Calling Variants from both Short and Long Reads

Jiajia Xu (u6528982)
Supervised by
Dr. Yu Lin, Dr Hardip Patel

COMP8755 Individual Computing Project (Semester 2,2019)
Oct 2019
A thesis submitted in part fulfillment of the degree of
Master of Computing
Research School of Computer Science
College of Engineering and Computer Science
The Australian National University

Contents

Acknowledgement

I really appreciate the opportunity of working with my two extraordinary supervisors Dr Yu Lin and Dr Hardip Patel. Their encouragement and recognition to me make me feel I am the best student in ANU. I wish this project could have lasted for longer such that I could learn more from them. I really enjoy the time working with them, exploring something that no one has ever done.

As a one semester project, everything has to be done in 12 weeks, which is very challenging for me. Although Dr Yu Lin is lecturing two courses during this semester, he still meets with me frequently to give me instant feedback to make sure I can proceed smoothly and on the right track. He is always very happy whenever I update my process to him, he is more like a friend rather than supervisor. Dr Hardip Patel kindly provided me with Server from NCIG and a seat next to him such that I could seek help from him easily. He helped with me with Genomics knowledge I totally had no idea before the project such that I could focus on experiments and data analysis. His kind help is the main reason I can finish this project on time. The thing I learnt from him is his extreme dedication to work but still ready to offer help any time I ask him.

I would like to thank Yanbo Li for teaching me what tools to use for simulating artificial genome and Dr Minh Bui for being my report examiner.

Last but not least, the unconditional love and support from my parents and my wife is why I can't stop chasing the truth which sets me free and rewriting the stars.

Abstract

With the rapid advances in next-generation sequencing (NGS) and now the third-generation sequencing, many statistical models have been proposed for variant calling in human genomes. Calling genetic variants from sequencing reads is challenging due to the ambiguity between real variants and errors or misalignments in the reads.

In 2018, an approach based on deep convolutional neural network, DeepVariant, has been developed to call genetic variants using pileups of aligned reads. It outperformed existing state-of-the-art tools. However, the performance of DeepVariant is mainly benchmarked against "confident regions" in the genome and its performance on other genomic regions is unknown. Second, DeepVariant did not make use of various genomic features (e.g., annotation information) in calling variants. Third, although DeepVariant can use both the short reads (e.g., Illumina®) and long reads (PacBio® and Oxford ONT®) separately, but it does not support incorporating both short and long reads from the same individual to call variants.

In this project, I address the above limitations of DeepVariant. Extensive benchmarking was performed to evaluate the performance of DeepVariant, which uses both publicly available datasets and simulated datasets with known truth and without restricting to "confident regions". I observed that DeepVariant achieves higher accuracy using short-read datasets than long-read datasets. Secondly, I have leveraged annotation information of genomes and variants (e.g., repeat/segmental duplication, genotype quality, read depth and variant allele fraction) to improve variant calling performance when both short and long reads are available for the same individual. The classifier I built is able to differentiate True Positives and False Positives variant calls to 97.359% accuracy. This work lays solid foundation for variant discovery and it will be incorporated into a practical pipeline for genomic analysis for the National Centre for Indigenous Genomics.

Chapter 1

Introduction

Precision medicine is an emerging approach for disease treatment and prevention that considers individual variability in the genome, environment, and lifestyle for each person. While current technology doesn’t allow us to obtain individual genome as a whole directly, what we can get are small pieces of genomic reads. These small reads are compared against the human reference genome to find differences (variations) for further interrogations. Calling genetic variants from next-generation sequencing (NGS) data has proven challenging because NGS reads not only have errors but also arise from a complex error process that depends on properties of the instrument, preceding data processing tools, and the genome sequence itself.

Standard human reference genome is composed of more than 50% repeat elements, including transposable elements and long duplicated regions. Ambiguous alignments due to repeat elements in the reference genome form a major source of errors in variant detection. Segmental Duplications (SDs) are long DNA sequences which have almost identical sequences (90-100%). They exist in multiple locations and are typically of length greater than 1kb. SDs can be tandem or interspersed and can be inter-chromosomal or intra-chromosomal [1].

1.1 Motivations

To solve the problem of identifying true genetic variations from errors in aligned reads, traditional variant callers use a variety of statistical techniques to model these error processes. Take the widely used GATK as an example [8], it uses logistic regression to model base errors, hidden Markov models to compute genotype likelihoods, and naive Bayes classification to identify variants. They are then filtered to remove likely false positives using a Gaussian mixture model with hand-crafted features capturing common error modes. All these techniques make the GATK a very promising tool on the Illumina sequencing platform but yet still imperfect. Generalizing these models to other sequencing technologies (e.g. PacBio) has been difficult due to the need for manual retuning or extending these statistical models [2].

Recently developed DeepVariant uses a single deep learning model to replace the assortment of statistical models. Deep learning is one of the machine learning techniques that can automatically extract features together with other mathematical patterns to make accurate predictions on classification or regression problems. DeepVariant begins with looking for candidate SNPs and INDELs in aligned reads. It will then predict one of the three diploid genotypes at a locus using a pileup image of the reference and reads data near each candidate variant. DeepVariant is able to call variants more accurately than GATK best-practices pipeline [3]. In addition, the model is generalizable to different sequencing technology (e.g. PacBio). To use this advantage, I want to apply DeepVariant on both Illumina and PacBio sequencing reads from the same individual to get a compound result. While there is still limited benchmarking on Deepvariant, a set of experiments are required to assess its performance. DeepVariant model is trained by authors of the software with labeled true genotypes without incorporating genomic annotations. I would like to incorporate annotation information of the human genome to improve overall variant calling results.

1.2 Objectives

Objectives of this project were achieved through following aims:

- Run DeepVariant using **empirical** genome sequence data for one individual.
    - Use both Illumina® and PacBio® data for the same individual and compare results with benchmark set provided by software authors to assess repeatability and replicability of DeepVariant analysis.

- Run DeepVariant using **simulated** genome sequence data for one individual.
    - Genome sequence data with same read lengths, coverage, error rates and mutational profile consistent with the human genome sequence data will provide absolute ground truth for variant calls contributing to model genomic properties and annotations to improve variant calls.

- Develop a classifier to differentiate between True Positives and False Positives
    - Use genomic properties extracted from simulated data to improve variant classification for improved accuracy in output

Chapter 2

Background

DNA sequencing and its applications in biomedical research and clinical genomics is increasing rapidly. One of the major implications of these advancements is the use genomics for predicting and preventing diseases and individualized therapeutics. Identification of variant in genome sequence data is one of the central data processing steps with room for improvements. Accurate variant identification can lead to reduced costs in the use of genomics. This project is aimed at improving variant calling process for genomic data.

2.1 Genome Sequencing

An organism's complete set of DNA is called its genome. Virtually every cell in humans contains a complete copy of the DNA inherited from the mother and the father. Genome consists of 23 pairs of chromosomes and are located in the nucleus of a human cell. If a cell's DNA is mutated, an abnormal protein may be produced, which can disrupt the body's usual processes and lead to a disease such as cancer [4].

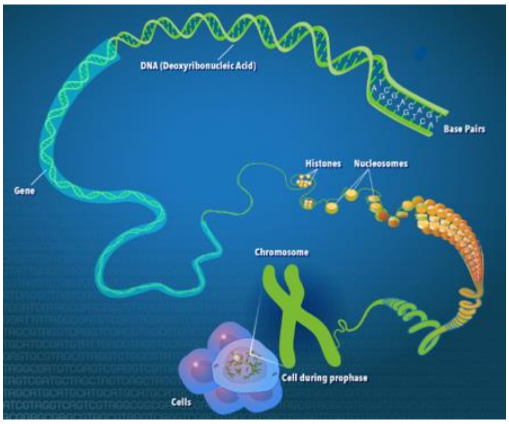


Figure 2.1: from DNA to Genome [5]

Genome sequencing is a technique used to infer the order of the bases in strands of DNAs that form chromosomes. There are three generations of sequencing technology (Table 2.1).

1. First generation sequencing can generate reads up to 1,000 base pair long with accuracy of 99.999%. High cost and low throughput renders this technique less suitable for large-scale application.
2. Next generation sequencing (NGS) has low cost and high throughput, with 100-400bp read lengths and accuracy from 99% to 99.99%, usually the longer the reads, lower the accuracy.
3. Third generation sequencing (TGS) can produce 1,000-100,000bp long reads. However, these reads contain high error rate of 5-15% [6].

Table 2.1: Sequencing technologies

Generation	Company (Technology)	Read length (bp)	Error rate (%)
1st	ABI (3730xl)	600-1000	0.1-1
2nd (NGS)	Roche (454)	230-400	1
	Illumina (HiSeq)	2×150	≥0.1
	ABI (SOLiD)	25-35	>0.01
3rd (TGS)	PacBio (SMRT)	~1,000-50,000	10-15
	Oxford (Nanopore)	~10,000-100,000	5-15

2.2 Variant calling and GATK

Biomedical research and clinical genomics use genome sequencing to search for genetic variations that may play a role in a disease or molecular phenotype. The disease-causing change may be as small as a single base pair substitution called Single Nucleotide Polymorphism (SNP), insertion/deletion of a single base pair to as large thousands of bases (INDEL).

A variant call is a decision that there is a nucleotide difference versus some reference at a given position in an individual genome. If there were very few errors in sequencing technology or during alignment, it is easy to decide a variant by simply counting the number of each allele. While both NGS and TGS reads have lots of sequencing and alignment errors, so a probabilistic method of computing genotype likelihood is developed, Genome Analysis Toolkit (GATK) is one of the most successful tools using this method [7].

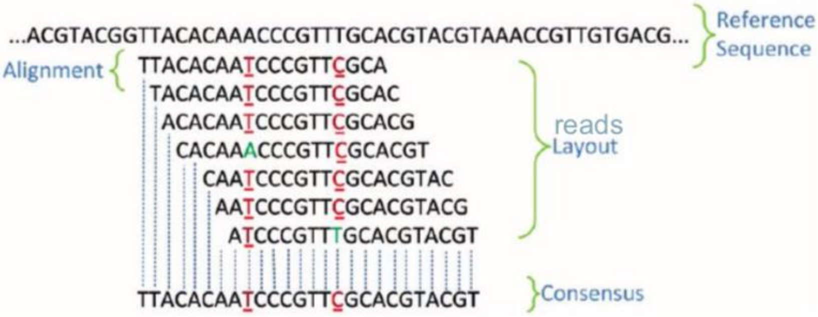


Figure 2.2: An example of calling variants from aligned reads against reference genome

GATK is using Bayesian estimation to find the most likely genotype, it calculates many parameters for each position of the genome. It can handle both SNP and INDEL calling, it uses standard input and output files and itself contains many tools for managing VCF files. That's why it has been used in many NGS projects, including the 1000 Genomes Project, The Cancer Genome Atlas, etc [8].

2.3 DeepVariant

DeepVariant is an analysis pipeline that uses a deep neural network to call genetic variants from next-generation and third-generation DNA sequencing data. It uses a set of programs to transform aligned sequencing reads into variant calls. DeepVariant relies on Nucleus, a library of Python and C++ code for reading and writing data in common genomics file formats (like SAM and VCF) designed for integration with the TensorFlow machine learning framework [9].

The overall workflow of DeepVariant is as follows. First, the aligned reads are scanned for sites that may be different from the reference genome. The read and reference data are encoded as a pileup image for each candidate variant site. A trained CNN calculates the genotype likelihoods for each site. A variant call is emitted if the most likely genotype is heterozygous or homozygous non-reference (Figure 2.3).

