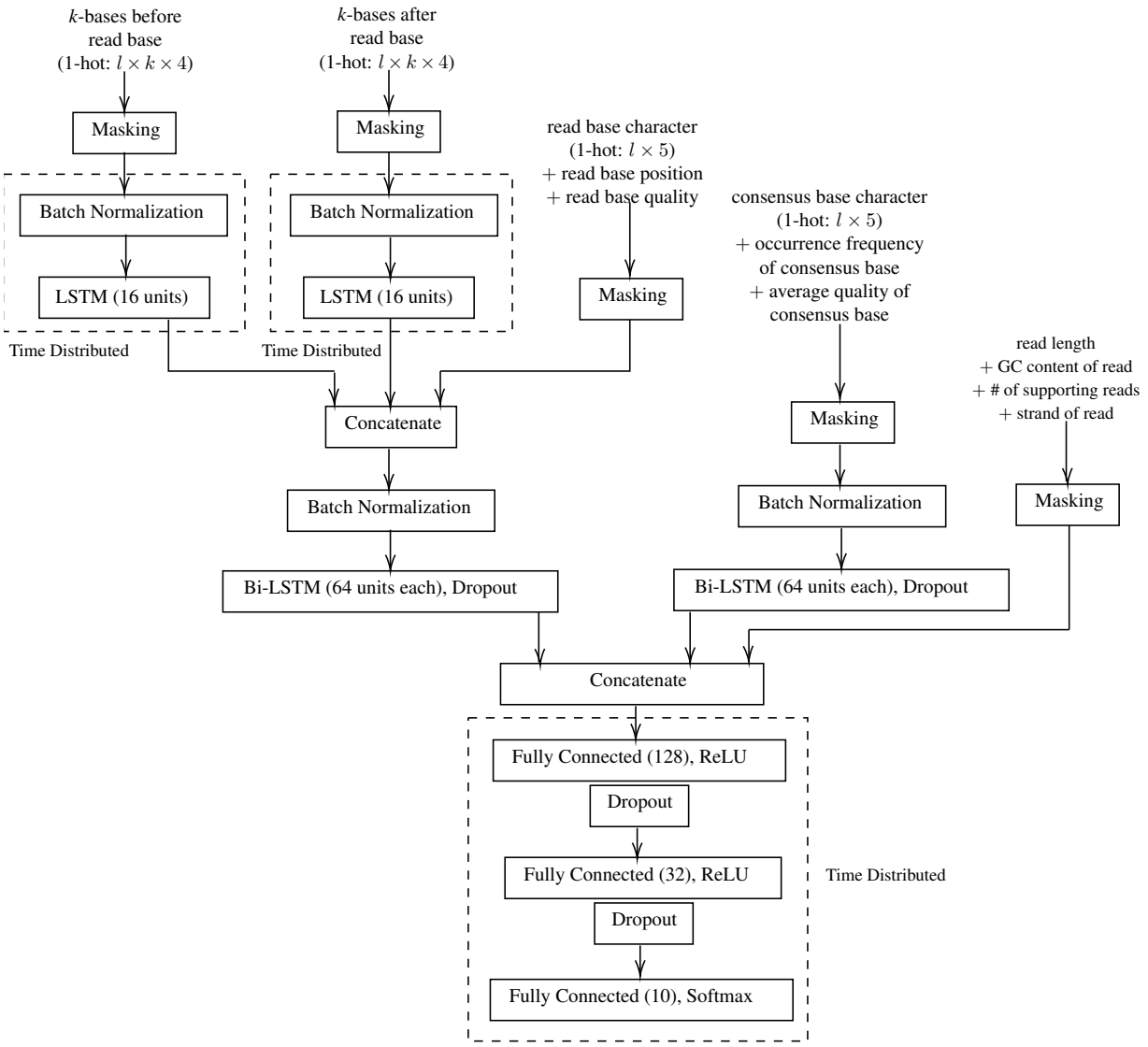


Figure 4.2: Neural network model architecture.

dropout rates effectively prevent overfitting. The final output layer uses softmax activation for the probabilities of the 10 classes; the remaining fully connected layers use ReLU activation. The LSTMs/Bi-LSTMs use tanh activation. The read bases and consensus bases are one-hot encoded with length 5, as they also include deletions '-' in addition to four regular base letters. The *k*-mers before and after each base are one-hot encoded with length 4, as they are physical sequences with only four regular base letters. Here, k is the size of surrounding *k*-mers, and l is the sequence length. The model has 130,882 parameters with 130,774 trainable parameters; non-trainable parameters are from the batch normalization layers.

The motivation of this model architecture is to concurrently extract the sequential features from the observed read and the consensus sequence in synchronization and to learn context dependencies along the read and the consensus sequence. NanoReviser [106], a signal-based

genomic corrector, uses CNNs to process raw electric signals, and CNNs are also used in applications that encode features into images [48]. While CNNs are good at extracting features from images/spatial data or data with strong locality, LSTMs/Bi-LSTMs are context-aware and excel at learning sequential features. Our input features are sequence-based and LSTMs/Bi-LSTMs are well suited for our application. DeepConsensus [5], a brand new method for consensus generation from PacBio HiFi sequencing subreads (published at the time we just completed this work), uses an encoder-only transformer architecture.

Given 5 inputs: X_{left_kmers} (k-mers before bases), X_{right_kmers} (k-mers after bases), X_{read_feats} (read base-related features), X_{cons_feats} (consensus-related features), and X_{rest_feats} (remaining read-level features), the formulation of our neural network model is summarized below:

$$X_{learnt\_left\_kmers} = TimeDistributed\text{-}LSTM^{(16)}(BatchNorm(Mask(X_{left\_kmers}))) \tag{4.1}$$

$$X_{learnt\_right\_kmers} = TimeDistributed\text{-}LSTM^{(16)}(BatchNorm(Mask(X_{right\_kmers}))) \tag{4.2}$$

$$X_{comb\_read\_feats} = CAT(X_{learnt\_left\_kmers}, X_{learnt\_right\_kmers}, Mask(X_{read\_feats})) \tag{4.3}$$

$$X_{learnt\_read\_feats} = Bi\text{-}LSTM^{(64\times 2)}(BatchNorm(X_{comb\_read\_feats})) \tag{4.4}$$

$$X_{learnt\_cons\_feats} = Bi\text{-}LSTM^{(64\times 2)}(BatchNorm(Mask(X_{cons\_feats}))) \tag{4.5}$$

$$X_{all\_feats} = CAT(X_{learnt\_read\_feats}, X_{learnt\_cons\_feats}, Mask(X_{rest\_feats})) \tag{4.6}$$

$$X_{all\_feats\_fc1} = ReLU(TimeDistributed\text{-}FC^{(128)}(X_{all\_feats})) \tag{4.7}$$

$$X_{all\_feats\_fc2} = ReLU(TimeDistributed\text{-}FC^{(32)}(X_{all\_feats\_fc1})) \tag{4.8}$$

$$Y_{predict} = Softmax(TimeDistributed\text{-}FC^{(10)}(X_{all\_feats\_fc2})) \tag{4.9}$$

where FC represents the fully connected layer, CAT represents concatenate, BatchNorm represents batch normalization, and the superscript $(\cdot)$ in LSTM, Bi-LSTM, and FC indicates the number of their output units. The model is implemented using Keras-TensorFlow (v 2.8.0).

For training the model, we use Adam optimizer [40]. The loss function we use is Sparse Categorical Cross Entropy. Weights in the network are initialized using Xavier uniform initialization [30]. The model training parameters are given in Table 4.1.

Table 4.1: Model training parameters.

Optimizer	Learning rate	Loss	Weights initialization	Batch size	Epochs
Adam optimizer	0.001	Sparse Categorical Cross Entropy	Xavier uniform initializer	256	170

4.2.4 Training data and preprocessing

To learn the error profile information, we use human sequencing reads for training the model, as simulated data does not capture the full scope of the error profile in real data (since ONT's error profile has not been well studied yet). We use NA12878 ONT transcriptomic cDNA reads [111] for training. NA12878 was sequenced by MinION using the 1D ligation kit (SQK-LSK108) and R9.4 chemistry (FLO-MIN106), and was basecalled using Guppy (v 4.2.2).

For preprocessing, we use Pychopper [67] (v1) to trim adapters/primers and poly-A tails, and to orient reads to the forward strand. However, there are still many poly-A tails remaining. Thus, we perform cutadapt [64] (v 3.5) three times to trim the remaining poly-A tails, since

poly-A tails also contain sequencing errors making it difficult to remove all with one trim (after 3 trimmings, there are still remaining poly-A tails). Pychopper identifies and keeps full-length reads and filters out the reads with the average quality score < 7. Cutadapt keeps the reads with the read length ≥ 20. The parameters we use for Pychopper and cutadapt can be found in Appendix Section 4.5.1. We take the first 3 million reads from the preprocessed NA12878 cDNA data for the downstream analysis, and their statistics are given in Table 4.2. We call this set of 3 million reads "human data set 1." More details about the training set are given in Section 4.3.1.

Table 4.2: Statistics of the first 3,000,000 reads of the preprocessed NA12878 cDNA data.

Max length	Min length	Average length	Median length	Mode length	Std dev
8,951	20	677.5	550	500	462.6

4.2.5 Clustering and MSA-POA

To perform MSA-POA on the supporting reads, we cluster the reads such that the reads from the same isoform approximately go to the same cluster. Currently, there are two RNA-seq long-read clustering tools: isONclust [80] and RATTLE's clustering module. isONclust uses a greedy clustering algorithm that uses base quality values. Although isONclust is originally for gene-level clustering, it can be adapted to do isoform clustering using higher mapped/aligned thresholds. RATTLE's clustering module has an isoform clustering mode; however, RATTLE's clustering requires a large amount of memory so it exceeds available memory for sizable data sets (it core dumps with memory allocation errors for clustering 50K reads on our server, which has 1007 Gb of RAM). Thus, we use isONclust (v 0.0.6.1) to cluster the reads.

