Universitat Autònoma de Barcelona

Adaptive calibration of MAPQ scores using machine learning techniques

Soroush H.

Tutor **Santiago Marco-Sola**
Tutor **Antonio Espinosa**

Thesis submitted in partial fulfilment of the requirements
for the master degree in Bionformatics
of the Universitat Autònoma de Barcelona, June 2018

Abstract

Nowadays, one can now get their genomes sequenced for just over 1000 USD and ascertain their genomic individualities; ranging from ancestry to genetic diseases they may be prone to [1]. The process can be simplified into four basic steps. First, a donor gives a blood sample to a medical professional to get it sequenced. Second, the DNA is extracted from the blood and goes through a machine (i.e. sequencer) which breaks it up for processing and reads those pieces, referred to as sequences or reads. Third, these reads have to now be aligned to re-create a digital version of the users genome. Lastly, the genome is sifted through to look for any patterns in the DNA which can be associated with medical markers, for example variation in the BRCA set of genes which are good indicators of breast cancer in women [2]. The first step can be performed by simply going to any local clinic. The second step has become automated with various sequencing machines and techniques. The last step is just a matter of passing the information obtained through an ever growing library to find patterns. The third step should just be a simple matter of aligning the reads given in step 2, but it's a bit more complicated than that. DNA is created from only four base pairs (i.e. A, C, G and T). With only 4 letters, a lot of redundancies begin to appear when trying to find the exact position of a read. With the most commonly used Illumina length currently being only 150 base pairs long, and a genome of 3 billion base pairs to generate, difficulties in mapping begin to arise [3,4].

Currently all available methods try to piece together the presented pieces against a reference DNA [5]. Humans have about 1 in 1000 different DNA letters, permutations, between one another, so after the tools find a few positions where a read could be, it's just a matter of taking the one deemed to be the most accurate [6]. Each tool uses its own method for determining the location, and then generates a score of how accurate it believes its prediction is. Being as accurate as possible is of paramount importance [19]. If a read is placed in the wrong location, the net result can be accidentally telling someone they have some hereditary disease in step 4, or worse yet, missing a preventable one. Current tools give inaccurate scores, as in, if one sums sum all the supposed '90% chance of being right' reads, they won't obtain anywhere near 90% of them being correctly placed and 10% misplaced. This occurs on high ends where a tool will claim a score of 60, which is theoretically 99.9999%, but then proceed to only obtain 99.99%. This nulls the reason of having a wide range of scores. Worst of all, it results in thousands of misplaced reads even after one filters against the lower scores. When dealing with human lives, and an ever growing population of over 7 billion [7], this should be reduced to as close to 0 as possible.

With no access to the mapping process itself, post mapping results can be evaluated and filtered in order to obtain better predictions than the mappers themselves. Machine learning, which is the use of statistical techniques in computer science to teach a model to learn and infer results was perfect for such an application [8]. By learning which reads the mapper has difficulty with, and which it underestimates, a model can be generated which predicts better than the mapper itself. After attempting many different machine learning methods, and running thousands of tests, I propose a tool with better scoring and accuracy, with results as high as 100% in some instances, which was not present in any other tool. The method used also demonstrates flexibility in that it can be trained to be used for different genomes, tools, or read lengths.

Acknowledgements

I would like to thank my family for their forever support. My father for pushing me to come and do a shotgun masters in Barcelona, because of his “You miss 100% of the shots you don’t take” mentality. My mother for her care, and desire for me to go explore whenever the opportunity arises; without her blessing I’m not sure I’d have been to convince myself to come. And my brother for being someone I strive to be the best role model for.

I would like to thank the bioinformatics masters department of the Universitat Autònoma de Barcelona for giving me the opportunity to be here and doing something I never thought, even a year ago, I would be doing. I would like to thank Edgar Gallant and Dahv Reinhart for last minute proof reading my paper.