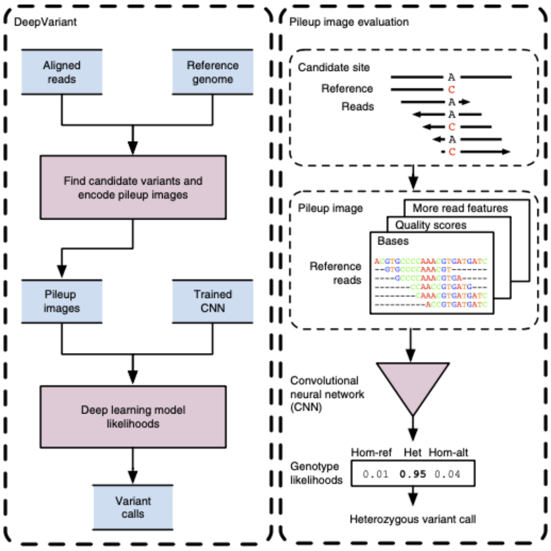


Figure 2.3: DeepVariant workflow [3]

DeepVariant version 0.8 was used in this project with WGS model and PacBio model. WGS model was trained with 12 * HG001 PCR-free, 2 * HG005 PCR-free, 4 * HG001 PCR+ and is best suited for Illumina Whole Genome Sequencing data. PacBio model was trained using HG002 genome with chromosomes 20, 21, 22 excluded [3]. PacBio model is not intensively trained, so its performance might not be as good as WGS model. And we should not test the same set of data on model which are also trained with, otherwise accuracy will be 100%. For example, we can’t test HG002 chromosomes 1-19 on PacBio model, while chromosomes 20, 21, 22 are fine. The training process of DeepVariant is stated in Figure 2.4.

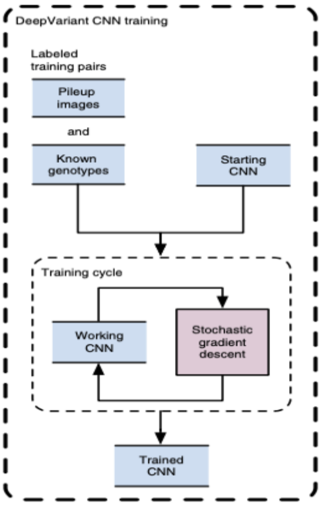


Figure 2.4: Training process of DeepVariant [3]

To compare the calling performance between DeepVariant and GATK on Illumina data, they are both tested on Genome in a Bottle benchmark sample NA12878 using 2x101 Illumina HiSeq data from the Platinum Genomes project, the precision-recall plot is shown below [3]. DeepVariant has better Recall and Precisions on both SNPs and INDELs.

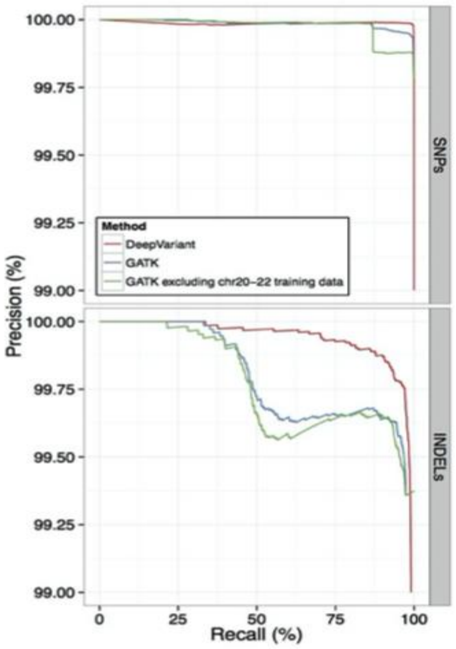


Figure 2.5: Precision-recall plot for DeepVariant (red) and GATK (green, blue) calls.

2.4 Confident Regions

At the time of comparing Query VCF file with Truth VCF file, whenever we are using Truth VCF file from Platinum Genomes or Genome in a Bottle, it comes with a Truth confident regions BED file. This is a concept we have to follow when calling variants for any real human genome, because confident regions cover the regions whose state has been clearly defined by truth set pipeline, including both variant non-reference sites and homozygous reference positions. Any variant called in confident regions that is not in the truth set can be regarded as false positive (FP), and any variant calling outside the confident region is not assessed [11].

This means we are not able to assess all variant calls made by a caller, it may give us a “fake” performance. To overcome this, this project simulates artificial human genomes as well as ground truth VCF files. With ground truth VCF file, we are able to assess the “real” performance of a variant caller and compare the performances with and without adding confident regions.

Chapter 3

Methodology

In this chapter, the procedures of approaches towards all staged objectives (Chapter 1.2 Objectives) are outlined. Besides, the tools and techniques involved in all experiments are clearly explained.

3.1 Procedure Overview

To have a comprehensive benchmarking of variant calling performance of DeepVariant, all experiments are run on both short and long reads (from Illumina and PacBio respectively). **First experiment** is run on a publicly available human genome reads of HG002 chromosome 20 with reference to hs37d5 (also known as GRCh37/hg19) using WGS and PacBio models. The results are benchmarked with officially released results from DeepVariant Github page [9] to check repeatability and replicability of DeepVariant.

As mentioned in Chapter 2.4, due to the constrains of confident regions on Truth VCF of real human genome, we decide to run experiment on artificial genome with DeepVariant to get a “real” result. **Second experiment** is carried out on reads of chromosome 20 as well from an artificial genome with SNP variation only regarding reference genome hs37d5. The aim of experiment 2 is check

DeepVariant performance on SNPs. **Third experiment** is run on reads of another artificial genome with both SNP and INDEL variations to check DeepVairant performance on SNPs and INDELs. Results from both experiments are analyzed together with genome annotation and variant information.

The final experiment is using reads of an artificial genome simulating on empirical sample reads used in National Center for Indigenous Genomics (NCIG) to test robustness of DeepVariant performance. The reads coverage and length will be slightly different from first 3 experiments (Table 3.1).

Table 3.1: Experiments Overview

Experiment	Genome	Illumina reads coverage/length	PacBio reads coverage/type	Confident regions	Purpose
1	Human genome HG002	50/150	30/CCS	Yes	repeatability and replicability
2	Artificial genome with SNP	50/150	30/CCS	No	Performance on SNP
3	Artificial genome with SNP & INDEL	50/150	30 & 100/CCS	No &Yes	Performance on SNP & INDEL
4	Artificial genome with SNP & INDEL	40/150	30/CLR	No	Robustness

All experiments are carried out with similar workflow. First, aligned reads from Illumina and PacBio are fed into DeepVariant WGS and PACBIO model respectively. Then the original VCF files generated from DeepVariant is compared with Truth VCF through Hap.py. After that, genome knowledge is integrated into the analysis of Illumina and PacBio variant calls. From the analysis, we expect to develop an algorithm to improve overall calling performance.

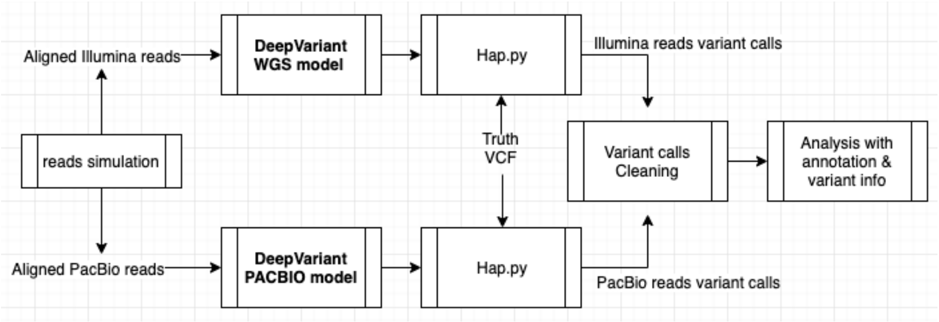


Figure 3.1: Experiment workflow

3.2 Reads simulation for artificial genome

Since experiments 2, 3 and 4 are using reads from artificial genome, simulation of aligned reads is necessary. In this project, hs37d5.fa is used as reference genome for all experiments. First, SNP and INDEL variants are added into reference genome to generate artificial genome (fasta file), from both reference and artificial genome, a ground truth VCF comes out. Then both Illumina and PacBio reads (fastq file) are generated from artificial genome. These reads are then aligned back to reference genome, getting sorted and indexed (bam file).

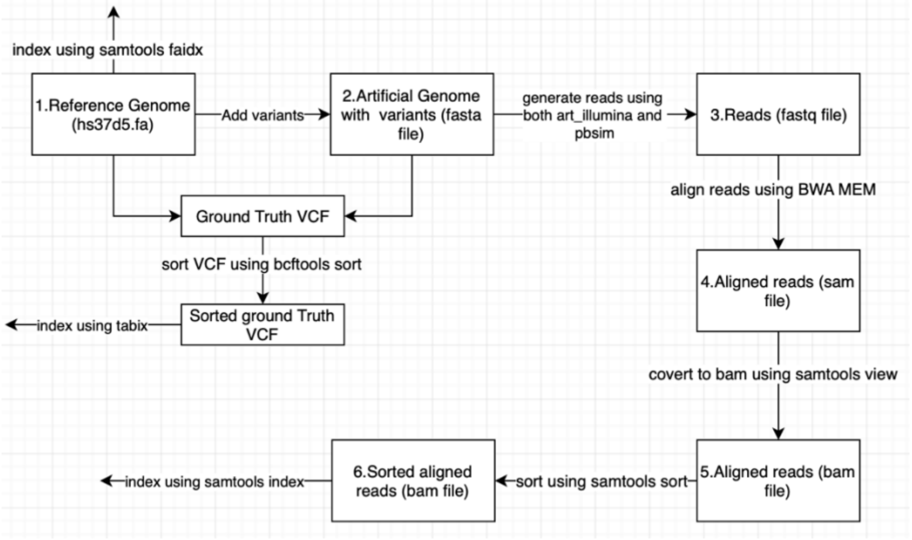

index using samtools faidx
1.Reference Genome (hs37d5.fa)
Add variants
2.Artificial Genome with variants (fasta file)
generate reads using both art_illumina and pbsim
3.Reads (fastq file)
align reads using BWA MEM
Ground Truth VCF
sort VCF using bcftools sort
Sorted ground Truth VCF
index using tabix
4.Aligned reads (sam file)
covert to bam using samtools view
5.Aligned reads (bam file)
sort using samtools sort
6.Sorted aligned reads (bam file)
index using samtools index

Figure 3.2: Simulation reads of artificial genome

3.3 Running DeepVariant and Hap.py

After getting reads ready, we can start to run DeepVariant, three files need to be provided as inputs:

1. A reference genome in FASTA format and its corresponding index file (.fai)
2. An aligned reads file in BAM format and its corresponding index file (.bai). The reads must be aligned to the reference genome.
3. A model checkpoint for DeepVariant.

Three steps will be executed:

1. "make_examples". It creates small variant candidates and stores them in TensorFlow format.
2. "call_variants". This applys DeepVariant to call variants.
3. "postprocess_variants". This converts data from TensorFlow format to VCF

The output of DeepVariant is a list of all variant calls in VCF format [9].

After getting resulting VCF files from DeepVariant, we need to benchmark them against gold standard truth datasets. While we can't compare VCF records individually due to complex variant representations. In a VCF file, we describe two haplotype sequences by means of REF-ALT pairs and genotypes. These variant calls do not uniquely represent the haplotype sequences, different variant calling methods may produce different variant representations.