The largest clusters can be fairly large (e.g. may contain nearly $\sim$10K reads), which are not suitable for MSA-POA. Therefore, to handle large clusters, we designed the following approach to perform clustering and extract the MSA-POA set (by setting the maximum MSA-POA set size; let M be the maximum MSA-POA set size-1):

1. Use isONclust to cluster the reads with the thresholds (see Appendix Section 4.5.1) for isoform clustering.
2. For clusters with $> M$ reads:
   (a) Sort the reads inside the cluster such that longer reads with higher quality appear earlier.
   (b) Use the top M reads in the sorted list to perform MSA-POA, which takes care of the first M reads.
   (c) For each of the remaining reads, group that read with the top M reads to perform MSA-POA, since the top M reads are more reliable (because they have higher quality and are longer).
3. For clusters with $\leq M$ reads, perform MSA-POA on the full cluster.

Thus, the MSA-POA sets have $\leq M + 1$ reads. We use $M = 800$ by default.

We use SPOA (v 4.0.7) to perform graph-based MSA; SPOA is a SIMD (single instruction multiple data)-accelerated POA algorithm developed in a consensus module Racon [103]. We

perform SPOA on full reads in each cluster to generate the consensus and the MSA alignment matrix. We only use clusters with at least 3 reads. In SPOA, we use Smith-Waterman alignment mode, with the gap opening penalty -8 and the gap extension penalty -6. The parameters we use for isONclust and SPOA can be found in Appendix Section 4.5.1.

4.2.6 Feature extraction

From the MSA alignment matrix, we compute the occurrence frequency of the consensus base in the MSA column and the average base quality score of the consensus base across its occurrences in the MSA column. The read quality sequence from the FASTQ file is converted into numerical values based on the Phred+33 scale. We construct the pairwise alignment between the read of interest and the consensus sequence by removing their common gaps. The read quality sequence, the consensus occurrence frequency sequence, and the consensus average quality sequence are also aligned with them. The left and right *k*-mers for each base position on the read are constructed based on the physical bases only. The shorter *k*-mers at the two ends of the read are padded to length *k* (see details in Section 4.2.8). Other features such as GC-content ($GC\text{-}content = count(G+C) \,/\, count(A+T+G+C)$), read length, the number of supporting reads in the MSA-POA set, and the strand of the read on cDNA are also extracted.

4.2.7 Creating true labels

We create the true labels based on the splice-aware alignment of the read to the reference genome. To extract the true read, we use minimap2 [52] (v 2.21-r1071) to perform the splice-aware alignment of the read to the reference genome (GRCh38). We filter out secondary alignments, supplementary alignments, and unaligned reads. We also skip chimeric alignments as they might be caused by false alignments. From the alignment of the read to the reference genome, we construct the true read, the pairwise alignment between the read and the true read, and true labels by parsing the *cs* tag and the CIGAR string in the SAM file. Soft clips are treated as insertions at the ends. When a read is mapped to the reverse strand of the genome, the *cs* tag and CIGAR are reverse complemented from the original sequence; in this case, we reverse complement the constructed true read, the pairwise alignment, and true labels. The parameters we use for minimap2 can be found in Appendix Section 4.5.1.

To combine the true labels with the features, we need to merge the two pairwise alignments—the pairwise alignment of the read vs. the true read and the pairwise alignment of the read vs. the consensus. We perform this merge by using the read as the "center" sequence, and the merge preserves existing gaps in the pairwise alignments. The merging proceeds column by column: if there is no gap in the read sequences in both pairwise alignments, or if both have gaps, put the character paired with the read sequence into the column. If there is a gap in the read sequence in alignment 1, but no gap in the read sequence in alignment 2, place a gap for sequences in alignment 2 at this column; and vice versa. All sequences with alignment 1 (read sequence, consensus sequence, read quality, consensus occurrence frequency, consensus average quality) match together consistently, and all sequences with alignment 2 (read sequence, true read sequence, true labels) match together consistently.

4.2.8 Padding and masking

One challenge here is highly variable sequence lengths (e.g. our read lengths range from 20 to 8,951 bp), while inputs to RNNs must have the same number of timesteps within each batch. Moreover, back-propagation across very long sequences may result in vanishing gradients and an unlearnable model, so in common practice, a limit of 250–500 timesteps is usually used for large LSTM models. Thus, we set a maximum sequence length l, and the reads longer than l are segmented into multiple sequences of length l. We use $l = 600$ by default (as our median read length is 550). Although those longer reads are segmented, their "read length" and "base position" features keep the original values. For those very long reads, the model learns the contextual dependencies within 600 bp, which is likely sufficient. On the other hand, for those sequences with lengths $< l$ (600), we pad the sequence to the maximum sequence length l (600) with value –1.

We mask the input sequences based on the padding value. The mask is propagated to the output, such that the layers in the network ignore those masked timesteps. The mask is eventually applied to the loss function, such that the loss on the padding placeholders would be ignored.

We also normalize the data columns of features, such that all feature values are within 0 and 1.

4.2.9 *De novo* error correction using the model

Through the prediction by the trained model, we error correct the reads *de novo* (without using a reference genome, transcriptome, or annotation). Our method for error correcting the reads is as follows. We correct the clusters one by one using the trained model. For each cluster:

1. Extract the features for all reads in the cluster; perform segmentation and padding.
2. Use the model to predict the class probabilities by using all features of reads in the cluster.
3. Convert the predicted probabilities into class labels; the model prediction is sequential.
4. Extract the mask by applying a standalone Masking layer to the input (as the model prediction no longer carries the mask).
5. For each read in the cluster:
   (a) Combine the predicted label segments for the read into a sequence of unmasked labels (with the padded timesteps removed) by applying the mask, as a read can span multiple segments.
   (b) Correct the read sequence using the predicted labels.
   (c) Re-construct the quality score sequence accordingly (keep original scores for unchanged bases; assign a score 10 to corrected bases).
6. Write the corrected read sequences and quality scores to a FASTQ file.

The output FASTQ files of all corrected clusters are merged together along with the original FASTQ files of the uncorrected clusters (i.e. clusters with < 3 reads).

The code and the trained model for deepCorrRNA are available at:
https://github.com/Kingsford-Group/dlerrorcorrectrna.

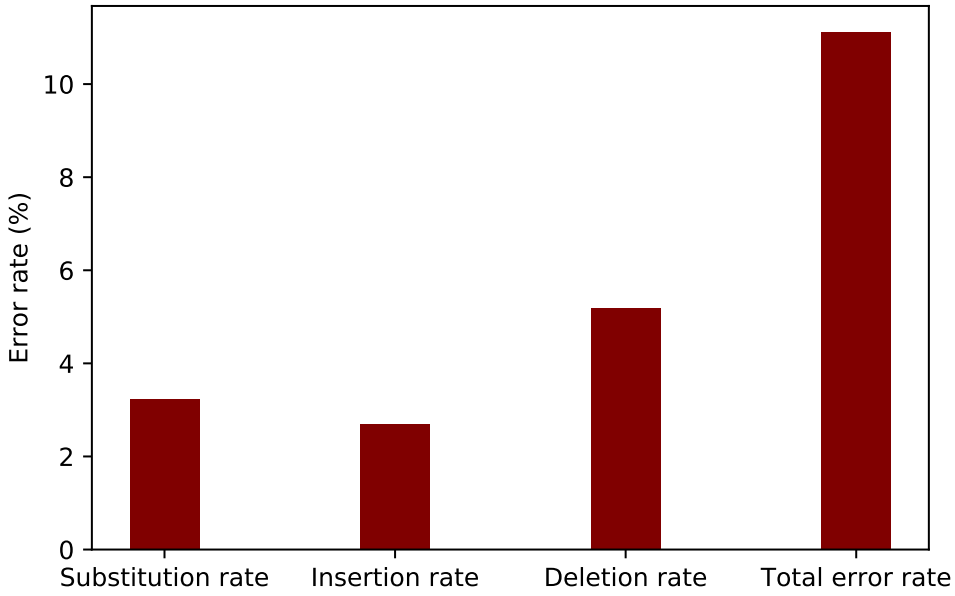


Figure 4.3: Error rates of the first 3 million reads of the preprocessed NA12878 cDNA data.

4.3 Results

4.3.1 Neural network model training and evaluation results

To create the training and test sets, we randomly sample N_c clusters from the clusters of the first 3 million reads of the NA12878 cDNA data [111]. From each sampled cluster, we randomly sample N_r reads if the cluster has $> N_r$ reads; otherwise, take all reads in the cluster. $N_c \gg N_r$. From these sampled reads, we split the training and test sets in a 4:1 ratio with random shuffling.

Fig. 4.3 shows the error rates of the first 3 million reads of the preprocessed NA12878 cDNA data, computed using NanoSim. Since the majority of the bases in the reads have no errors, the 10 classes are imbalanced—while the "No error" class is the majority class, the other 9 classes are minority classes (refer to Table 4.15 for the class label distribution). A common practice to handle imbalanced classes is using class weights, such that the model training could pay more attention to the minority classes. Scikit-learn is often used to compute class weights, and in scikit-learn's formula, the class weights are inversely proportional to their frequencies. However, we found that using the scikit-learn computed class weights that are exactly inversely proportional to the frequencies and with large differences of values, the model training became unstable, and F1-scores on the minority classes became much lower than those without using class weights. Although adjusting the class weights by significantly reducing their value differences proportionally could yield a stabler training and improve the F1-scores on the minority classes, we found that the class weights boost the recall of the minority classes at the price of decreasing their precision consistently regardless of the value tuning. Although smaller class weight values may achieve a better balance between precision and recall and increase the F1-scores for the minority classes, the F1-scores' improvements are small and the precision is still reduced. However, the precision is very important for our application, since we do not want to introduce many new

errors in the error correction process. In our application, sacrificing the precision to boost the recall would not be beneficial, as our top priority is to get a lower total error rate.

Thus, we use a different approach instead of class weights to improve the model performance on the minority classes (to increase the precision and recall together). We progressively enlarge the training set by approximately doubling N_c and N_r with each increase. Tables 4.16–4.18 show the classification reports of the models trained with the data set generated from $N_c = 2000$ randomly sampled clusters and $N_r = 10$ randomly sampled reads per cluster, the data set generated from $N_c = 5000$ randomly sampled clusters and $N_r = 20$ randomly sampled reads per cluster, and the data set generated from $N_c = 10000$ randomly sampled clusters and $N_r = 40$ randomly sampled reads per cluster, respectively. We can see that with each increase in the training set, the precision and recall mostly increase simultaneously for the minority classes, which significantly improves their F1-scores. Meanwhile, the model performance on the majority class ("No error") is maintained or slightly improved, which is desirable as we want to maintain the high accuracy for the correct bases. Therefore, we further increase the training set by doubling N_c and N_r again, and train our model with the data set generated from $N_c = 20000$ randomly sampled clusters and $N_r = 80$ randomly sampled reads per cluster; the training and test sets used for our model are shown in Table 4.3.