Last but not least I would like to thank my masters coordinator Santiago Marco-Sola who without none of this would be possible. He not only allowed me to work on a project I found interesting, but he supported me in every direction I chose to take it, all the while being flexible and patient throughout. Moltes gràcies.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

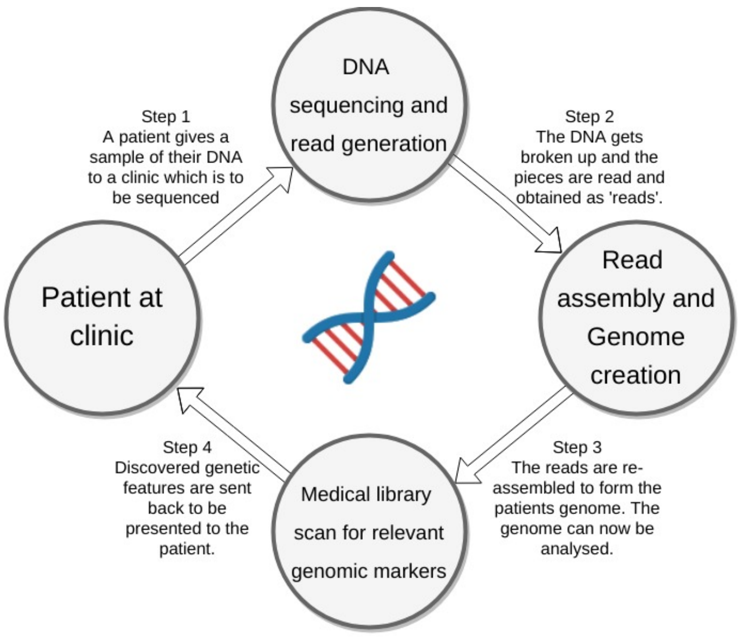
DNA sequencing and read generation
Step 1
A patient gives a sample of their DNA to a clinic which is to be sequenced
Step 2
The DNA gets broken up and the pieces are read and obtained as 'reads'.
Patient at clinic
Read assembly and Genome creation
Step 4
Discovered genetic features are sent back to be presented to the patient.
Step 3
The reads are re-assembled to form the patients genome. The genome can now be analysed.
Medical library scan for relevant genomic markers

Figure 1.1: The circle of genome sequencing.

1.1 A Story

When computers were still in their infancy, Watson, Crick, Wilkins and Franklin solved the double helix shape of DNA with the use of X-ray crystallography [9]. Half a century later, the first rough draft of the complete human genome was completed with the aid of over 1000 scientists, now archaic computers, and a 3 billion dollar budget [2]. Today, with around 1000 USD and a few days time, anyone can have their genome sequenced. As this capability in both technology and computation has grown and proceeds to do so, so too have the challenges and hurdles which accompany it. Accuracy is of the utmost importance, for having a single nucleotide wrong can change a diagnosis from ’potential cancer marker’ to ’completely healthy individual’ [11].

1.2 Summary

MapQ Score computation was refined through the use of machine learning techniques. Logistic Regression, Random Forests, XGBoost, Support Vector Machines, and Neural Networks are all machine learning algorithms which were, in some way, tested for their potential. Random DNA reads of desired lengths were generated using Mason [12], and GEM3 [10] was utilised to generate predictors through its attempts of mapping said reads to a desired spot. These predictors were then fed to each individual machine learning algorithm in various ways, alongside the result of match or miss by the Gem3 tool given the reads predictors. The tools were then able to surmise whether there was in fact a hit, or a miss, through pattern recognition between the predictors and the outcome.

On samples of 10 million 150 bp reads, current tools would obtain in the hundreds and even thousands of mistakes when attempting to correctly identify the position of the read even in the most sensitive of cases. A model was generated which is capable of consistently achieving false positives in the single digits, and sometimes zero when sensitivity is a concern, with an overall better Accuracy and F1 score than other methods. More accurate results, as well as better scores, were obtained through a plethora of methods, demonstrating the power and potential of machine learning for this application.

The results on different read lengths also indicate the flexibility of the method in working for other mappers and read ranges were someone inclined to generate one themselves.

1.3 Objectives

My desire with this practicum was to obtain skills in the field of machine learning while dabbling in the field of bioinformatics. With a desire to advance towards more Computer Science oriented work, specifically in the growing field of machine learning, having the good fortune of having our Algorithms professor, Santiago Marco-Sola, as my tutor was more than I could have asked for. I feel confident in having fulfilled this masters, that I have become much more cultivated and confident in the fields of Machine learning and bioinformatics.