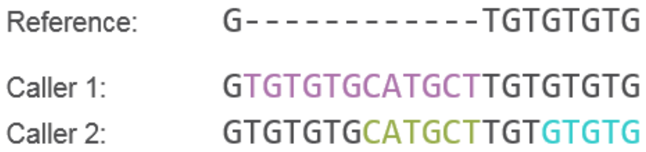
Hap.py is a set of programs based on htslib to benchmark variant calls against gold standard truth datasets. From the VCF file, Hap.py produces a graph-based representation of the VCF alleles, create all possible haplotype sequences, and compare these by alignment / exact matching. Here is an example where Hap.py (https://github.com/Illumina/hap.py) is needed:

Caller 1:

CHROM	***POS***	***REF***	***ALT***	***GT***
chrQ	*10*	*G*	*GTGTGTGCATGCT*	*0/1*

Caller 2:

CHROM	***POS***	***REF***	***ALT***	***GT***
chrQ	*16*	*G*	*GCATGCT*	*0/1*
chrQ	*19*	*T*	*TGTGTG*	*0/1*



```
Reference:     G------------TGTGTGTG

Caller 1:      GTGTGTGCATGCTTGTGTGTG
Caller 2:      GTGTGTGCATGCTTGTGTGTG
```

Both callers in this example are able to produce the same ALT sequences, we are able to match them up with Hap.py [10].

Hap.py requires the following files as inputs: 1. Query VCF file. 2. Truth VCF file. 3. Truth confident regions BED file (optional).

3.4 Variant calls cleaning

Data cleaning is always necessary in variant calling results as most variant callers might make duplicated calls on the same position. Duplicated variant calls usually have different variant type decisions, e.g. one SNP and one INDEL. This project removes all duplicated variant calls on the same position from Query VCF file to avoid any ambiguity. Here is an example of duplicated variant calls for same position, the variant is called twice with different variant types (INDEL and SNP) at position 57991437.

Table 3.2: Example of duplicated variant calls in Query VCF

POS	REF	ALT	FORMAT	QUERY
57991437	G	T	GT:QQ:BD:BK:BI:BVT:BLT	1/0:40.6:UNK::tv:SNP:het
57991437	G	GTTT,GTTT	GT:QQ:BD:BK:BI:BVT:BLT	2/1:40.6:UNK::i1_5:INDEL:hetalt

3.5 Compare Illumina/PacBio Query VCFs with Truth VCF

The Truth VCF used in experiment 1 is from National Institute of Standards and Technology (NIST), as part of the Genome in a Bottle project, it comes with BED files which splits the whole genome into confident regions and non-confident regions. As shown in Figure 3.3, yellow circle stands for Truth VCF, green and red circle stands for Query VCFs from Illumina and PacBio reads. The areas labelled with 1, 3, 5 above the vertical line stand for confident regions and areas labelled with 2, 4 and 6 stand for non-confident regions.

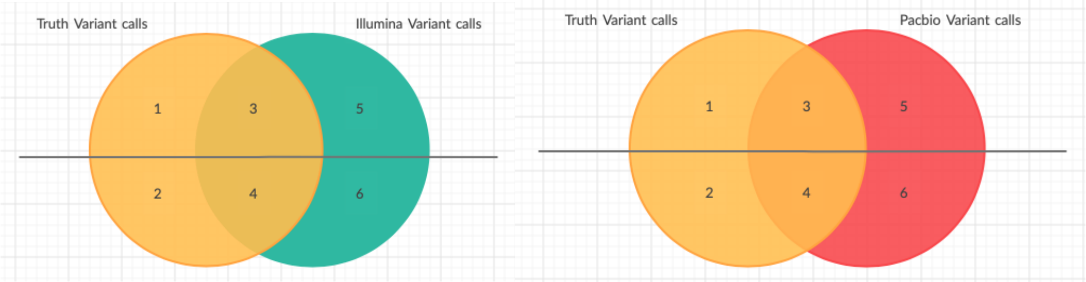


Figure 3.3 Compare Truth VCF with Query VCFs from Illumina and PacBio respectively

Category 3 is called by both Truth VCF and Query VCF, which is classified as True Positives (TP), category 1 is called only by Truth VCF, which is False Negatives (FN), category 5 called only in Query VCF, which is False Positives (FP). Categories 2, 4 and 6 are not assessed, therefore Unknown.

Table 3.3 Categories and their classifications

Category	Classification	Confident regions
1	FN	Yes
2	Unknown	No
3	TP	Yes
4	Unknown	No
5	FP	Yes
6	Unknown	No

With Truth VCF, we can classify all variant calls from either Illumina or PacBio reads into 6 categories. We are able to calculate Recall, Precision and F1 score from them and show the performance of variant calling. Formula for Recall = TP/(TP+FN), Precision = TP/ (TP+FP), F1 score =2 * Precision *Recall / (Precision + Recall).

All three VCFs shall also be compared together to show the intersections between each other. For experiment 1, since there’s BED file (used to indicate confident regions) from Truth VCF, only variant calls from confident regions are involved. For experiment 2 and 3, all variants calls are included.

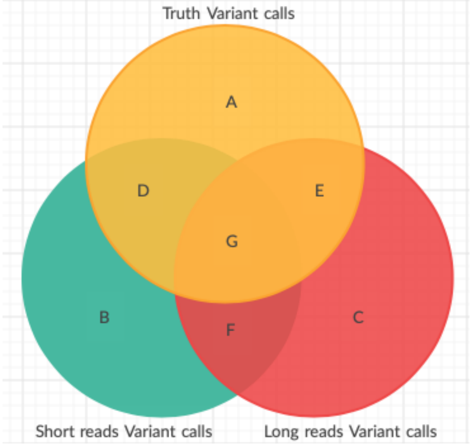


Figure 3.4 Comparison between Illumina and PacBio’s Query VCFs with presence of Truth VCF

As shown in Figure 3.4, region A is the union of False Negative (FN) from both Query VCFs. Region G is intersection of True Positive (TP) from both Query VCFs. Region D is standalone TP of Illumina's Query VCF, and region E is standalone TP of PacBio's Query VCF. Region F is intersection of False Positive (FP) from both Query VCFs. Region B is standalone FP of Illumina's Query VCF, and region C is standalone FP of PacBio's Query VCF.

Table 3.4 Different regions and their classifications

Region	Classification
A	FN
B	FP
C	FP
D	TP
E	TP
F	FP
G	TP

3.6 Annotation and Classification

With presence of Truth VCF, it is easy to analyze Query VCFs and classify variant calls into FN, TP and FP. While when DeepVariant is applied to an unknown genome, without Truth VCF, analyzing Query VCFs will be difficult. When comparing two Query VCFs from Illumina and PacBio, it will end up in Figure 3.5.

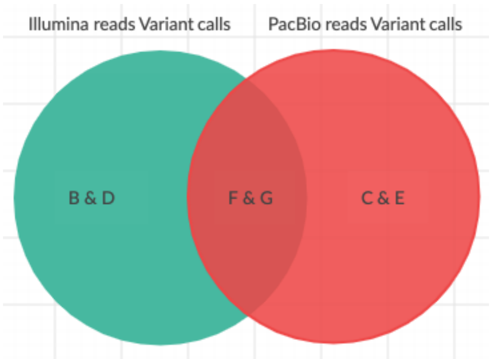


Figure 3.5 Comparison between Illumina and PacBio Query VCFs without presence of Truth VCF

Since region B and D, F and G, C and E are all fixed together, we can no longer differentiate between TP and FP, therefore can't find out correct variant calls. An algorithm must be developed to differentiate between B and D, F and G as well as C and E.

Annotation information (repeated sequence and segmental duplications) is introduced into the analysis of each region to see if there's any tendency between B and D, F and G as well as C and E. Besides, other variant information like SNP/INDEL ratio, genotype quality, read depth and variant allele fraction is also used as indicators to classify TP and FP.

Chapter 4

Results and Discussions

4.1 Experiment 1: Human genome HG002

On the official Github page of DeepVariant [9], authors of the software have released its variant call quality metrics for Illumina and PacBio reads of human genome HG002 (Table 4.1). Since HG002 reads are publicly available, it's a good chance to run DeepVariant on the same dataset to check its repeatability and replicability and its performance on human genome.

Table 4.1: DeepVariant official variant call quality for Illumina reads (HG002 chromosome 1-22) and PacBio reads (HG002 chromosome 20)

Type	# FN	#FP	Recall	Precision	F1_Score
INDEL	1488	944	0.996798	0.998048	0.997423
SNP	1576	725	0.999483	0.999762	0.999623

Type	#TP	#FN	#FP	Recall	Precision	F1_Score
INDEL	9992	180	156	0.982304	0.985202	0.983751
SNP	65167	75	97	0.998850	0.998515	0.998683

The results from experiment 1 will be compared with those in Table 4.1. After that, variant calls on chromosome 20 from both Illumina and PacBio reads are then extracted and analyzed together. The reason why we can't evaluate variant calls on chromosome 1-22 from PacBio reads is because PACBIO model of DeepVariant is trained on HG002 chromosome 1-19 (explained in Chapter 2.3).

The reads, reference and VCF files used in experiment 1 are list here (original sources in Appendix 5):

- BAM file: HG002_NIST_150bp_50x.bam (Illumina reads)/pacbio.8M.30x.bam (PacBio reads)
- Reference file: hs37d5.fa.gz
- Truth VCF: HG002_GRCh37_GIAB_highconf_CG-IllFB-IllGATKHC-Ion-10X-SOLID_CHROM1-22_v.3.3.2_highconf_triophased.vcf
- Truth BED: HG002_GRCh37_GIAB_highconf_CG-IllFB-IllGATKHC-Ion-10X-SOLID_CHROM1-22_v.3.3.2_highconf_noinconsistent.bed

After running experiment 1, the original results from DeepVariant and Hap.py are listed in Table 4.2, they are just for reference, still need to be cleaned before further analysis. They exactly match with those from Table 4, which means DeepVariant is very good in repeatability and is completely replicable.

Table 4.2: Experiment 1 original results: variant call performance for Illumina reads (HG002 chromosome 1-22), Illumina reads (HG002 chromosome 20) and PacBio reads (HG002 chromosome 20)

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
INDEL	464764	463276	1488	913836	944	430280	742	0.996798	0.998048	0.997423
SNP	3047837	3046261	1576	3739925	725	691146	122	0.999483	0.999762	0.999623

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
INDEL	10172	10119	53	19637	27	9088	25	0.9948	0.9974	0.9961
SNP	65242	65201	41	78507	21	13244	2	0.9994	0.9997	0.9995
OVERALL	75414	75320	94	98144	48	22332	27	0.9988	0.9994	0.9991

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
INDEL	10172	9992	180	19941	156	9399	116	0.9823	0.9852	0.9838
SNP	65242	65167	75	80493	97	15156	15	0.9989	0.9985	0.9987
OVERALL	75414	75159	255	100434	253	24555	131	0.9966	0.9966	0.9966

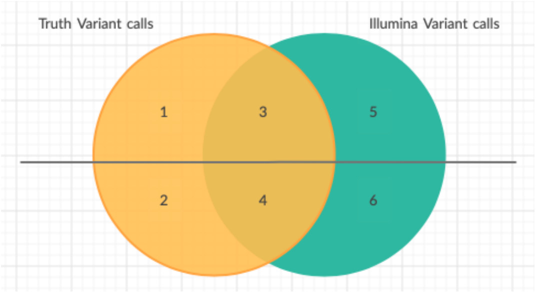
4.11 Cleaning of variant calls

In Illumina Variant calls, there are 148 duplicated variant calls (variant calls with same positions but different variant types (SNP or INDEL)), in PacBio Variant calls, there are 195 duplicated variants calls. Since they have different called variant types in exactly the same positions, they are all removed from Truth, Illumina and PacBio VCFs. After that, there are 77454 variant calls left in Truth VCF, 97979 variant calls in Illumina Query VCF and 100389 variant calls in PacBio Query VCF.

Table 4.3: VCFs and number of variant calls after cleaning

VCF	Number of variant calls
Truth VCF	77454
Illumina Query VCF	97979
PacBio Query VCF	100389

4.12 Compare Illumina VCF with Truth VCF



There are supposed to be 6 categories by comparing Illumina VCF and Truth VCF following the Venn diagram above, while totally 9 categories are discovered (Table 4.4). We need to figure out what are the three extra categories 7,8 and 9.