This approach works because with each increase in the training set, we are increasing the occurrences of the minority classes to help the model learn from them. In our application, we are not seeking to even out the performance between the majority and minority classes; we want to maintain the high accuracy of the majority class while improving the performance of the minority classes. By progressively enlarging the training set, we are observing the improvement on the minority classes and also observing the saturation of the improvement after the training set reaches a certain size. This helps estimate the best possible improvement that we may reach.

Table 4.3: Model trained with the data set created from $N_c = 20000$ randomly sampled clusters & $N_r = 80$ randomly sampled reads per cluster.

Training set	Test set	Epochs	Test accuracy
253444 reads (516129 samples after segmentation)	63361 reads (129468 samples after segmentation)	170	0.9634

The training history as illustrated in Fig. 4.4 indicates that the model is not overfit, and 170 epochs are sufficient for this training set. Here, we use the validation split of 0.1 during the training. The training accuracy is slightly lower than the validation accuracy, because we use dropouts in the model. Dropouts are turned off at testing time, so the model at testing time is more robust and can lead to a higher accuracy.

Our model achieves the test accuracy of 0.9634. The classification report of our model evaluated using the test set is shown in Table 4.4. Although the model still favors the majority class ("No error") to give it a high F1-score (which is desirable in our application), its performance (F1-scores) on the minority classes is substantially improved. Notably, the precision of the minority classes are all above 0.8, which is important for our application. The model performs better on the deletion classes than the substitution classes; one possible reason is that the deletion classes are $\sim$1.6X more frequent than the substitution classes. The insertion class has about equal performance to the substitution classes, although it is larger; a possible reason is that the

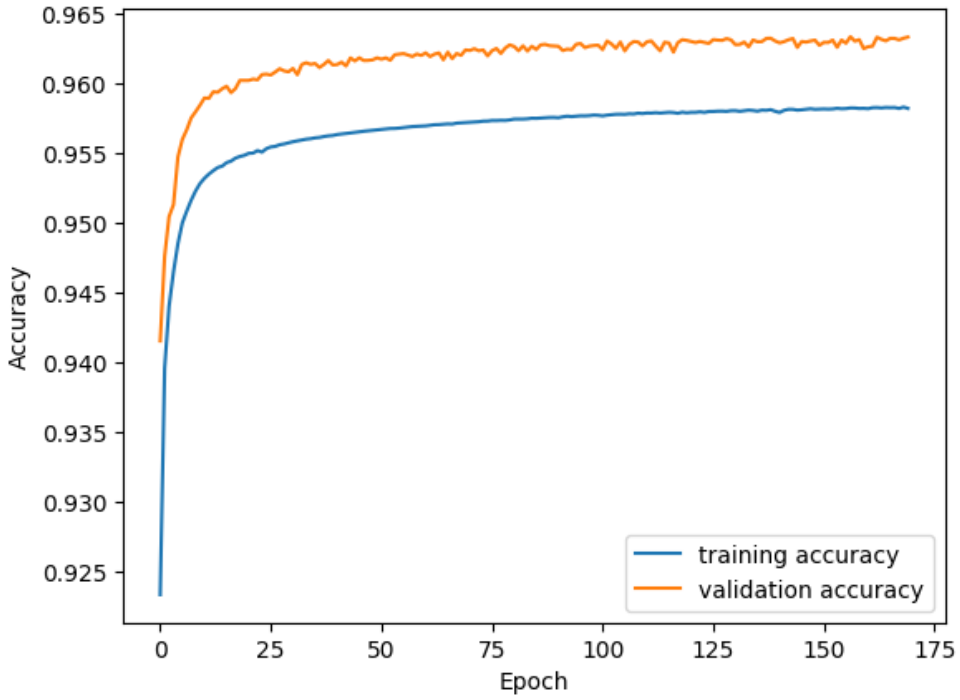


Figure 4.4: Training history of the model (trained with the data set from $N_c = 20000$ random clusters & $N_r = 80$ random reads per cluster).

situation for insertions is complicated (due to leftover poly-A tails, etc.), so it would be harder for the model to learn. In the classification report, the "support" is the number of bases in a class in the true labels of the test set; the "macro avg" is the unweighted mean over all classes; the "weighted avg" is the support-weighted mean over all classes.

The confusion matrix of the model evaluated using the test set is shown in Table 4.5. To handle the masked timesteps and 2D labels in computing the confusion matrix, we extract the mask by applying a standalone Masking layer to the input and convert 2D labels into unmasked labels using the mask. In the confusion matrix, rows are true labels, columns are predicted labels, and the numbers are the number of bases belonging to a class. The confusion matrix illustrates that deletions are never misclassified as substitutions or insertions. And substitutions are never misclassified as deletions. Insertions are almost never misclassified as deletions, but are misclassified as substitutions for a set of bases. The deleted base letters have a set of misclassifications across each other. Substitutions are misclassified as insertions for a set of bases, but the substituted base letters have fewer misclassifications across each other. Some correct bases are misclassified as error bases for all 9 types of errors, with insertions the most, but their fractions are small ($< 0.75\%$ in total). On the other hand, for all 9 types of errors, some error bases are not detected and are mis-treated as no error, with insertions the most and deletions the least.

We also further enlarged the training set by doubling N_c and N_r once again to retrain the model. However, we found that the retrained model did not show beneficial improvement over this model on the minority classes. After the training set reaches a certain size, the improvement on the minority classes tends to become saturated. Thus, this model is used for error correcting

Table 4.4: Classification report of the model (trained with the data set created from $N_c = 20000$ randomly sampled clusters & $N_r = 80$ randomly sampled reads per cluster), evaluated using the test set.

	precision	recall	F1-score	support
Insertion	0.80	0.64	0.72	2342599
Substituted_A	0.82	0.63	0.71	347636
Substituted_C	0.82	0.68	0.74	456338
Substituted_G	0.83	0.64	0.72	412506
Substituted_T	0.83	0.63	0.72	365707
Deleted_A	0.83	0.83	0.83	612480
Deleted_C	0.85	0.82	0.83	650999
Deleted_G	0.82	0.84	0.83	677938
Deleted_T	0.87	0.80	0.83	605985
No_error	0.98	0.99	0.98	52845896
accuracy			0.96	59318084
macro avg	0.85	0.75	0.79	59318084
weighted avg	0.96	0.96	0.96	59318084

Table 4.5: Confusion matrix of the model (trained with the data set created from $N_c = 20000$ randomly sampled clusters & $N_r = 80$ randomly sampled reads per cluster), evaluated using the test set.

	Insertion	Sub_A	Sub_C	Sub_G	Sub_T	Del_A	Del_C	Del_G	Del_T	No_error
Insertion	**1510393**	19968	30238	22201	19391	0	1	5	4	740398
Sub_A	28754	**217645**	5188	1570	3297	0	0	0	0	91182
Sub_C	34429	3612	**309202**	4783	1596	0	0	0	0	102716
Sub_G	35717	2017	6200	**264304**	3614	0	0	0	0	100654
Sub_T	30762	3364	1484	4885	**230581**	0	0	0	0	94631
Del_A	0	0	0	0	0	**507824**	24555	32077	17717	30307
Del_C	0	0	0	0	0	26859	**530834**	41064	19223	33019
Del_G	0	0	0	0	0	28764	26762	**570023**	16900	35489
Del_T	0	0	0	0	0	28450	27148	35996	**481857**	32534
No_error	241455	17242	23260	19293	18662	21069	16970	18826	16786	**52452333**

the reads in deepCorrRNA.

4.3.2 ML-based deepCorrRNA achieves similar total error rate reductions to state-of-the-art isONcorrect on human data

To evaluate the error correction results, we use the NanoSim (v 3.0.0) read analysis module to compute the base-level error rates. The NanoSim read analysis is based on aligning the reads to the reference genome/transcriptome using minimap2. For human data, the Ensembl GRCh38 references are used in evaluation. For comparison, we also run isONcorrect (v 0.0.8) with default parameters. The parameters we use for NanoSim and isONcorrect can be found in Appendix Section 4.5.1. We evaluate deepCorrRNA on a human ONT transcriptomic cDNA data set (Table 4.6. Human data set 2 has no overlaps with the human data set 1 that is used in training).

ML-based deepCorrRNA achieves similar reductions in the total error rates to state-of-the-art isONcorrect on human data. Table 4.7 shows the error correction results for the human data set 2 that contains middle 500,000 reads of preprocessed NA12878 cDNA data [111]. The post-

Table 4.6: Human ONT cDNA data set used in evaluation, sequenced by MinION.

Data set	Description	Chemistry/Kit	Basecaller	Max length	Median length	% Reads uncorrectable
Human data set 2	Middle 500,000 reads of preprocessed NA12878 cDNA	R9.4 SQK-LSK108	Guppy (v 4.2.2)	7,307	580	12.03%: singletons; 3.13%: in 2-reads clusters. $\Rightarrow$ 15.16% not corrected.

correction error rates here are for all reads in this data set including 15.16% of reads that are not corrected (which are singletons or in 2-reads clusters). deepCorrRNA performs better than isONcorrect on correcting substitutions and insertions, while isONcorrect performs better than deepCorrRNA on correcting deletions. In human data set 2, deepCorrRNA has a slightly lower post-correction total error rate than isONcorrect, yet their values are similar.

An interesting phenomenon is that deepCorrRNA's performance on reducing deletions seems lower than on reducing substitutions and insertions, while the model evaluation shows that the F1-scores on deletions are higher than the F1-scores on substitutions and insertions. The reason for this phenomenon is that, given the insertion class's precision of 0.80 (which is lower than other classes' precision), the false positive insertions cause the error correction process to remove those bases that are correct bases or substituted bases. This increases the post-correction deletion rate. In the confusion matrix (Table 4.5), the numbers in the first column except for the first row are false positive insertions. In the last row ("No_error"), the number in the first column is $>$10X larger than the numbers in other error-type columns. The four numbers in the first column corresponding to the substitution rows are also relatively not low. These false positive insertions will become deletions after error correction, contributing to the resulting deletion rate.