Table 4.4 Illumina VCF categories and their attributes

Categories	Confident region	Truth variant type	Illumina variant type	Truth decision for call	Illumina decision for call	Number of variant calls
1	Yes	het/hetalt/homalt	not called	FN	.	67
2	No	het/hetalt/homalt	not called	Unknown	Unknown	107

3	Yes	het/hetalt/homalt	het/hetalt/homalt	TP	TP	75235
4	No	het/hetalt/homalt	het/hetalt/homalt	Unknown	Unknown	1995
5	Yes	not called	het	.	FP	21
6	No	not called	het/hetalt/homalt	Unknown	Unknown	20062
7	Yes	het/hetalt/homalt	het/hetalt/homalt	FN	FP	25
8	Yes	not called	het/homalt	.	TP	442
9	Yes	het/homalt	not called	TP	.	25
SUM	NA	NA	NA	NA	NA	97979

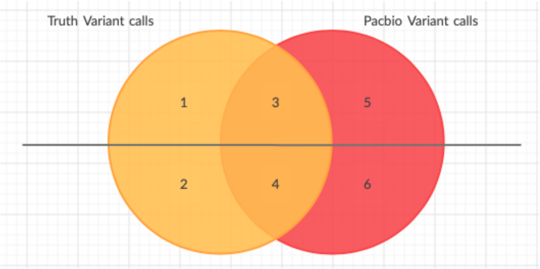
Following the concept of confident regions, any variant call in non-confident region, no matter it's in Truth or Illumina Query VCF, its decision is unknown. The extra category 7,8 and 9 are all in confident regions. Number of variant calls in all 9 categories add up to 97979 (Illumina Query VCF size), number of variant calls from category 1,2,3,4,8,9 add up to 77454 (Truth VCF size), which means our analysis doesn't misclassify any variant call. It seems category 7 cannot be mapped into Venn diagram, it is because DeepVariant made a wrong call that its truth variant type ≠ Illumina variant type (genotype mismatch). Category 7 is double counted as FN and FP. Category 8 and 9 are counted into TP because they are true variant calls with a different variant representation from Truth VCF (described in Chapter 3.3).

When calculating the recall and precision of calling performance, we only include categories in confident regions (category 1,3,5 and 7,8,9). The total number of variant calls whose decision is FN is 67+25=92, whose decision is TP is 75235+442+25=75702, whose decision is FP is 21+25=46. Therefore Precision = 75702/ (75702+46) =0.999393, Recall = 75702/ (75702+92) = 0.998786, F1 score = 2*0.999393*0.998786/ (0.999393+0.998786) = 0.999089.

Table 4.5 Variant calling performance for Illumina reads in experiment 1

Type	TRUTH.TP	TRUTH.FN	QUERY.FP	Recall	Precision	F1
OVERALL	75702	92	46	0.998786	0.999393	0.999089

4.13 Compare PacBio VCF with Truth VCF



The same Venn diagram analysis is applied to comparison between PacBio Query VCF and Truth VCF. Still 9 categories are found, and number of variant calls in all 9 categories add up to 100389 (PacBio VCF size), number of variant calls from category 1,2,3,4,7,9 add up to 77454 (Truth VCF size).

Table 4.6 PacBio VCF categories and their attributes

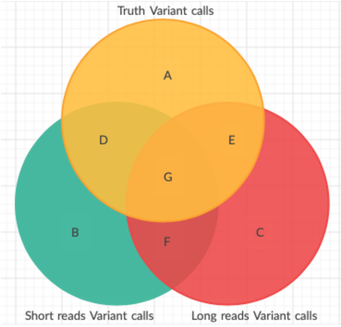
Categories	Confident region	Truth variant type	PacBio variant type	Truth decision for call	PacBio decision for call	Number of variant calls
1	Yes	het/hetalt/homalt	not called	FN	.	121
2	No	het/hetalt/homalt	not called	Unknown	Unknown	169
3	Yes	het/hetalt/homalt	het/hetalt/homalt	TP	TP	75067
4	No	het/hetalt/homalt	het/hetalt/homalt	Unknown	Unknown	1933
5	Yes	not called	het	.	FP	117
6	No	not called	het/hetalt/homalt	Unknown	Unknown	22334
7	Yes	het/hetalt/homalt	het/hetalt/homalt	FN	FP	116
8	Yes	not called	het/homalt	.	TP	484
9	Yes	het/homalt	not called	TP	.	48
Sum	NA	NA	NA	NA	NA	100389

When calculating the recall and precision of calling performance, we only include categories in confident regions (category 1,3,5 and 7,8,9). The total number of variant calls whose decision is FN is 237, whose decision is TP is 75599, whose decision is FP is 46. Therefore Precision = 75599/ (75599+233) = 0.996927, Recall = 75599 / (75599+237) = 0.996875, F1 score = 2 * 0.996927 * 0.996875 / (0.996927+0.996875) = 0.996901.

Table 4.7 Variant calling performance for PacBio reads in experiment 1

Type	TRUTH.TP	TRUTH.FN	QUERY.FP	Recall	Precision	F1
OVERALL	75599	237	233	0.996875	0.996927	0.996901

4.14 Compare Illumina, PacBio VCFs with Truth VCF

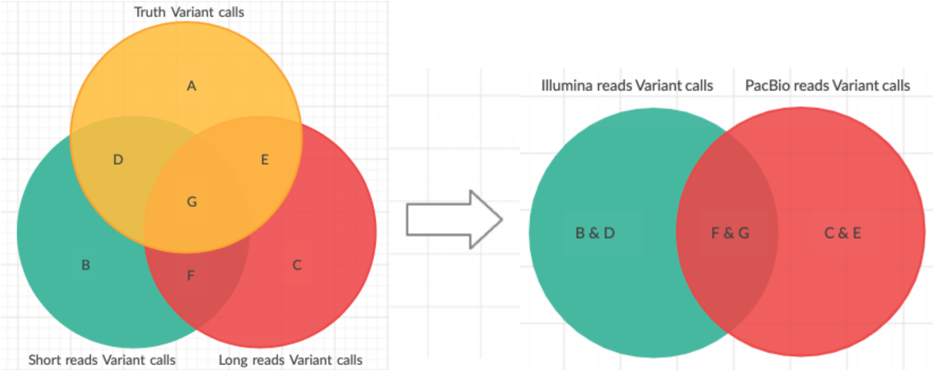


This comparison still only involves confident regions' variant calls. Number of variant calls in each region is listed in Table 4.8, they are used for further analysis with annotation information.

Table 4.8 Number of variant calls in regions for experiment 1

Region	Number of variant calls	Classification
A	308	FN
B	29	FP
C	216	FP
D	244	TP
E	141	TP
F	17	FP
G	75458	TP

4.15 Analysis with annotation information


Truth Variant calls
A
D
E
G
B
F
C
Short reads Variant calls
Long reads Variant calls
Illumina reads Variant calls
PacBio reads Variant calls
B & D
F & G
C & E

Without presence of Truth VCF, we need to differentiate between region B and D, F and G as well as C and E in order to obtain as many True Positives variant calls. Therefore, the characteristics of these regions are studied to see if there's any tendency that can be used to classify them. We believe variant calls in different regions shall have different repeats/segmental duplications proportions. While there's no particular tendency found (Table 4.9), and we suspect that it is because we only have variant calls in confident regions, which causes bias.

Table 4.9: Number of variant calls in repeats and segmental duplications in experiment 1

Region	Number of variant calls	No. of variant calls in repeats	Percentage of repeats (%)
A	308	240	77.92
B	29	24	82.76
C	216	180	83.33
D	244	190	77.87
E	141	126	89.36
F	17	15	88.24
G	75458	41985	55.64

Region	Number of variant calls	No. of variant calls in segmental duplications	Percentage of segmental duplications (%)
A	308	4	1.29
B	29	0	0
C	216	2	0.93
D	244	2	0.82
E	141	1	0.71
F	17	0	0
G	75458	480	0.64

4.16 Summary for experiment 1

Due to the exact match of variant calls from experiment 1 with officially published results, we can trust the performance of DeepVariant in the following experiments. Also, to make sure the results are precisely correct, each experiment in this project is run at least twice. I will not proceed to analysis unless I get two copies of same results.

F1 score is used as overall performance measure as it combines Recall and Precision. In experiment 1, F1 score of variant calls from Illumina reads is 0.999089, and F1 score from PacBio reads is 0.996901. But this is only for confident regions, we are not able to assess performance over the whole genome regions yet because we are using Truth VCF from Genome in a Bottle, which are not ground truth and are only reliable on confident regions.

Also, analysis with annotation information is not accurate within confident regions only. Therefore, experiment 1 only proves DeepVariant's good repeatability and accuracy on confident regions, so we decide to simulate reads for artificial genome to get ground truth VCF and run more extensive experiments.

4.2 Experiment 2: Artificial genome with SNP

The initial purpose of experiment 2 is to test DeepVariant's performance over the whole region (not just confident regions) regarding SNP variants only. To reduce the experiment time and be consistent with analysis done in experiment 1, we are still analyzing variants on chromosome 20.

For experiment 2, hs37d5 is still used as reference genome, artificial genome is generated by adding SNP variants into reference genome. Around 0.1% of positions in chromosome 20 is randomly assigned with SNP variants (59422 variants out of 63025520 positions), so ground truth VCF contains 59422 SNPs.

After that, both Illumina and PacBio reads are generated and aligned. To be consistent with input reads used in experiment 1, Illumina reads has length 150bp with 50 coverage, PacBio reads has 30 coverage and are circular consensus sequencing (CCS). When reference genome, aligned reads and ground truth VCF are ready, they are all sorted, indexed and then fed into DeepVariant. The original results from DeepVariant and Hap.py are displayed in Table 4.10.

Table 4.10: Experiment 2 original results: variant call performance for Illumina reads and PacBio reads (artificial genome chromosome 20 with SNP)

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
SNP	59422	59073	349	59081	8	0	8	0.994127	0.999865	0.996987

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
INDEL	0	0	0	4971	4971	0	0	0.00	0.00	0.00
SNP	59422	58958	464	59096	138	0	138	0.992191	0.997665	0.994921
OVERALL	59422	58958	464	64067	5109	0	138	0.992191	0.920255	0.954870

In experiment 2, Illumina and PacBio VCFs don't have any duplicated variant calls, so we can directly use original results for analysis without data cleaning. As we can see from Table 4.10, variant calls from Illumina has F1 score of 0.996987, which maintains high standard. While F1 score for PacBio reads is only 0.954870 due to the 4971 False Positives INDEL variant calls. Will DeepVariant also make large amounts of FP INDEL calls even with reads that contain both SNPs and INDELs (real world genome always has both)? This concern triggers experiment 3.

Also, we are wondering, will increase coverage of reads improve variant calling performance? The performance of PacBio reads from experiment 1 is higher than experiment 2, is it due to confident regions? To answer these two questions, two sub-experiments 2-1 and 2-2 are carried out.

4.21 Compare Illumina, PacBio VCF with Truth VCF

Table 4.11: Number of variant calls in regions for experiment 2

Region	Number of variant calls
A (FN)	562
B (FP)	8
C (FP)	5109
D (TP)	213
E (TP)	98
F (FP)	0
G (TP)	58860

Region B has only 8 variant calls, Region F has 0 variant calls (they are negligible), so we no longer have to differentiate between Region B and D, F and G. We only need to find if we can differentiate Region C and Region E with annotation information (Figure 4.1).

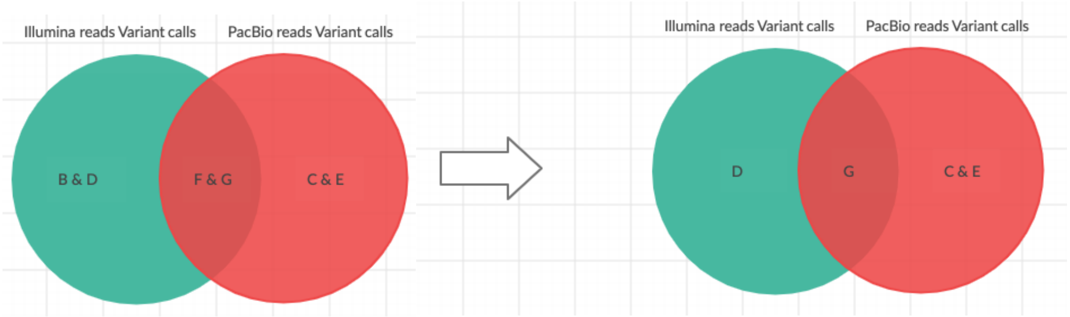

Illumina reads Variant calls
PacBio reads Variant calls
B & D
F & G
C & E
Illumina reads Variant calls
PacBio reads Variant calls
D
G
C & E

Figure 4.1: When Region B and Region F are negligible

4.22 Analysis with annotation information

Table 4.12: Number of variant calls in repeats and segmental duplications in experiment 2

Region	Number of variant calls	No. of variant calls in repeats	Percentage of repeats (%)
A (FN)	562	379	67.44
B (FP)	8	7	87.5
C (FP)	5109	3513	68.76
D (TP)	213	131	61.50
E (TP)	98	83	84.69
F (FP)	0	0	0
G (TP)	58860	29678	50.42

Region	Number of variant calls	No. of variant calls in segmental duplications	Percentage of segmental duplications (%)
A (FN)	562	259	46.09
B (FP)	8	1	12.5
C (FP)	5109	389	7.61
D (TP)	213	53	24.88
E (TP)	98	17	17.35
F (FP)	0	0	0
G (TP)	58860	1571	2.67

From Table 4.12, we can see percentage of repeats/segmental duplications is higher in E than in C, we will keep investigating this in experiment 3 to see if this characteristic maintains the same. If yes, they can be used to classify region C and region E variant calls.

4.23 Analysis with variant information

There is also variant information that might be useful to separate region C and E, like SNP/INDEL ratio, genotype quality, read depth and variant allele fraction. When calculating the SNP/INDEL ratio in experiment 2, number of INDELs in region C is 4971 (97.30%), number of SNPs is 138 (2.7%). And all variant calls in region E are SNPs, which is a good starting point for classifying region C and E (Table 4.13). We only need to find a method to differentiate between Region C SNP and Region E SNP, all INDEL False Positives are filtered out from here.

Table 4.13 SNP/INDEL ratio for Region C and E

	Total number of variant calls	Number of INDEL	Number of SNP	SNP percentage (%)
Region C (FP)	5109	4971	138	2.7
Region E (TP)	98	0	98	100

We extract SNPs from region C and region E, and analyze their genotype quality (Figure 4.2), read depth (Figure 4.3) and variant allele fraction (Figure 4.4) distributions. They all have slightly different distributions.

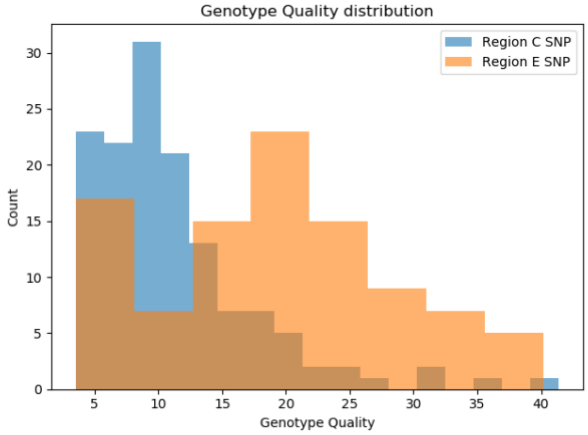


Figure 4.2: Genotype quality distribution for Region C SNP and Region E SNP

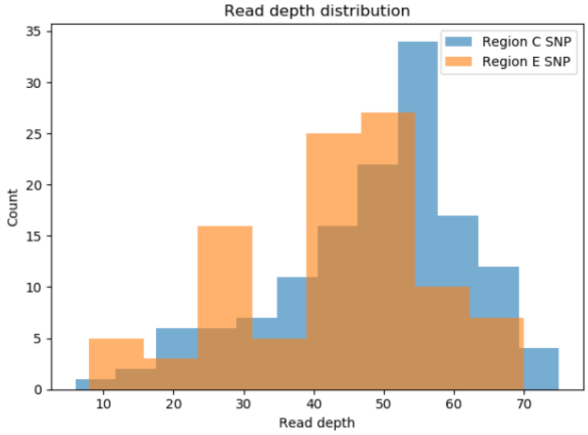


Figure 4.3: Read depth distribution for Region C SNP and Region E SNP

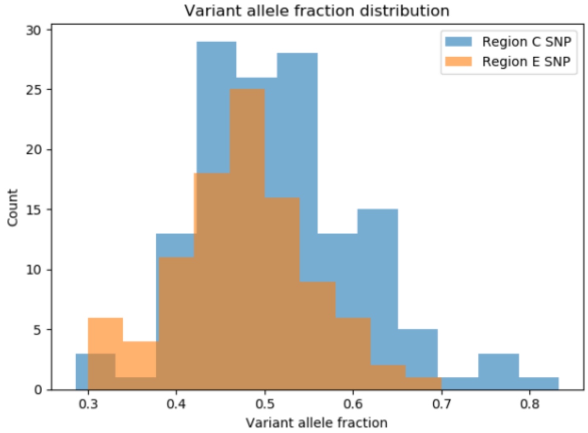


Figure 4.4: Variant allele fraction distribution for Region C SNP and Region E SNP

4.24 Sub-experiments 2-1, 2-2: Reads coverage, confident regions and variant calls performance

Sub-experiment 2-1 runs PACBIO model of DeepVariant with a different reads coverage (100), number of False Positives INDEL calls decreases from 4971 (coverage 30) to 2549 (coverage 100), so F1 score increases from 0.9549 (coverage 30) to 0.9737 (coverage 100). When the variant calls are further evaluated on confident regions only, number of False Positives INDEL calls drops to 1212, F1 score increases to 0.9861.

Table 4.14: Variant call performance for PacBio reads with coverage 100, sub-experiment 2-1

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
INDEL	0	0	0	2549	2549	0	0	0.00	0.00	0.00
SNP	59422	58940	482	59088	148	0	148	0.9919	0.9975	0.9947
OVERALL	59422	58940	482	53537	2697	0	148	0.9919	0.9562	0.9737

Table 4.15: Variant call performance for PacBio reads with coverage 100 on confident regions only, sub-experiment 2-2

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
INDEL	0	0	0	2549	1212	1337	0	0.00	0.00	0.00
SNP	53181	52977	204	59088	72	6039	72	0.9962	0.9986	0.9974
OVERALL	53181	52977	204	53537	1284	7376	72	0.9962	0.9763	0.9861

4.25 Summary for experiment 2

F1 score of Illumina reads maintains high standard (0.996987) comparing to that from experiment 1 (0.999089). While F1 score of PacBio reads shows a dramatic drop from 0.996901 (experiment 1) to 0.954870, which is due to the calling of large amounts of False Positive INDELs.

When analyzing VCFs, region B and F contain few variant calls that they can be neglected (Illumina Query VCF has few False Positives), we only need to find a way to classify Region C (FP) and Region E (TP).

When further analyzing annotation and variant information for Region C and E, Region C contains mostly INDEL calls and Region E contains all SNP calls. They also have different characteristics regarding repeats/segmental duplications, genotype quality, read depth and variant allele fraction.

From results of sub-experiments 2-1 and 2-2, increasing reads coverage and evaluating on confident regions only will both increase F1 score for PACBIO model of DeepVariant.

4.3 Experiment 3: Artificial genome with SNP and INDEL

The process of running experiment 3 is similar to experiment 2, but artificial genome will be generated by adding both SNP and INDEL variants into reference genome. The number of SNPs added is 79250, number of insertions is 9975 and number of deletions is 9971 (ratio 8:1:1). Both insertions and deletions are single base pair INDELs.

Table 4.16: Experiment 3 original results: variant call performance for Illumina reads and PacBio reads (artificial genome chromosome 20 with SNP and INDEL)

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
INDEL	19946	19855	91	19841	0	0	0	0.995438	1.0000	0.997714
SNP	79250	78911	339	78926	1	0	0	0.995722	0.99987	0.997850
OVERALL	99196	98766	430	98767	1	0	0	0.995665	0.999990	0.997823

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
INDEL	19946	19748	198	39373	19639	0	134	0.990073	0.501206	0.665510
SNP	79250	79022	228	79207	171	0	9	0.997123	0.997841	0.997482
OVERALL	99196	98770	426	118580	19810	0	143	0.995705	0.832940	0.907078

From Table 4.16, experiment 3's F1 score for Illumina reads (0.997823) is still good referring to experiment 1 (0.999089) and 2 (0.996987). While F1 score for PacBio reads is even worse (0.907078) than experiment 2 (0.954870). There are 19639 False Positive INDEL calls, which is the same as experiment 2.

4.31 Compare Illumina, PacBio VCF with Truth VCF

Data cleaning is required in experiment 3 as there are duplicated variant calls, in Illumina VCF, there are 28 duplicated variant calls (variant calls with same positions but different variant types (SNP or INDEL)), in PacBio VCF, there are 92 duplicated variants calls, totally 93 variant calls are removed from all three VCFs. After cleaning, number of variant calls are displayed in Table 4.17.

Table 4.17: Number of variant calls in regions for experiment 3

Region	Total variant calls
A (FN)	505
B (FP)	0
C (FP)	19686
D (TP)	86
E (TP)	154
F (FP)	0
G (TP)	98609

Same as experiment 2, variant calls in Region B and F can be neglected (there's none in experiment 3).

4.32 Analysis with annotation information

Also, percentage of repeats/segmental duplications is higher in Region E than in Region C (same as experiment 2).

Table 4.18: Number of variant calls in repeats and segmental duplications in experiment 3

Region	Number of variant calls	No. of variant calls in repeats	Percentage of repeats (%)

A (FN)	505	338	66.93
B (FP)	0	0	0
C (FP)	19686	11029	56.02
D (TP)	86	55	63.95
E (TP)	154	126	81.82
F (FP)	0	0	0
G (TP)	98609	48874	49.56

Region	Number of variant calls	No. of variant calls in segmental duplications	Percentage of segmental duplications (%)
A (FN)	505	265	52.48
B (FP)	0	0	0
C (FP)	19686	944	4.80
D (TP)	86	13	15.12
E (TP)	154	31	20.13
F (FP)	0	0	0
G (TP)	98609	2874	2.91

4.33 Analysis with variant information

Table 4.19 SNP/INDEL ratio for Region C and E

	Total number of variant calls	Number of INDEL	Number of SNP	SNP percentage (%)
Region C (FP)	19686	19574	112	0.57
Region E (TP)	154	39	126	81.82

Since there are 19574 INDEL in Region C and only 39 INDEL in Region E, it will be very difficult to differentiate between them. We will use the same strategy in experiment 2, only classify Region C SNP and Region E SNP.

The distributions of genotype quality (Figure 4.5), read depth (Figure 4.6) and variant allele fraction (Figure 4.7) for Region C SNP and Region E SNP are analyzed. Still, they all have different distributions.