Table 4.7: Error correction results for human data set 2: middle 500,000 reads of preprocessed NA12878 cDNA data.

	Original reads (filtered reads with quality $<$7)	**deepCorrRNA** (error correction output reads)	**isONcorrect** (error correction output reads)
Substitution rate (%)	3.24026	1.37935	1.67937
Insertion rate (%)	2.85817	1.34842	1.61290
Deletion rate (%)	4.96583	3.51204	3.04169
Total error rate (%)	11.06425	6.23980	6.33395

We also examine how error correction by deepCorrRNA may affect variants in the reads. Since Genome in a Bottle (GIAB) has benchmark (high-confidence) variant calls [104] for NA12878, we perform a post hoc analysis on variants for the human data set 2 (i.e. middle 500,000 reads of preprocessed NA12878 cDNA data). In this analysis, we use NA12878 (HG001) GRCh38 benchmark (v4.2.1) VCF (SNPs and indels) from the GIAB. We align the original reads in the human data set 2 to the reference genome (GRCh38) using minimap2 (v2.21-r1071) splice-aware alignment. From the alignments, we use "*bcftools mpileup*" and "*bcftools call*" (bcftools v1.15.1) to do variant calls for the original reads. We use the same process (with the same parameters) to do variant calls for the error-corrected reads from deepCorrRNA. Table 4.8 shows that while 5138 GIAB variants are found in the original reads, 5382 GIAB variants are found in deepCorrRNA error correction output reads. Thus, 244 more (4.75% more) GIAB variants are found in deepCorrRNA error correction output reads than in the original reads. This is because

deepCorrRNA is reference-free and uses long reads themselves to do self-correction, so that most variants are retained. While error correction could lose some variants when the minor allele with a very low number of reads covering it gets "corrected," error correction by deepCorrRNA helps more variants to be recovered when the errors in the reads are removed. This indicates that, overall, error correction by deepCorrRNA would benefit variants analysis.

Table 4.8: Variants analysis results for human data set 2: middle 500,000 reads of preprocessed NA12878 cDNA data.

	In original reads	**In deepCorrRNA error correction output reads**
Number of GIAB variants found	5138	5382

4.3.3 deepCorrRNA achieves similar total error rate reductions to isONcorrect on mouse and Drosophila data, demonstrating *de novo* capability

Since our model is trained using human data, we evaluate deepCorrRNA on different organisms without retraining to examine its *de novo* error correction capability. For evaluation, the different organism needs to have a good reference genome, but in the error correction process, we do not use any of its references as if the references are unavailable. We use mouse and Drosophila data for this examination. For mouse data, the Ensembl GRCm38 references are used in evaluation. We evaluate deepCorrRNA on three mouse ONT transcriptomic cDNA data sets (Table 4.9) with varying error rate profiles.

Table 4.9: Mouse ONT cDNA data sets used in evaluation, sequenced by MinION.

Data set	**Description**	**Chemistry/Kit**	**Basecaller**	**Max length**	**Median length**	**% Reads uncorrectable**
Mouse data set 1	First 500,000 reads of preprocessed **ERR3363660** cDNA	R9.4.1 PCS108	Guppy (v 2.3.5)	8,111	700	14.08%: singletons; 4.51%: in 2-reads clusters. ⇒ 18.59% not corrected.
Mouse data set 2	First 500,000 reads of preprocessed **SRR17960984** cDNA	R9.4.1 SQK-LSK109	Guppy (v 4.0.11)	8,017	750	6.1%: singletons; 3.13%: in 2-reads clusters. ⇒ 9.23% not corrected.
Mouse data set 3	First 500,000 reads of preprocessed **SRR17960971** cDNA	R9.4.1 SQK-LSK109	Guppy (v 4.0.11)	8,418	780	8.99%: singletons; 5.26%: in 2-reads clusters. ⇒ 14.25% not corrected.

deepCorrRNA achieves similar reductions in the total error rates to isONcorrect on mouse data, demonstrating its *de novo* error correction capability. Table 4.10 shows the error correction results for the mouse data set 1 that contains first 500,000 reads of preprocessed ERR3363660 [87] cDNA data. The post-correction error rates here are for all reads in the data set including 18.59% of reads that are not corrected (which are singletons or in 2-reads clusters). This mouse data set has lower original error rates than the human data set. As in the human data set, deepCorrRNA performs better than isONcorrect on correcting substitutions and insertions, while isONcorrect

performs better than deepCorrRNA on correcting deletions. deepCorrRNA has a slightly lower post-correction total error rate than isONcorrect, and their values are similar.

Table 4.10: Error correction results for mouse data set 1: first 500,000 reads of preprocessed ERR3363660 cDNA data.

	Original reads (filtered reads with quality <7)	**deepCorrRNA** (error correction output reads)	**isONcorrect** (error correction output reads)
Substitution rate (%)	2.91618	1.08498	1.32605
Insertion rate (%)	2.38456	1.05038	1.20536
Deletion rate (%)	4.26372	2.45860	2.14280
Total error rate (%)	9.56445	4.59396	4.67421

Table 4.11 shows the error correction results for the mouse data set 2 that contains first 500,000 reads of preprocessed SRR17960984 [20] cDNA data. The post-correction error rates here are for all reads in the data set including 9.23% of reads that are not corrected (which are singletons or in 2-reads clusters). This mouse data set has a different original error-rate distribution from the previous data sets—while in the prior data sets, the deletion rate is the highest and the insertion rate is the lowest, in this data set, the insertion rate is the highest and the substitution rate is the lowest. As in the previous data sets, deepCorrRNA performs better than isONcorrect on correcting substitutions and insertions, while isONcorrect performs better than deepCorrRNA on correcting deletions. deepCorrRNA has a slightly higher post-correction total error rate than isONcorrect, but their values are similar. deepCorrRNA has 54.36% reduction on the total error rate after error correction.

Table 4.11: Error correction results for mouse data set 2: first 500,000 reads of preprocessed SRR17960984 cDNA data.

	Original reads (filtered reads with quality <7)	**deepCorrRNA** (error correction output reads)	**isONcorrect** (error correction output reads)
Substitution rate (%)	3.21844	1.07116	1.22315
Insertion rate (%)	3.87380	1.48348	1.73728
Deletion rate (%)	3.72607	2.38326	1.64315
Total error rate (%)	10.81831	4.93791	4.60359

Table 4.12 shows the error correction results for the mouse data set 3 that contains first 500,000 reads of preprocessed SRR17960971 [20] cDNA data. The post-correction error rates here are for all reads in the data set including 14.25% of reads that are not corrected (which are singletons or in 2-reads clusters). This mouse data set has a similar original error-rate distribution to mouse data set 2, but its total error rate is higher than all previous data sets. As in all other data sets, deepCorrRNA performs better than isONcorrect on correcting substitutions and insertions, while isONcorrect performs better than deepCorrRNA on correcting deletions. deepCorrRNA has a slightly lower post-correction total error rate than isONcorrect, and their values are similar.

Mouse transcriptome and human transcriptome share some similarities. Therefore, to further examine the *de novo* error correction capability, we also use Drosophila that is far different from human. We evaluate deepCorrRNA on a Drosophila ONT transcriptomic cDNA data set without retraining (Table 4.13). The Ensembl BDGP6.32 references are used in evaluation.

Table 4.12: Error correction results for mouse data set 3: first 500,000 reads of preprocessed SRR17960971 cDNA data.

	Original reads (filtered reads with quality <7)	**deepCorrRNA** (error correction output reads)	**isONcorrect** (error correction output reads)
Substitution rate (%)	3.61341	1.36674	1.72010
Insertion rate (%)	4.33099	1.85731	2.48390
Deletion rate (%)	4.15452	3.04277	2.19067
Total error rate (%)	12.09893	6.26682	6.39467

Table 4.13: Drosophila ONT cDNA data set used in evaluation, sequenced by MinION.

Data set	**Description**	**Kit**	**Basecaller**	**Max length**	**Median length**	**% Reads uncorrectable**
Drosophila data set	First 500,000 reads of preprocessed **SRR15541957** cDNA	SQK-PCS109	unknown	8,828	690	3.58%: singletons; 1.35%: in 2-reads clusters. ⇒ 4.93% not corrected.

deepCorrRNA achieves similar reductions in the total error rates to isONcorrect on Drosophila data, further demonstrating its *de novo* error correction capability. Table 4.14 shows the error correction results for the Drosophila data set that contains first 500,000 reads of preprocessed SRR15541957 [11] cDNA data. The post-correction error rates here are for all reads in the data set including 4.93% of reads that are not corrected (which are singletons or in 2-reads clusters); this percentage of uncorrected reads is lower than those of human and mouse data sets. As in the human and mouse data sets, deepCorrRNA performs better than isONcorrect on correcting substitutions and insertions, while isONcorrect performs better than deepCorrRNA on correcting deletions. deepCorrRNA has a slightly higher post-correction total error rate than isONcorrect, yet their values are similar. deepCorrRNA has 60.98% reduction on the total error rate after error correction.

Table 4.14: Error correction results for Drosophila data set: first 500,000 reads of preprocessed SRR15541957 cDNA data.

	Original reads (filtered reads with quality <7)	**deepCorrRNA** (error correction output reads)	**isONcorrect** (error correction output reads)
Substitution rate (%)	3.37010	1.00464	1.21871
Insertion rate (%)	2.22500	0.76037	0.81495
Deletion rate (%)	3.94168	1.95646	1.43723
Total error rate (%)	9.53678	3.72147	3.47089

deepCorrRNA's correction performance on the mouse data with various error rate profiles and on the data of Drosophila that is far different from human demonstrates its ability to error correct the reads for non-model organisms when a high-quality reference is unavailable.

4.4 Discussion

Our results show that the ML-based deepCorrRNA has comparable error-rate reductions to state-of-the-art ONT-specific isONcorrect and demonstrates effective *de novo* error correction capa-

bility, while it takes into account the error profile associated information systematically, which makes this correction method generalizable. Learning error profile related features systematically through the deep neural network in deepCorrRNA avoids manual parameters such as setting thresholds for consensus frequency, quality, etc. and other heuristics in prior graph-only methods, and allows the model to learn the relationships between harmful *k*-mers, etc. and error occurrences/types to help error detection.

deepCorrRNA presents a generalizable method that may be applied to different technologies. While the current model is trained with ONT reads, our model can be used as a pre-trained model for other technologies. For a different technology (e.g. PacBio), we can train a model with the training reads from that technology by using our current model as a pre-trained model—using one of the three transfer learning strategies: 1) re-learning all weights using the architecture of the pre-trained model (with or without its nanopore-trained weights as initialization), 2) learning the weights of some layers (higher layers) while leaving other layers (initial layers) frozen, or 3) using the pre-trained model as a fixed feature extraction mechanism (only training the fully connected layers). As a result, deepCorrRNA can be a versatile error correction tool that fits to different technologies—the implementation supports applying such various models using *–technology* as a parameter and loading the trained model of the user-specified technology to error correct the reads. Assessing the approaches' generalizability on other sequencing technologies is a direction for future work.

The precision of the insertion class plays a critical role in reducing the deletion rate, since false positive insertions would become deletions post-correction. An important factor that could affect the precision of the insertion class is the clustering accuracy. When a read that should not belong to the same cluster as the other reads is clustered together with the other reads, the read could have a region of insertions when aligned to the consensus of the cluster, although this region is not actually insertions with respect to the true read. We found these cases in the clusters. This could confuse the model and make it harder to learn and predict. To increase the clustering accuracy, we experimented with a stricter clustering with isONclust by increasing the mapped/aligned thresholds and reducing the window-size for minimizers. However, the stricter clustering causes 27.4% of reads to be singletons in the human data set 1, compared to just 8.3% of reads being singletons with the clustering parameters used for the experiments here. The clustering accuracy gain is not worth leaving 27.4% of reads not clustered and thus not corrected. Since the model performance can be somewhat limited by the clustering accuracy, it would be beneficial to develop a new clustering tool for RNA-seq long reads that performs accurate isoform-based clustering (such that only the reads from the same isoform are clustered together) and is able to handle large data sets (although RATTLE has an isoform clustering mode, it has memory problems for sizable data sets). This is also a direction for future work.

Class weights boost the recall of the minority classes but sacrifice their precision, which is not beneficial to our application. To further improve the model performance on the minority classes, other approaches may also be explored, such as optimized loss functions for imbalanced data [53]. Furthermore, the feature set of the model could be further extended when additional sequence-level error-related factors can be identified. Moreover, analyzing the feature importance in the model for all features may offer some insights on how much the error profile related features contribute to the model performance gain. These are more directions for future work.

The current model is trained with ONT cDNA reads. deepCorrRNA can be extended to ONT

direct RNA sequencing technology by retraining the model using direct RNA sequencing reads with a slightly modified feature set (removing the strand feature or labeling all reads as forward strand; encoding base letters 'ACGU'). The preprocessing should be modified as Pychopper is only applicable to cDNA reads. Additionally, another direction for future work could be training the model with the combination of the reads from multiple organisms and comparing the model performance with the current model.

4.5 Appendix

4.5.1 Parameters used in software tools or packages

For Pychopper:
cdna_classifier.py ⟨input.fastq⟩ ⟨output.fastq⟩ -t 85
For cutadapt:
a) for the first and the second trims:
cutadapt -a "A{60}" -O 20 -m 20 -e 0.05 -j 80 -o ⟨output.fastq⟩ ⟨input.fastq⟩
b) for the third trim:
cutadapt -a "A{60}" -O 16 -m 20 -e 0 -j 80 -o ⟨output.fastq⟩ ⟨input.fastq⟩
For isONclust:
isONclust –t 32 –ont –mapped_threshold 0.9 –aligned_threshold 0.7 –fastq ⟨input.fastq⟩ –outfolder ⟨output_folder⟩
isONclust write_fastq –N 1 –clusters final_clusters.tsv –fastq ⟨input.fastq⟩ –outfolder ⟨output_folder⟩
For SPOA:
spoa -l 0 -r 2 ⟨input.fastq⟩ > ⟨output.fasta⟩
For minimap2:
minimap2 –cs -ax splice -t 85 GRCh38.fa ⟨input.fastq⟩ -o ⟨output.sam⟩
For NanoSim:
a) for human:
read_analysis.py transcriptome -i ⟨input.fastq⟩ -rg GRCh38.fa -rt Homo_sapiens.GRCh38.cdna.all.fa -annot Homo_sapiens.GRCh38.90.gtf –no_model_fit -o ⟨output_dir/output_prefix⟩ -t 39
b) for mouse:
read_analysis.py transcriptome -i ⟨input.fastq⟩ -rg GRCm38.fa -rt Mus_musculus.GRCm38.cdna.all.fa -annot Mus_musculus.GRCm38.92.gtf –no_model_fit -o ⟨output_dir/output_prefix⟩ -t 39
c) for Drosophila:
read_analysis.py transcriptome -i ⟨input.fastq⟩ -rg BDGP6.32.fa -rt Drosophila_melanogaster.BDGP6.32.cdna.all.fa -annot Drosophila_melanogaster.BDGP6.32.107.gtf –no_model_fit -o ⟨output_dir/output_prefix⟩ -t 39
For isONcorrect:
run_isoncorrect –t 20 –fastq_folder ⟨fastq_folder⟩ –outfolder ⟨output_folder⟩

4.5.2 Additional tables

Table 4.15: The distribution of class true labels for the 10 classes.

Class Labels	Label Percent (%)
Insertion	4.006
Substituted_A	0.605
Substituted_C	0.785
Substituted_G	0.705
Substituted_T	0.639
Deleted_A	1.048
Deleted_C	1.108
Deleted_G	1.163
Deleted_T	1.055
No_error	88.885

Table 4.16: Classification report of the model trained with the data set generated from $N_c = 2000$ randomly sampled clusters and $N_r = 10$ randomly sampled reads per cluster, evaluated using the test samples.

	precision	recall	F1-score	support
Insertion	0.74	0.52	0.61	92555
Substituted_A	0.74	0.60	0.66	14685
Substituted_C	0.78	0.61	0.69	18815
Substituted_G	0.75	0.60	0.67	17085
Substituted_T	0.77	0.59	0.67	15263
Deleted_A	0.76	0.75	0.76	25287
Deleted_C	0.76	0.76	0.76	27018
Deleted_G	0.75	0.77	0.76	28111
Deleted_T	0.77	0.75	0.76	25410
No_error	0.97	0.99	0.98	2179135
accuracy			0.95	2443364
macro avg	0.78	0.69	0.73	2443364
weighted avg	0.95	0.95	0.95	2443364

The data set in Table 4.16 has 10184 training reads and 2546 test reads (21151 training samples and 5305 test samples after segmentation), and the model is trained for 150 epochs. The "support" is the number of bases.

The data set in Table 4.17 has 34916 training reads and 8730 test reads (73182 training samples and 18171 test samples after segmentation), and the model is trained for 160 epochs. The "support" is the number of bases.

The data set in Table 4.18 has 95657 training reads and 23915 test reads (198325 training samples and 49976 test samples after segmentation), and the model is trained for 169 epochs. The "support" is the number of bases.

Table 4.17: Classification report of the model trained with the data set generated from $N_c = 5000$ randomly sampled clusters and $N_r = 20$ randomly sampled reads per cluster, evaluated using the test samples.

	precision	recall	F1-score	support
Insertion	0.76	0.63	0.69	330233
Substituted_A	0.82	0.60	0.69	49821
Substituted_C	0.82	0.64	0.72	64132
Substituted_G	0.81	0.62	0.70	57523
Substituted_T	0.82	0.60	0.70	52125
Deleted_A	0.82	0.78	0.80	86826
Deleted_C	0.81	0.79	0.80	91237
Deleted_G	0.79	0.82	0.80	95415
Deleted_T	0.83	0.77	0.80	87444
No_error	0.97	0.99	0.98	7446245
accuracy			0.96	8361001
macro avg	0.83	0.72	0.77	8361001
weighted avg	0.96	0.96	0.96	8361001

Table 4.18: Classification report of the model trained with the data set generated from $N_c = 10000$ randomly sampled clusters and $N_r = 40$ randomly sampled reads per cluster, evaluated using the test samples.

	precision	recall	F1-score	support
Insertion	0.79	0.62	0.70	883373
Substituted_A	0.80	0.64	0.71	135457
Substituted_C	0.82	0.65	0.73	176332
Substituted_G	0.81	0.65	0.72	158108
Substituted_T	0.82	0.63	0.71	142858
Deleted_A	0.85	0.79	0.82	238239
Deleted_C	0.83	0.81	0.82	251470
Deleted_G	0.78	0.85	0.81	262717
Deleted_T	0.84	0.79	0.82	236303
No_error	0.98	0.99	0.98	20563989
accuracy			0.96	23048846
macro avg	0.83	0.74	0.78	23048846
weighted avg	0.96	0.96	0.96	23048846

Chapter 5

Conclusion

The work presented here has improved our ability to study and analyze complex transcriptomes through the development of new algorithms and computational methods as well as data analysis.

In Chapter 2, we developed the first long-read transcript assembler Scallop-LR, and thus opened up the ability to assemble single-molecule RNA-seq long reads to identify many transcripts and novel isoforms that are missed by Iso-Seq Analysis. Scallop-LR's algorithms are designed to deal with the challenges and characteristics in long reads. By representing long reads as long phasing paths and preserving phasing paths in assembly, Scallop-LR handles the long read lengths and makes full use of the information in long reads. Taking advantage of long-read-specific features such as the read boundary information, Scallop-LR constructs more accurate splice graphs to improve the transcript assembly. Using a post-assembly clustering algorithm, Scallop-LR reduces false negatives in assembly. Scallop-LR also deals with the higher error rates in long reads. Analyzing 26 PacBio data sets with Scallop-LR, Iso-Seq Analysis, and StringTie, we quantified the amount by which transcript assembly improved the Iso-Seq results, revealing the advantage of long-read transcript assembly. Using combined multiple evaluation methods, we show that Scallop-LR not only correctly assembles more known transcripts but also finds more potential novel isoforms than Iso-Seq Analysis. The sensitivity of the Iso-Seq method is limited by the factor that not all CCS reads represent full transcripts [78]. We demonstrate that Scallop-LR can improve this situation by identifying more true transcripts and potential novel isoforms through transcript assembly. With long-read-specific optimizations, Scallop-LR assembles more known transcripts and potential novel isoforms than StringTie for the human transcriptome. Our results indicate that long-read transcript assembly by Scallop-LR can reveal a more complete human transcriptome and benefit transcriptome studies.