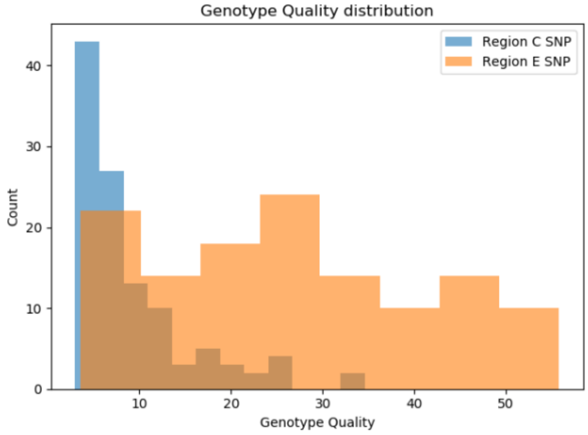


Figure 4.5: Genotype quality distribution for Region C SNP and Region E SNP

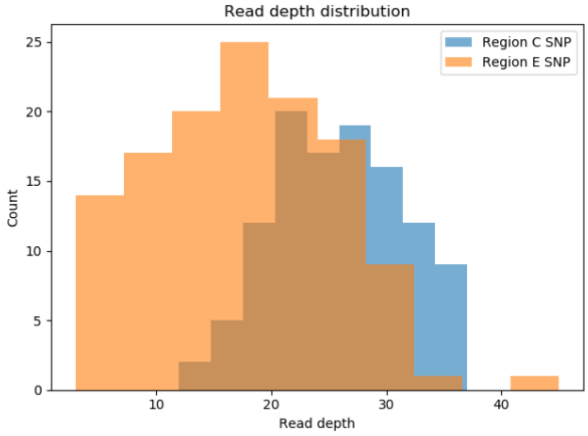


Figure 4.6: Read depth distribution for Region C SNP and Region E SNP

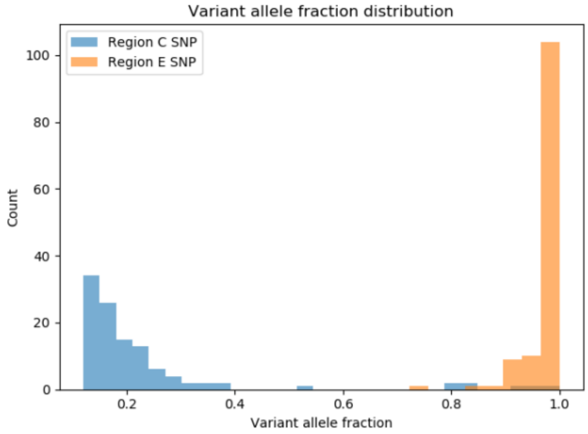


Figure 4.7: Variant allele fraction distribution for Region C SNP and Region E SNP

4.34 Classifier

To increase the accuracy of classification for Region C SNP and Region E SNP, all annotation and variant information mentioned above is used in construction of classifier.

Four different classifiers are built including random forest, support vector machine (SVM), multi-layer perceptron (MLP) and adaboost. Regarding training data attributes for classifiers, if a variant call is in repeats/segmental duplication, it has value 1, otherwise 0. If a variant call is from Region C, its predication value is 0, otherwise 1. All other attribute values (genotype quality, read depth, variant allele fraction) are normalized between 0 and 1 to avoid any weighting bias. The four classifiers are trained with 70% of data and tested on the other 30%. Classifiers are run ten times each, here are the accuracy rates (Table 4.20).

Table 4.20: Classifiers and accuracy rates (%)

Classifier	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	Trial 8	Trial 9	Trial 10	Average
Random forest	98.61	100	98.61	98.61	98.61	97.22	97.22	94.44	94.44	95.83	97.36
SVM	97.22	100	98.61	95.83	97.22	94.44	94.44	95.83	94.44	94.44	96.25
MLP	95.83	97.22	97.22	90.28	95.83	95.83	91.67	88.88	93.06	95.83	94.17
Adaboost	94.44	97.22	97.22	94.44	97.22	94.44	95.83	90.27	94.44	95.83	95.14

Among 4 classifiers, random forest maintains the highest average accuracy rate and is very stable. It has average accuracy rate as 97.359%, which means, 126 * 97.359%= 123 Region C SNP variant calls can be added into True Positives, 112 * 97.359%= 109 Region E SNP plus 19574 Region E INDEL variant calls can be filter out as False Positives.

4.35 Summary of experiment 3

Experiment 3 simulates the artificial genome with both SNP and INDEL which is very like real-world genome, its performance should be most reliable among first 3 experiments. Experiment 3 verifies the assumptions made in experiment 2 including poor variant call performance on PacBio reads, negligible Region B and F, difference of annotation & variant information on Region C and E. With the help of random forest classifier, 97.359% of True Positives variant calls can be extracted and 97.359% of False Positives can be filtered out.

4.4 Experiment 4: Empirical sample reads from NCIG

After first three experiments, we already have a good understanding on the performance of DeepVariant, and we have a classifier to help to increase number of True Positives variant calls. But how about its performance on the specific reads used in National Centre for Indigenous Genomics (NICG)?

Training with one type of reads, can DeepVariant also perform well with other types of reads? In experiment 4, we wish to test robustness of DeepVariant.

NCIG are mostly using Illumina reads and small amount of PacBio reads. Its Illumina reads has 150bp length with 40 coverage, mean size for DNA fragments is 350bp (standard deviation is 20bp). PacBio reads are all continuous long reads (CLR). Both reads are simulated with SNP and INDEL variants, 19586 INDELs and 79499 SNPs are added into reference genome.

Unfortunately, due to time constraints and instability of server, testing on PacBio reads is still working in progress. Here only shows results for Illumina reads.

Table 4.21: Experiment 4 original results: variant call performance for Illumina reads

Type	TRUTH.TOTAL	TRUTH.TP	TRUTH.FN	QUERY.TOTAL	QUERY.FP	QUERY.UNK	FP.gt	Recall	Precision	F1
INDEL	19586	19552	34	19542	2	0	2	0.998264	0.999898	0.999080
SNP	79499	79324	175	79342	5	0	3	0.997799	0.999937	0.998967
OVERALL	99085	98876	209	98884	7	0	5	0.997891	0.999929	0.998909

F1 score on NCIG Illumina reads is still high (0.998909), there are only 209 False Negatives and 7 False Positives variant calls, which maintains same high performance as in previous experiments (>0.995). This verifies DeepVariant's robustness on different types of Illumina reads to some extent.

Chapter 5

Conclusion and Future Work

In this project, I performed a comprehensive benchmarking of DeepVariant performance using both Illumina and PacBio reads. In general, DeepVariant achieves much higher accuracy in all short-read datasets (e.g., maintaining >0.995 F1 score) compared to relatively poor accuracy in long-read datasets (e.g., ~0.9 score). When the reads coverage decreases, DeepVariant's performance gap between short-read datasets and long-read dataset gets wider. Moreover, DeepVariant has a poor accuracy when evaluating on the whole genome instead of "confident regions" only.

Even DeepVariant shows poor performance on PacBio reads, but if we have separate calling results from both short-reads and long-reads of the same individual, is it possible to use genomic feature information to make an improved calling result? I studied both genome annotation information and variant-associated information and assessed their effectiveness to distinguish True Positives (TP) and False Positives (FP) in variants called by long reads. I found that repeat/segmental duplication annotation, genotype quality, read depth and variant allele fraction have the potential to distinguish true positives and false positives.

I applied four different classifiers, including random forest, support vector machine, multi-layer perceptron and adaboost, to incorporate the above genomic features to perform TP/FP classification. The average accuracy rages from 94%-97% for different classifier, which can be used to expand the TP variants and thus improve the variant calling.

Due to the short project period (July 2019 – Oct 2019) and limited computational resources, more experiments on different types of reads shall be performed and the training of DeepVariant shall be carried out with respect to other types of reads. Moreover, the accuracy of TP/FP classifier may be further improved with more training data from different types of reads in future experiments.

Appendices

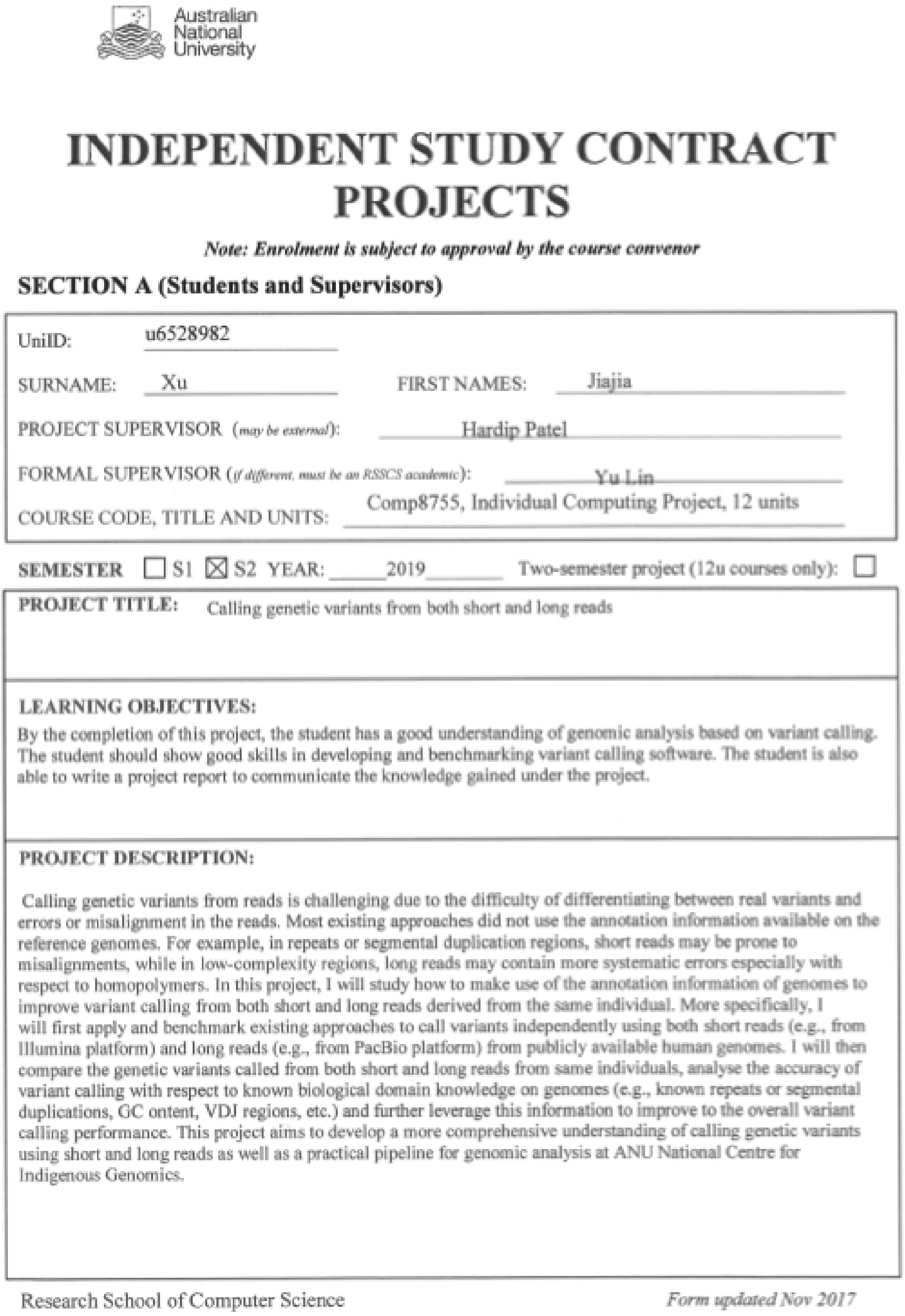
Appendix 1

Final Project Description

"Calling genetic variants from reads is challenging due to the difficulty of differentiating between real variants and errors or misalignment in the reads. Most existing approaches did not use the annotation information available on the reference genomes. For example, in repeats or segmental duplication regions, short reads may be prone to misalignments, while in low-complexity regions, long reads may contain more systematic errors especially with respect to homopolymers. In this project, I will study how to make use of the annotation information of genomes to improve variant calling from both short and long reads derived from the same individual. More specifically, I will first apply and benchmark existing approaches to call variants independently using both short reads (e.g., from Illumina platform) and long reads (e.g., from PacBio platform) from publicly available human genomes. I will then compare the genetic variants called from both short and long reads from same individuals, analyse the accuracy of variant calling with respect to known biological domain knowledge on genomes (e.g., known repeats or segmental duplications, GC ontent, VDJ regions, etc.) and further leverage this information to improve to the overall variant calling performance. This project aims to develop a more comprehensive understanding of calling genetic variants using short and long reads as well as a practical pipeline for genomic analysis at ANU National Centre for Indigenous Genomics."