In Chapter 3, we developed the first representative set selection approach that can scale to collections of the size of the full set of public human RNA-seq samples in the SRA, thus enabling comprehensive evaluation and parameter optimization of RNA-seq analysis tools by using representative RNA-seq samples. To handle the memory and runtime challenges in the k-mer counting approach that selects a representative set based on k-mer similarities between RNA-seq samples, we developed a novel method: hierarchical representative set selection. Our hierarchical representative set selection algorithm breaks representative set selection into sub-selections and hierarchically selects representative samples through multiple levels. We also developed a novel seeded-chunking method and designed a mean^2-weighting scheme. We demonstrate that our

novel hierarchical representative set selection method can greatly reduce the runtime and memory usage of the direct representative set selection with the full similarity matrix computation, while still achieving summarization quality close to that of the direct representative subset selection. We show that hierarchical representative set selection substantially outperforms random sampling on the entire SRA set of human RNA-seq samples, thus providing a practical solution to representative set selection on large databases like the SRA and facilitating comprehensive evaluation of bioinformatics tools for transcriptome analyses.

In Chapter 4, we developed the first *de novo* RNA-seq long-read error correction method that incorporates the error profile related information systematically using deep learning. To error correct Nanopore RNA-seq long reads for non-model organisms without a good reference, we developed a novel, error-profile-aware error correction method deepCorrRNA to self-correct RNA-seq long reads *de novo*. deepCorrRNA combines a graph-based MSA-POA and a Bi-LSTMs/LSTMs-based deep neural network that incorporates the error profile related information systematically. Evaluating deepCorrRNA on five ONT RNA-seq data sets of three organisms, we show that ML-based deepCorrRNA achieves similar reductions in the total error rates to state-of-the-art ONT-specific RNA-seq error corrector isONcorrect. While its model is trained with the human data, deepCorrRNA's correction performance on different organisms (mouse and Drosophila) demonstrates its effective *de novo* error correction capability and transferability. With comparable error reductions to the state of the art and employing generalized (technology/organism independent) error profile related features comprehensively, deepCorrRNA presents a generalizable method that may be applied to different technologies. High-quality, generalizable, *de novo* error correction for RNA-seq is possible through machine learning, and we show that through our novel method deepCorrRNA, which takes advantage of both graph-based MSA-POA and deep neural network's strengths while integrating the error profile associated information. deepCorrRNA's *de novo* error correction capability for RNA-seq long reads can benefit the transcriptome study of various non-model organisms.

There are also limitations in this work, which open up the opportunity for future work. Currently, Scallop-LR is designed for PacBio long reads. For ONT RNA-seq long reads, it was found in experiments that for both direct RNA and cDNA with full-length enrichment, for transcript lengths < 2.5kb, the reads are mostly full-length; for transcript lengths > 2.5kb, the reads are less likely to be full-length [93]. Hence, before the direct RNA technology is improved, transcript assembly could still be useful for ONT long reads. Thus, a direction for future work is to adapt Scallop-LR to ONT long reads by rebuilding the analysis pipeline and changing the transcript boundary identification functionality.

Another direction for future work is developing a hybrid transcript assembler that combines short reads and long reads. There have been hybrid genome assemblers using both short and long reads, such as MaSuRCA [118] and hybridSPAdes [2]. Although there are two *de novo* transcript assemblers using hybrid sequencing, IDP-denovo [28] and Trinity [32], Trinity and IDP-denovo do not assemble long reads; they assemble short reads and use long reads to improve the assembly of short reads. Thus, a direction for future work is to develop a reference-based hybrid transcript assembler that can assemble both short reads and long reads simultaneously, thus combining the advantages of short reads (low error rates, high throughput) and long reads (long read lengths).

We use the partial Hausdorff distance to evaluate our hierarchical representative set selection. To perform further evaluation on the representative sets, a useful evaluation method could be

using the representative set as the training set to train classifiers to compare the classification accuracy. The classifiers could take the gene expression vectors or transcript abundance vectors as input features, and we could compare the classification accuracy of the models trained by using different representative sets. This is also a direction for future work.

In our k-mer counting-based approach for representative set selection, we download 10,000 reads from each RNA-seq sample to represent that sample. We chose this number since prior researchers found that 10,000 reads are sufficient to tell what genome this sample belongs to, and we do not have extra disk space to hold more reads from all RNA-seq samples in the SRA. We envision that downloading more reads (such as 100,000 reads) would represent each sample better, and thus the selected representative samples based on the subsets of reads would better represent the SRA full set. Thus, a direction for future work is to assess the effect of downloading different numbers of reads from an RNA-seq sample (such as 10K reads vs 100K reads) on the selected representative set, when extra disk space becomes available.

Because deepCorrRNA integrates generalized (technology/organism independent) error profile related features systematically, this method is generalizable and may be applied to different technologies. While the current model of deepCorrRNA is trained with ONT reads, applying this method to different technologies can be done through transfer learning by using the current model as a pre-trained model. Thus, a future work is to apply this method to a different technology (such as PacBio) to experimentally show the generalizability of deepCorrRNA's method on other sequencing technology.

In deepCorrRNA, the precision of the insertion class plays a critical role in the output deletion rate, since false positive insertions would become deletions post-correction. An important factor that affects the precision of the insertion class is the clustering accuracy. When a read that should not belong to the same cluster as the other reads is clustered with them, the read could have a region of insertions when aligned to the consensus of the cluster, although this region is not insertions with respect to the true read. This could confuse the model learning and prediction. Since the model performance can be somewhat limited by the clustering accuracy, it would be beneficial to develop a new clustering tool for RNA-seq long reads that performs accurate isoform-based clustering and is able to handle large data sets. This is an interesting direction for future work.

To further improve deepCorrRNA's model performance on the minority classes, other approaches may also be explored, such as optimized loss functions for imbalanced data [53]. Furthermore, the feature set of the model could be further extended when additional sequence-level error-related factors can be identified. The current model is trained with ONT cDNA reads. deepCorrRNA can be extended to ONT direct RNA sequencing technology by retraining the model using direct RNA sequencing reads while encoding RNA base letters. These are more directions for future work.

The new algorithms, computational methods, data analysis, and ideas presented in this dissertation will improve the study of transcriptomes to gain new biological insights as the cost of third-generation sequencing decreases and data availability increases. Analyzing the novel isoforms identified by Scallop-LR through assembling single-molecule RNA-seq long reads could further our understanding on how altered expressions of genetic variations lead to complex human diseases, and thus help develop more effective treatments to those diseases. RNA-seq analysis tools including read mappers, transcript assemblers, and expression abundance estimators

can be validated and optimized using the representative RNA-seq samples selected by our hierarchical representative set selection method; these tools can be used in transcription profiling that may lead to the discovery of novel biomarkers, help predict therapy response, or help identify the molecular mechanisms leading to a disease. *De novo* error correction of RNA-seq long reads by deepCorrRNA can improve the transcriptome analyses of diverse non-model organisms; studying the transcriptomes of non-model organisms could address a wide range of important evolutionary and ecological questions.

Bibliography

[1] S.L. Amarasinghe, S. Su, X. Dong, et al. Opportunities and challenges in long-read sequencing data analysis. *Genome Biology*, 21:30, 2020. 1.1, 2.1

[2] D. Antipov et al. hybridSPAdes: an algorithm for hybrid assembly of short and long reads. *Bioinformatics*, 32(7):1009–1015, 2016. 1.1, 2.1, 5

[3] K. Au et al. Improving PacBio long read accuracy by short read alignment. *PLoS ONE*, 7 (10):e46679, 2012. 1.3

[4] K. Au et al. Characterization of the human ESC transcriptome by hybrid sequencing. *PNAS*, 110(50):E4821–E4830, 2013. 1.1, 2.1

[5] G. Baid, D.E. Cook, K. Shafin, et al. DeepConsensus improves the accuracy of sequences with a gap-aware sequence transformer. *Nature Biotechnology*, 2022. doi: https://doi.org/10.1038/s41587-022-01435-7. 4.2.3

[6] E. Bao and L. Lan. HALC: high throughput algorithm for long read error correction. *BMC Bioinformatics*, 18(1):204, 2017. 1.3

[7] T. Barrett, K. Clark, R. Gevorgyan, V. Gorelenkov, et al. BioProject and BioSample databases at NCBI: facilitating capture and organization of metadata. *Nucleic Acids Research*, 40(D1):D57–D63, 2012. 3.1

[8] Miroslav Blumenberg. *Transcriptome Analysis*. IntechOpen, Rijeka, 2019. 1

[9] C. Boutsidis, M. Mahoney, and P. Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of SODA*, pages 968–977, 2009. 1.4, 3.1

[10] A. Bronstein, M. Bronstein, and R. Kimmel. *Numerical geometry of non-rigid shapes*. Springer-Verlag New York, New York, 2009. 3.2.3

[11] O. Brosh, D.K. Fabian, R. Cogni, et al. A novel transposable element-mediated mechanism causes antiviral resistance in *Drosophila* through truncating the Veneno protein. *PNAS*, 119(29):e2122026119, 2022. 4.3.3

[12] E. Bushmanova, D. Antipov, A. Lapidus, and A. Prjibelski. rnaSPAdes: a de novo transcriptome assembler and its application to RNA-Seq data. *GigaScience*, 8(9):giz100, 2019. 1.2

[13] E. Bushmanova et al. rnaQUAST: a quality assessment tool for *de novo* transcriptome assemblies. *Bioinformatics*, 32(14):2210–2212, 2016. 2.2.2

[14] B. Bushnell. BBMap: a fast, accurate, splice-aware aligner. *9th Annual Genomics of*

Energy and Environment Meeting, pages LBNL–7065E, 2014. 1.2, 2.2.4

[15] S. Canzar, S. Andreotti, D. Weese, et al. CIDANE: comprehensive isoform discovery and abundance estimation. *Genome Biology*, 17:16, 2016. 1.2