Appendix 2

Independent Study Contract



Australian National University

INDEPENDENT STUDY CONTRACT PROJECTS

Note: Enrolment is subject to approval by the course convenor

SECTION A (Students and Supervisors)

UniID: u6528982

SURNAME: Xu FIRST NAMES: Jiajia

PROJECT SUPERVISOR (*may be external*): Hardip Patel

FORMAL SUPERVISOR (*if different, must be an RSSCS academic*): Yu Lin

COURSE CODE, TITLE AND UNITS: Comp8755, Individual Computing Project, 12 units

SEMESTER □ S1 ☒ S2 YEAR: 2019 Two-semester project (12u courses only): □

PROJECT TITLE: Calling genetic variants from both short and long reads

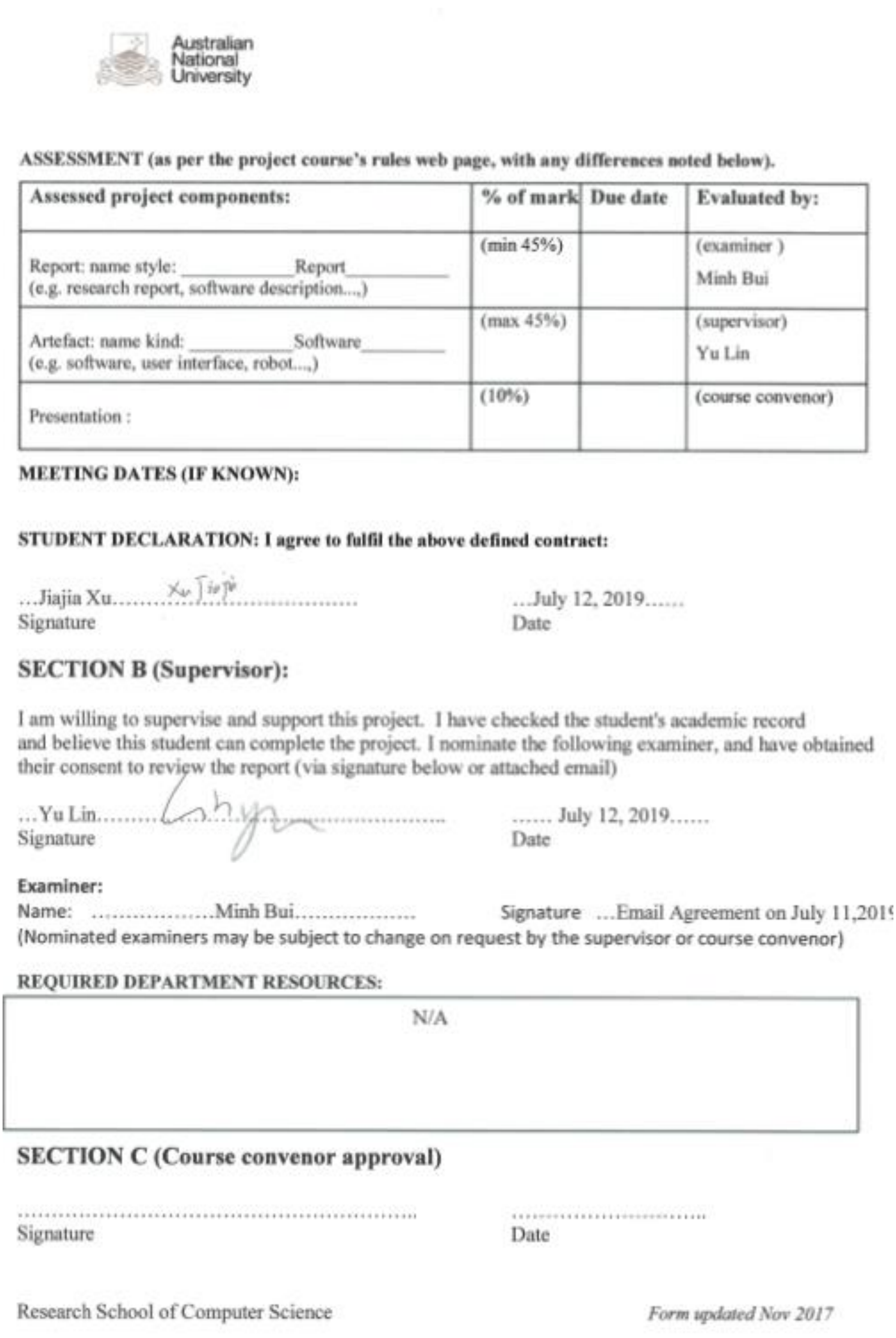
LEARNING OBJECTIVES:

By the completion of this project, the student has a good understanding of genomic analysis based on variant calling. The student should show good skills in developing and benchmarking variant calling software. The student is also able to write a project report to communicate the knowledge gained under the project.

PROJECT DESCRIPTION:

Calling genetic variants from reads is challenging due to the difficulty of differentiating between real variants and errors or misalignment in the reads. Most existing approaches did not use the annotation information available on the reference genomes. For example, in repeats or segmental duplication regions, short reads may be prone to misalignments, while in low-complexity regions, long reads may contain more systematic errors especially with respect to homopolymers. In this project, I will study how to make use of the annotation information of genomes to improve variant calling from both short and long reads derived from the same individual. More specifically, I will first apply and benchmark existing approaches to call variants independently using both short reads (e.g., from Illumina platform) and long reads (e.g., from PacBio platform) from publicly available human genomes. I will then compare the genetic variants called from both short and long reads from same individuals, analyse the accuracy of variant calling with respect to known biological domain knowledge on genomes (e.g., known repeats or segmental duplications, GC ontent, VDJ regions, etc.) and further leverage this information to improve to the overall variant calling performance. This project aims to develop a more comprehensive understanding of calling genetic variants using short and long reads as well as a practical pipeline for genomic analysis at ANU National Centre for Indigenous Genomics.

Research School of Computer Science *Form updated Nov 2017*



Australian National University

ASSESSMENT (as per the project course's rules web page, with any differences noted below).

Assessed project components:	% of mark	Due date	Evaluated by:
Report: name style: ___________Report__________ (e.g. research report, software description...,)	(min 45%)		(examiner) Minh Bui
Artefact: name kind: ___________Software________ (e.g. software, user interface, robot...,)	(max 45%)		(supervisor) Yu Lin
Presentation :	(10%)		(course convenor)

MEETING DATES (IF KNOWN):

STUDENT DECLARATION: I agree to fulfil the above defined contract:

...Jiajia Xu.......Xu Jiajia...................... ...July 12, 2019......
Signature Date

SECTION B (Supervisor):

I am willing to supervise and support this project. I have checked the student's academic record and believe this student can complete the project. I nominate the following examiner, and have obtained their consent to review the report (via signature below or attached email)

...Yu Lin.. ...... July 12, 2019......
Signature Date

Examiner:
Name: ....Minh Bui.................. Signature ...Email Agreement on July 11,201
(Nominated examiners may be subject to change on request by the supervisor or course convenor)

REQUIRED DEPARTMENT RESOURCES:

N/A

SECTION C (Course convenor approval)

....
Signature Date

Research School of Computer Science *Form updated Nov 2017*

Appendix 3

Description of software

The whole system contains following files:

- Bash scripts: 1. run DeepVariant and Hap.py with different inputs and parameters for different experiments 2. Simulate aligned reads
- Python files: 1. Generate artificial genome 2. Analysis with genome knowledge 3. Classifiers to differentiate True Positives and False Positives

In particular, for running DeepVariant and Hap.py on an Ubuntu server, with docker installed:

- run_wgs_real_docker.sh: run DeepVariant and Hap.py with Illumina reads from human genome HG002
- run_pacbio_real_docker.sh: run DeepVariant and Hap.py with PacBio reads from human genome HG002
- run_illumina_simulate_docker.sh: run DeepVariant and Hap.py with Illumina reads from artificial genome
- run_pacbio_simulate_docker.sh: run DeepVariant and Hap.py with PacBio reads from artificial genome

For simulating Illumina and PacBio aligned reads on an Ubuntu server:

- Aligned_Illumina_reads.sh: for simulating aligned Illumina reads from an artificial genome
- Aligned_PacBio_reads.sh: for simulating aligned PacBio reads from an artificial genome

For analysis with genome knowledge, using raw VCF files output from DeepVariant and Hap.py, and Truth VCF:

- Chr20_Truth_csv_real.py: Clean and Extract chromosome 20 variant calls only from Illumina, PacBio and Truth VCF. Store them into CSV for further process. For experiment 1 only.
- Chr20_Truth_csv_simulate.py: Clean and Extract chromosome 20 variant calls only from Illumina, PacBio and Truth VCF. Store them into CSV for further process. For experiments 2 and 3.
- Data_cleaning_real.py: Clean and extract all important info from Hap.py VCF file, remove duplicated variant calls. Convert Hap.py VCF files into csv files for further process. For experiment 1 only.
- Data_cleaning_simulate.py: Clean and extract all important info from Hap.py VCF file, remove duplicated variant calls. Convert Hap.py VCF files into csv files for further process. For experiments 2 and 3.
- Regions_real.py: Read CSV variant calls from Illumina, PacBio, and Truth. Divide all three files by intersections into 7 regions. For experiment 1 only.
- Regions_experiment2.py: Read CSV variant calls from Illumina, PacBio, and Truth. Divide all three files by intersections into 7 regions. Draw distribution graphs for variant information. Generate CSV for region C and E SNP. For experiment 2 only.
- Regions_experiment3.py: Read CSV variant calls from Illumina, PacBio, and Truth. Divide all three files by intersections into 7 regions. Draw distribution graphs for variant information. Generate CSV for region C and E SNP. For experiment 3 only.
- Repeats_real.py: Check proportion of repeats in each region, for experiment 1 only.
- Repeats_experiment2.py: Check proportion of repeats in each region, for experiment 2 only.
- Repeats_experiment3.py: Check proportion of repeats in each region, for experiment 3 only.

- SDs_real.py: Check proportion of segmental duplications in each region, for experiment 1 only.
- SDs_experiment2.py: Check proportion of segmental duplications in each region, for experiment 2 only.
- SDs_experiment3.py: Check proportion of segmental duplications in each region, for experiment 3 only.