[16] Z. Chang, G. Li, J. Liu, et al. Bridger: a new framework for de novo transcriptome assembly using RNA-seq data. *Genome Biology*, 16:30, 2015. 1.2

[17] CS. Chin, D. Alexander, P. Marks, et al. Nonhybrid, finished microbial genome assemblies from long-read SMRT sequencing data. *Nature Methods*, 10:563–569, 2013. 1.3

[18] H. Cho et al. High-resolution transcriptome analysis with long-read RNA sequencing. *PLOS One*, 9(9):e108095, 2014. 1.1, 2.1

[19] I. Choi, A. Ponsero, M. Bomhoff, K. Youens-Clark, J. Hartman, et al. Libra: scalable k-mer–based tool for massive all-vs-all metagenome comparisons. *GigaScience*, 8(2): giy165, 2019. 3.2.2

[20] V. D'Angeli, E. Monzón-Casanova, L.S. Matheson, et al. Polypyrimidine tract binding protein 1 regulates the activation of mouse CD8 T cells. *Eur. J. Immunol.*, 52(7):1058–1068, 2022. 4.3.3, 4.3.3

[21] M. Daszykowski, B. Walczak, and D. Massart. Representative subset selection. *Analytica Chimica Acta*, 468(1):91–103, 2002. 1.4, 3.1

[22] I. de la Rubia, A. Srivastava, W. Xue, et al. RATTLE: reference-free reconstruction and quantification of transcriptomes from Nanopore sequencing. *Genome Biology*, 23:153, 2022. 1.3, 4.1

[23] C. Delahaye and J. Nicolas. Sequencing DNA with nanopores: Troubles and biases. *PLoS ONE*, 16(10):e0257521, 2021. 4.1, 4.2.2, 4.2.2

[24] A. Dobin et al. STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, 29(1):15–21, 2013. 1.2, 2.2.4

[25] E. Elhamifar, G. Sapiro, and R. Vidal. Finding exemplars from pairwise dissimilarities via simultaneous sparse recovery. In *Advances in Neural Information Processing Systems 25 (NIPS)*, pages 19–27, 2012. 1.4, 3.1

[26] E. Elhamifar, G. Sapiro, and R. Vidal. See all by looking at a few: Sparse modeling for finding representative objects. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1600–1607, 2012. 1.4, 3.1

[27] B. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315: 972–976, 2007. 1.4, 3.1

[28] S. Fu et al. IDP-denovo: *de novo* transcriptome assembly and isoform annotation by hybrid sequencing. *Bioinformatics*, 34(13):2168–2176, 2018. 1.2, 2.4, 5

[29] S. Garcia, J. Derrac, J. Cano, and F. Herrera. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(30):417–435, 2012. 1.4, 3.1

[30] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th International Conference on Artificial Intelligence*

and Statistics, pages 249–256, 2010. 4.2.3

[31] S. Goodwin, J. Gurtowski, S. Ethe-Sayers, et al. Oxford Nanopore sequencing, hybrid error correction, and de novo assembly of a eukaryotic genome. *Genome Res*, 25(11): 1750–1756, 2015. 1.3

[32] M. Grabherr, B. Haas, M. Yassour, et al. Full-length transcriptome assembly from RNA-Seq data without a reference genome. *Nature Biotechnology*, 29(7):644–652, 2011. 1.2, 2.4, 5

[33] M. Guttman, M. Garber, J. Levin, et al. Ab initio reconstruction of cell type–specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs. *Nature Biotechnology*, 28:503–510, 2010. 1.2

[34] T. Hackl, R. Hedrich, J. Schultz, et al. proovread: large-scale high-accuracy PacBio correction through iterative short read consensus. *Bioinformatics*, 30(21):3004–3011, 2014. 1.3

[35] B. Hie, H. Cho, B. DeMeo, B. Bryson, and B. Berger. Geometric sketching compactly summarizes the single-cell transcriptomic landscape. *Cell Systems*, 8(6):483–493.E7, 2019. 1.4, 3.1, 3.2.1

[36] JF. Hughes et al. Chimpanzee and human Y chromosomes are remarkably divergent in structure and gene content. *Nature*, 463(7280):536–9, 2010. 2.2.3

[37] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9): 850–863, 1993. 3.2.1

[38] D. Kim et al. TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome Biology*, 14(4):R36, 2013. 1.2, 2.2.4

[39] D. Kim et al. HISAT: a fast spliced aligner with low memory requirements. *Nature Methods*, 12(4):357–360, 2015. 1.2, 2.2.4

[40] D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv*, page 1412.6980, 2014. 4.2.3

[41] MA. Komor et al. Identification of differentially expressed splice variants by the proteogenomic pipeline splicify. *Mol Cell Proteomics*, 16(10):1850–1863, 2017. 2.2.3

[42] S. Koren, M. Schatz, B. Walenz, et al. Hybrid error correction and de novo assembly of single-molecule sequencing reads. *Nature Biotechnology*, 30:693–700, 2012. 1.3, 4.1

[43] S. Koren, B.P. Walenz, K. Berlin, et al. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. *Genome Research*, 27(5):722–736, 2017. 1.1, 1.3, 2.1, 4.1

[44] P. K. Korhonen et al. Common workflow language (CWL)-based software pipeline for *de novo* genome assembly from long- and short-read data. *GigaScience*, 8(4):giz014, 2019. 1.1, 2.1

[45] S. Kovaka, A.V. Zimin, G.M. Pertea, et al. Transcriptome assembly from long-read RNA-seq alignments with StringTie2. *Genome Biology*, 20:278, 2019. 1.2

[46] K. Križanović et al. Evaluation of tools for long read RNA-seq splice-aware alignment. *Bioinformatics*, 34(5):748–754, 2018. 2.2.4

[47] A. Kuosmanen et al. On using longer RNA-seq reads to improve transcript prediction accuracy. *9th International Joint Conference on Biomedical Engineering Systems and Technologies*, 3(BIOINFORMATICS):272–277, 2016. 2.1

[48] N. LaPierre, R. Egan, W. Wang, et al. De novo Nanopore read quality improvement using deep learning. *BMC Bioinformatics*, 20:552, 2019. 4.2.3

[49] C. Lee, C. Grasso, and M.F. Sharlow. Multiple sequence alignment using partial order graphs. *Bioinformatics*, 18(3):452–464, 2002. 1.3

[50] R.M. Leggett, D. Heavens, M. Caccamo, et al. NanoOK: multi-reference alignment analysis of nanopore sequencing data, quality and error profiles. *Bioinformatics*, 32(1):142–144, 2016. 4.1, 4.2.2

[51] R. Leinonen et al. The sequence read archive. *Nucleic Acids Res*, 39(suppL1):D19–21, 2011. 1.4, 1.5, 2.1, 2.2.3, 3.1

[52] H. Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34:3094–3100, 2018. 1.2, 2.2.4, 4.2.7

[53] M. Li, X. Zhang, C. Thrampoulidis, et al. AutoBalance: optimized loss functions for imbalanced data. In *Proceedings of 35th Conference on NeurIPS*, 2021. 4.4, 5

[54] Q. Li, V. Kecman, and R. Salman. A chunking method for Euclidean distance matrix calculation on large dataset using multi-GPU. In *The Ninth International Conference on Machine Learning and Applications*, pages 208–213, 2010. 3.1

[55] W. Li, J. Feng, and T. Jiang. IsoLasso: a LASSO regression approach to RNA-Seq based transcriptome assembly. *J. Computational Biology*, 18(11):1693–1707, 2011. 1.2

[56] L. Lima, C. Marchet, S. Caboche, et al. Comparative assessment of long-read error correction software applied to Nanopore RNA-sequencing data. *Briefings in Bioinformatics*, 21(4):1164–1181, 2020. 1.1, 1.3, 4.1

[57] B. Liu, Y. Liu, J. Li, et al. deSALT: fast and accurate long transcriptomic read alignment with de Bruijn graph-based index. *Genome Biology*, 20:274, 2019. 2.3.4, 2.5.4, 2.36

[58] J. Liu, T. Yu, T. Jiang, et al. TransComb: genome-guided transcriptome assembly via combing junctions in splicing graphs. *Genome Biology*, 17:213, 2016. 1.2

[59] N. Loman, J. Quick, and J. Simpson. A complete bacterial genome assembled *de novo* using only nanopore sequencing data. *Nature Methods*, 12:733–735, 2015. 1.3, 4.1

[60] M.A. Madoui, S. Engelen, C. Cruaud, et al. Genome assembly using Nanopore-guided long and error-free DNA reads. *BMC Genomics*, 16(1):327, 2015. 1.3, 4.1

[61] G. Marçais and C. Kingsford. A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. *Bioinformatics*, 27(6):764–770, 2011. 3.2.2

[62] L. Maretty, J.A. Sibbesen, and A. Krogh. Bayesian transcriptome assembly. *Genome Biology*, 15:501, 2014. 1.2

[63] J. Marić, I. Sović, K. Križanović, et al. Graphmap2 - splice-aware RNA-seq mapper for

long reads. *bioRxiv*, page 720458, 2019. 1.2

[64] M. Martin. Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet.journal*, 17(1):10–12, 2011. 4.2.4

[65] P. Morisse, C. Marchet, A. Limasset, et al. Scalable long read self-correction and assembly polishing with multiple sequence alignment. *Scientific Reports*, 11:761, 2021. 1.3, 4.1

[66] T. O'Grady et al. Global transcript structure resolution of high gene density genomes through multi-platform data integration. *Nucleic Acids Res*, 44(18):e145, 2016. 2.2.3

[67] Oxford Nanopore Technologies. Pychopper: a tool to identify, orient and trim full-length Nanopore cDNA reads. https://github.com/epi2me-labs/pychopper, 2019. 4.2.4

[68] Pacific Biosciences. Understanding accuracy in SMRT sequencing. https://www.pacb.com/wp-content/uploads/2015/09/Perspective_UnderstandingAccuracySMRTSequencing.pdf, 2015. 2.2.4

[69] Pacific Biosciences. SMRT Tools Reference Guide v5.1.0. https://www.pacb.com/wp-content/uploads/SMRT_Tools_Reference_Guide_v510.pdf, 2018. 1.1, 1.5, 2.1