For generating artificial genomes for experiments 2,3 and 4:

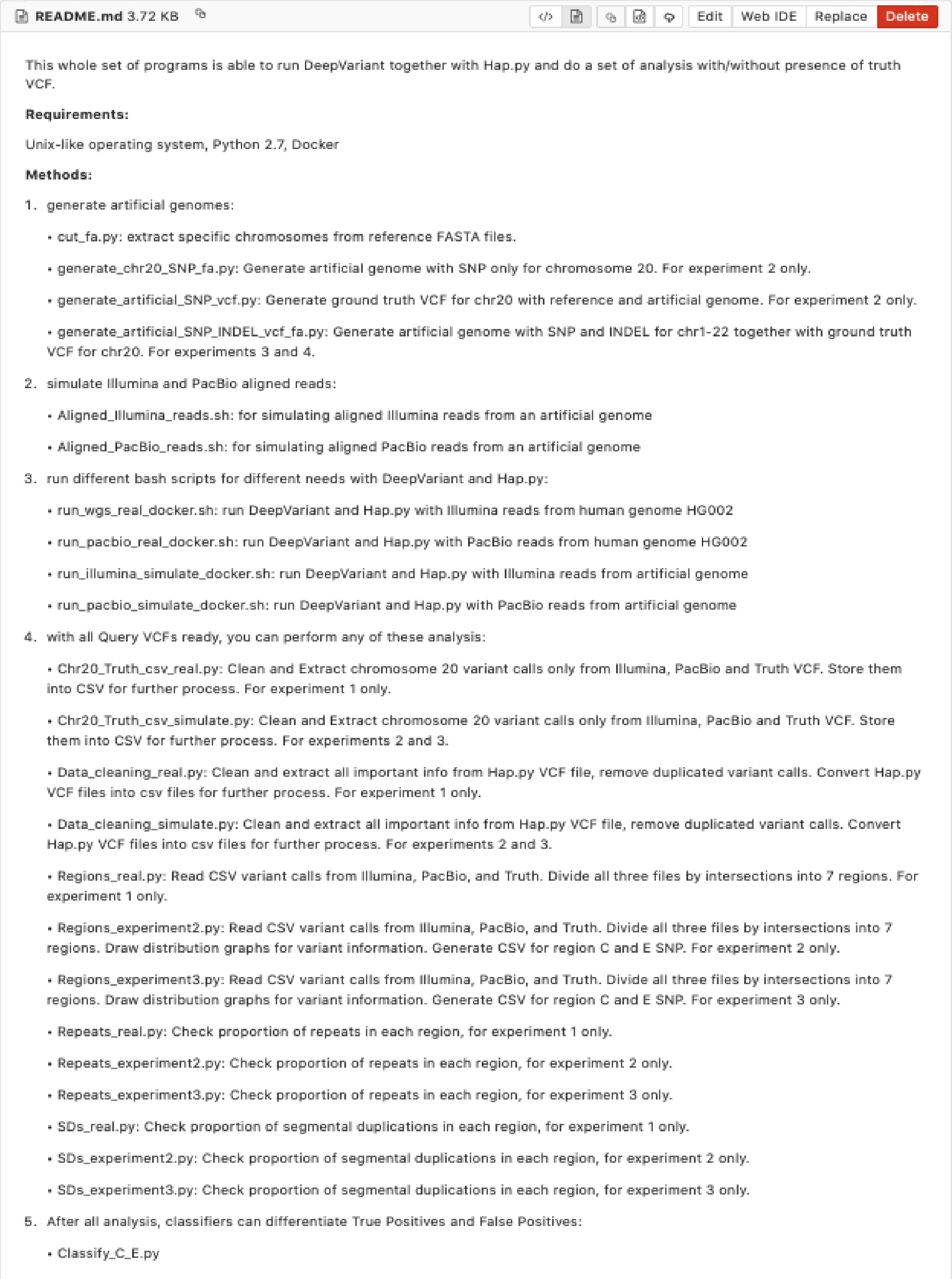
- cut_fa.py: extract specific chromosomes from reference FASTA files.
- generate_chr20_SNP_fa.py: Generate artificial genome with SNP only for chromosome 20. For experiment 2 only.
- generate_artificial_SNP_vcf.py: Generate ground truth VCF for chr20 with reference and artificial genome. For experiment 2 only.
- generate_artificial_SNP_INDEL_vcf_fa.py: Generate artificial genome with SNP and INDEL for chr1-22 together with ground truth VCF for chr20. For experiments 3 and 4.

For Classifiers to differentiate True Positives and False Positives:

- Classify_C_E.py

Appendix 4

README



This whole set of programs is able to run DeepVariant together with Hap.py and do a set of analysis with/without presence of truth VCF.

Requirements:

Unix-like operating system, Python 2.7, Docker

Methods:

1. generate artificial genomes:
   - cut_fa.py: extract specific chromosomes from reference FASTA files.
   - generate_chr20_SNP_fa.py: Generate artificial genome with SNP only for chromosome 20. For experiment 2 only.
   - generate_artificial_SNP_vcf.py: Generate ground truth VCF for chr20 with reference and artificial genome. For experiment 2 only.
   - generate_artificial_SNP_INDEL_vcf_fa.py: Generate artificial genome with SNP and INDEL for chr1-22 together with ground truth VCF for chr20. For experiments 3 and 4.
2. simulate Illumina and PacBio aligned reads:
   - Aligned_Illumina_reads.sh: for simulating aligned Illumina reads from an artificial genome
   - Aligned_PacBio_reads.sh: for simulating aligned PacBio reads from an artificial genome
3. run different bash scripts for different needs with DeepVariant and Hap.py:
   - run_wgs_real_docker.sh: run DeepVariant and Hap.py with Illumina reads from human genome HG002
   - run_pacbio_real_docker.sh: run DeepVariant and Hap.py with PacBio reads from human genome HG002
   - run_illumina_simulate_docker.sh: run DeepVariant and Hap.py with Illumina reads from artificial genome
   - run_pacbio_simulate_docker.sh: run DeepVariant and Hap.py with PacBio reads from artificial genome
4. with all Query VCFs ready, you can perform any of these analysis:
   - Chr20_Truth_csv_real.py: Clean and Extract chromosome 20 variant calls only from Illumina, PacBio and Truth VCF. Store them into CSV for further process. For experiment 1 only.
   - Chr20_Truth_csv_simulate.py: Clean and Extract chromosome 20 variant calls only from Illumina, PacBio and Truth VCF. Store them into CSV for further process. For experiments 2 and 3.
   - Data_cleaning_real.py: Clean and extract all important info from Hap.py VCF file, remove duplicated variant calls. Convert Hap.py VCF files into csv files for further process. For experiment 1 only.
   - Data_cleaning_simulate.py: Clean and extract all important info from Hap.py VCF file, remove duplicated variant calls. Convert Hap.py VCF files into csv files for further process. For experiments 2 and 3.
   - Regions_real.py: Read CSV variant calls from Illumina, PacBio, and Truth. Divide all three files by intersections into 7 regions. For experiment 1 only.
   - Regions_experiment2.py: Read CSV variant calls from Illumina, PacBio, and Truth. Divide all three files by intersections into 7 regions. Draw distribution graphs for variant information. Generate CSV for region C and E SNP. For experiment 2 only.
   - Regions_experiment3.py: Read CSV variant calls from Illumina, PacBio, and Truth. Divide all three files by intersections into 7 regions. Draw distribution graphs for variant information. Generate CSV for region C and E SNP. For experiment 3 only.
   - Repeats_real.py: Check proportion of repeats in each region, for experiment 1 only.
   - Repeats_experiment2.py: Check proportion of repeats in each region, for experiment 2 only.
   - Repeats_experiment3.py: Check proportion of repeats in each region, for experiment 3 only.
   - SDs_real.py: Check proportion of segmental duplications in each region, for experiment 1 only.
   - SDs_experiment2.py: Check proportion of segmental duplications in each region, for experiment 2 only.
   - SDs_experiment3.py: Check proportion of segmental duplications in each region, for experiment 3 only.
5. After all analysis, classifiers can differentiate True Positives and False Positives:
   - Classify_C_E.py

Appendix 5

Original sources of input reads, reference and VCF files for experiment 1

- BAM file: HG002_NIST_150bp_50x.bam.

The original FASTQ file comes from the PrecisionFDA Truth Challenge. We ran it through PrecisionFDA's BWA-MEM app with default setting, and then got the HG002_NIST_150bp_50x.bam file as output. The file was further processed using SAMtools 1.9 and HTSlib 1.9 to address formatting problems caused by an older version of HTSlib:

```
samtools view -bh HG002_NIST_150bp_50x.bam -o
HG002_NIST_150bp_50x.bam
```

The FASTQ files are originally from the Genome in a Bottle Consortium.

- BAM file: pacbio.8M.30x.bam.

Publicly available PacBio BAM file. ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/AshkenazimTrio/HG002_NA24385_son/PacBio_SequelII_CCS_11kb/HG002.SequelII.pbmm2.hs37d5.whatshap.haplotag.RTG.10x.trio.bam

- FASTA file: hs37d5.fa.gz

The original file came from: ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/technical/reference/phase2_reference_assembly_sequence. Because DeepVariant requires bgzip files, we had to unzip and bgzip it, and create corresponding index files.

- Truth VCF and BED

These come from NIST, as part of the Genome in a Bottle project. They are downloaded from ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/release/AshkenazimTrio/HG002_NA24385_son/NISTv3.3.2/GRCh37/

Bibliography

1. Audano, P. A., Sulovari, A., Graves-Lindsay, T. A., Cantsilieris, S., Sorensen, M., Welch, A. E., … Eichler, E. E. (2019). Characterizing the Major Structural Variant Alleles of the Human Genome. Cell, 176(3), 663–675.e19. doi:10.1016/j.cell.2018.12.019

2. Telenti, A., Lippert, C., Chang, P. C., & DePristo, M. (2018). Deep learning of genomic variation and regulatory network data. Human molecular genetics, 27(R1), R63–R71. doi:10.1093/hmg/ddy115

3. Ryan Poplin, PiChuan Chang, David Alexander, Scott Schwartz, Thomas C olthurst, Alexander Ku, Dan Newburger, Jojo Dijamco, Nam Nguyen, Pega h T. Afshar, Sam S.Gross, Lizzie Dorfman, Cory Y. McLean, Mark A. DePristo. (2018). Creating a universal SNP and small indel variant caller with deep neural networks.
bioRxiv 092890; doi: https://doi.org/10.1101/092890

4. Alberts B, Johnson A, Lewis J, et al. Molecular Biology of the Cell. 4th edition. New York: Garland Science; 2002. The Structure and Function of DNA. Available from: https://www.ncbi.nlm.nih.gov/books/NBK26821/

5. Alberts B, Johnson A, Lewis J, et al. Molecular Biology of the Cell. 4th edition. New York: Garland Science; 2002. Chromosomal DNA and Its Packaging in the Chromatin Fiber. Available from: https://www.ncbi.nlm.nih.gov/books/NBK26834/

6. Liu, L., Li, Y., Li, S., Hu, N., He, Y., Pong, R., ... Law, M. (2012). Comparison of next-generation sequencing systems. Journal of biomedicine & biotechnology, 2012, 251364. doi:10.1155/2012/251364

7. Muzzey, D., Evans, E. A., & Lieber, C. (2015). Understanding the Basics of NGS: From Mechanism to Variant Calling. Current genetic medicine reports, 3(4), 158–165. doi:10.1007/s40142-015-0076-8

8. H.Richard Johnston, Pankaj Chopra, Thomas S.Wingo, Viren Patel, International Consortium on Brain and Behavior in 22q11.2 Deletion Syndrome, Michael P. Epstein, Jennifer G. Mulle, Stephen T. Warren, Michael E. Zwick, David J. Cutler. (2017). Simplified approach to whole-genome sequencing. Proceedings of the National Academy of Sciences Mar 2017, 114 (10) E1923-E1932; DOI:10.1073/pnas.1618065114

9. Google (2019). google/deepvariant. [online] GitHub. Available at: https://github.com/google/deepvariant [Accessed 23 Sep. 2019].

10. Illumina (2019). Illumina/hap.py. [online] GitHub. Available at: https://github.com/Illumina/hap.py [Accessed 31 Aug. 2019].

11. Eberle, M. A., Fritzilas, E., Krusche, P., Källberg, M., Moore, B. L., Bekritsky, M. A., ... Bentley, D. R. (2017). A reference data set of 5.4 million phased human variants validated by genetic inheritance from sequencing a three-generation 17-member pedigree. Genome research, 27(1), 157–164. doi:10.1101/gr.210500.116