[70] Pacific Biosciences. ARCHIVED: Intro to the Iso-Seq Method: Full-length transcript sequencing. https://www.pacb.com/blog/intro-to-iso-seq-method-full-leng, June 2, 2014. 2.1

[71] G. Pai, R. Talmon, A. Bronstein, and R. Kimmel. DIMAL: Deep isometric manifold learning using sparse geodesic sampling. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 819–828, 2019. 3.2.3

[72] F. Pan, W. Wang, A. Tung, and J. Yang. Finding representative set from massive data. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, page 8, 2005. 1.4, 3.1

[73] Q. Pan et al. Deep surveying of alternative splicing complexity in the human transcriptome by high-throughput sequencing. *Nature Genetics*, 40(12):1413–1415, 2008. 1

[74] G Pertea and M. Pertea. GFF Utilities: GffRead and GffCompare. *F1000Research*, 9:304, 2020. doi: 10.12688/f1000research.23297.1. 2.2.2

[75] M. Pertea et al. StringTie enables improved reconstruction of a transcriptome from RNA-seq reads. *Nature Biotechnology*, 33(3):290–295, 2015. 1.2, 1.5, 2.1, 2.2.4

[76] M. Pertea et al. Transcript-level expression analysis of RNA-seq experiments with HISAT, StringTie, and Ballgown. *Nature Protocols*, 11(9):1650–1667, 2016. 1.2, 2.1, 2.2.4

[77] F.J. Rang, W.P. Kloosterman, and J. de Ridder. From squiggle to basepair: computational approaches for improving nanopore sequencing read accuracy. *Genome Biology*, 19:90, 2018. 4.1

[78] A Rhoads and K. Au. PacBio sequencing and its applications. *Genomics Proteomics Bioinformatics*, 13:278–289, 2015. 1.1, 2.1, 2.3.1, 5

[79] G. Robertson, J. Schein, R. Chiu, et al. De novo assembly and analysis of RNA-seq data. *Nature Methods*, 7:909–912, 2010. 1.2

[80] K. Sahlin and P. Medvedev. De novo clustering of long-read transcriptome data using a greedy, quality value-based algorithm. *J of Comp Biology*, 27(4):472–484, 2020. 4.2.5

[81] K. Sahlin and P. Medvedev. Error correction enables use of Oxford Nanopore technology for reference-free transcriptome analysis. *Nature Communications*, 12:2, 2021. 1.1, 1.3, 1.5, 4.1

[82] L. Salmela and E. Rivals. LoRDEC: accurate and efficient long read error correction. *Bioinformatics*, 30(24):3506–3514, 2014. 1.3, 4.1

[83] L. Salmela, R. Walve, E. Rivals, et al. Accurate self-correction of errors in long reads using de Bruijn graphs. *Bioinformatics*, 33(6):799–806, 2017. 1.3, 4.1

[84] J. Schreiber, J. Bilmes, and W. Noble. apricot: Submodular selection for data summarization in Python. *Journal of Machine Learning Research*, 21(161):1–6, 2020. 1.4, 3.1

[85] M. Schulz, D. Zerbino, M. Vingron, and E. Birney. Oases: robust de novo RNA-seq assembly across the dynamic range of expression levels. *Bioinformatics*, 28(8):1086–1092, 2012. 1.2

[86] JS. Seo et al. *De novo* assembly and phasing of a Korean human genome. *Nature*, 538 (7624):243–247, 2016. 2.2.3

[87] C. Sessegolo, C. Cruaud, C. Da Silva, et al. Transcriptome profiling of mouse samples using nanopore sequencing of cDNA and RNA molecules. *Scientific Reports*, 9:14908, 2019. 4.3.3

[88] Mingfu Shao and Carl Kingsford. Accurate assembly of transcripts through phase-preserving graph decomposition. *Nature Biotechnology*, 35:1167–1169, 2017. 1.2, 1.5, 2.1

[89] D. Sharon et al. A single-molecule long-read survey of the human transcriptome. *Nature Biotechnology*, 31(11):1009–1014, 2013. 1.2, 2.1, 2.4

[90] L. Shi et al. Long-read sequencing and *de novo* assembly of a Chinese genome. *Nature Communications*, 7:12065, 2016. 1.1, 2.1, 2.2.3

[91] D. Sim, O. Kwon, and R. Park. Object matching algorithms using robust Hausdorff distance measures. *IEEE Transactions on Image Processing*, 8(3):425–429, 1999. 3.2.1

[92] R. Smith-Unna et al. TransRate: reference-free quality assessment of *de novo* transcriptome assemblies. *Genome Research*, 26(8):1134–1144, 2016. 2.2.2

[93] C. Soneson, Y. Yao, A. Bratus-Neuenschwander, et al. A comprehensive examination of Nanopore native RNA sequencing for characterization of complex transcriptomes. *Nature Communications*, 10:3359, 2019. 5

[94] A.D. Tang, C.M. Soulette, M.J. van Baren, et al. Full-length transcript characterization of SF3B1 mutation in chronic lymphocytic leukemia reveals downregulation of retained introns. *Nature Communications*, 11:1438, 2020. 1.3, 4.1

[95] M. Tardaguila et al. SQANTI: extensive characterization of long-read transcript sequences for quality control in full-length transcriptome identification and quantification. *Genome Research*, 28:396–411, 2018. 2.2.2

[96] H. Tilgner et al. Defining a personal, allele-specific, and single-molecule long-read transcriptome. *PNAS*, 111(27):9869–9874, 2014. 1.1, 2.1

[97] G. Tischler and E.W. Myers. Non hybrid long read consensus using local de Bruijn graph assembly. *bioRxiv*, page 106252, 2017. 1.3

[98] A.I. Tomescu, A. Kuosmanen, R. Rizzi, et al. A novel min-cost flow method for estimating transcript expression with RNA-Seq. *BMC Bioinformatics*, 14(Suppl 5):S15, 2013. 1.2

[99] C. Trapnell, B. Williams, G. Pertea, et al. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature Biotechnology*, 28:511–515, 2010. 1.2

[100] E. Tseng et al. Altered expression of the FMR1 splicing variants landscape in premutation carriers. *Biochim Biophys Acta*, 1860(11):1117–1126, 2017. 1.1, 2.1

[101] Laura H. Tung and Carl Kingsford. Practical selection of representative sets of RNA-seq samples using a hierarchical approach. *Bioinformatics*, 37(Supplement_1):i334–i341, 2021. doi: 10.1093/bioinformatics/btab315. 3

[102] Laura H. Tung, Mingfu Shao, and Carl Kingsford. Quantifying the benefit offered by transcript assembly with Scallop-LR on single-molecule long reads. *Genome Biology*, 20: 287, 2019. doi: 10.1186/s13059-019-1883-0. 1.2, 2

[103] R. Vaser, I. Sović, N. Nagarajan, and M. Šikić. Fast and accurate de novo genome assembly from long uncorrected reads. *Genome Research*, 27(5):737–746, 2017. 4.2.5

[104] J. Wagner, N.D. Olson, L. Harris, et al. Benchmarking challenging small variants with linked and long reads. *Cell Genomics*, 2(5):100128, 2022. 4.3.2

[105] B. Wang et al. Unveiling the complexity of the maize transcriptome by single-molecule long-read sequencing. *Nature Communications*, 7:11708, 2016. 1.1, 2.1

[106] L. Wang, L. Qu, L. Yang, et al. NanoReviser: An error-correction tool for Nanopore sequencing based on a deep learning algorithm. *Front Genet*, 11:900, 2020. 1.3, 4.1, 4.2.3

[107] Y. Wang, S. Tang, Y. Zhang, J. Li, and D. Wang. Representative selection based on sparse modeling. *Neurocomputing*, 139:423–431, 2014. 1.4, 3.1

[108] Y. Wang, Y. Zhao, A. Bollas, Y. Wang, and K.F. Au. Nanopore sequencing technology, bioinformatics and applications. *Nature Biotechnology*, 39:1348–1365, 2021. 1.1, 1.1, 4.1

[109] J. Weirather et al. Characterization of fusion genes and the significantly expressed fusion isoforms in breast cancer by hybrid sequencing. *Nucleic Acids Res*, 43(18):e116, 2015. 1.1, 2.1

[110] R. R. Wick et al. Unicycler: Resolving bacterial genome assemblies from short and long sequencing reads. *PLoS Computational Biology*, 13(6):e1005595, 2017. 1.1, 2.1

[111] R.E. Workman, A.D. Tang, P.S. Tang, et al. Nanopore native RNA sequencing of a human poly(A) transcriptome. *Nature Methods*, 16:1297–1305, 2019. 4.2.4, 4.3.1, 4.3.2

[112] TD Wu and CK Watanabe. GMAP: a genomic mapping and alignment program for mRNA and EST sequences. *Bioinformatics*, 21(9):1859–1875, 2005. 1.2, 2.2.2

[113] D. Wyman and A. Mortazavi. TranscriptClean: variant-aware correction of indels, mismatches and splice junctions in long-read transcripts. *Bioinformatics*, 35(2):340–342, 2019. 1.3, 4.1

[114] C.L. Xiao, Y. Chen, S.Q. Xie, et al. MECAT: fast mapping, error correction, and de novo assembly for single-molecule sequencing reads. *Nature Methods*, 14:1072–1074, 2017. 1.3, 4.1

[115] Y. Xie, G. Wu, J. Tang, et al. SOAPdenovo-Trans: de novo transcriptome assembly with short RNA-Seq reads. *Bioinformatics*, 30(12):1660–1666, 2014. 1.2

[116] C. Yang, J. Chu, R.L. Warren, and I. Birol. NanoSim: nanopore sequence read simulator based on statistical characterization. *GigaScience*, 6(4):gix010, 2017. 4.1, 4.2.2

[117] A. Zimin et al. The MaSuRCA genome assembler. *Bioinformatics*, 29(21):2669–2677, 2013. 1.1, 2.1

[118] A. V. Zimin et al. Hybrid assembly of the large and highly repetitive genome of *Aegilops tauschii*, a progenitor of bread wheat, with the MaSuRCA mega-reads algorithm. *Genome Research*, 27(5):787–792, 2017. 1.1, 2.1, 5