The project evolved over the months, as projects tend to do.The initial goal was to obtain a background in the field, create tools to benchmark my results, and create a simple Logistic Regression model [14] to obtain more accurate MapQ scores [15]. Those ended up being relatively rudimentary, and as I learned more, the faster I was able to learn, apply and improve other tools. Improving MapQ score Accuracy, obtaining higher accuracy through lower false positives, and demonstrating flexibility for multiple read lengths, all through the use of multiple machine learning methods, were the final three goals of my project.

1.4 Outline

The background section covers all the ground level concepts ranging from how mapping tools work to machine learning techniques and performance goals. In the methods, step by step procedures are presented; How data is prepared and processed, how models are taught and ran, and how the result are obtained and analysed. The Results section contains the most important figures and tables of this thesis along with a discussion surrounding the findings. These include improvements to MapQ Score Accuracy, read mapping accuracy, as well as general performance measurement tools such as F1 score. It also contains suggestions on future improvements and limitations of the methods. The last section is the conclusion where the results are summarised and final deductions are presented.

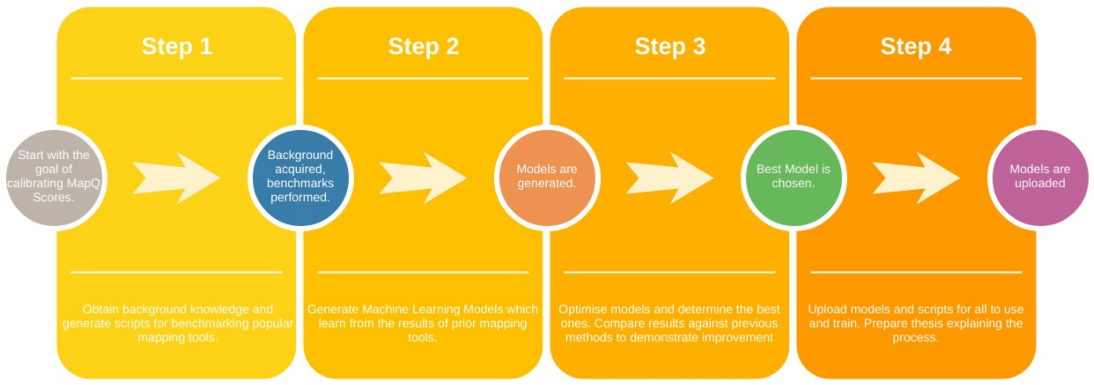
Step 1
Step 2
Step 3
Step 4
Start with the goal of calibrating MapQ Scores.
Background acquired, benchmarks performed.
Models are generated.
Best Model is chosen.
Models are uploaded
Obtain background knowledge and generate scripts for benchmarking popular mapping tools.
Generate Machine Learning Models which learn from the results of prior mapping tools.
Optimise models and determine the best ones. Compare results against previous methods to demonstrate improvement
Upload models and scripts for all to use and train. Prepare thesis explaining the process.

Figure 1.2: Step by step work process and milestones.

Chapter 2

Background

2.1 High-throughput Sequencing

High-throughout sequencing, previously dubbed "next-generation sequencing", is to thank for the fast and low cost reality of quick genome sequencing. Methods range from Illumina tech completing a genome in 1 day at a cost of 5 cents per 1 million bases, to Sanger sequencing being capable in only 20 minutes at a much larger cost of 2400 dollars per 1 million bases [16,17]. All the methods function in a similar way. In order to save time, DNA is blasted into small components which are each read, and all these pieces, dubbed 'reads', are then aligned together in order to generate a whole genome [16,17]. Rather than blindly aligning the reads one at a time, which can be time consuming, these reads are fed to a beefy computer which performs the alignments through a method known as Sequence Mapping. Sequence Mapping is the process of comparing reads with the reference genome in order to generate the one of interest [18].

There are many mapping tools which attempt to place the piece of DNA into the correct position in the genome for assembly. Their functions can be explained rather easily. They go through the genome, find a position(s) where the piece of DNA could be, and attempt to give a score of confidence of whether it is there or not [5]. They each have their own custom methods of achieving this, but almost all of these tools tend to first build their own hashed index from the genome to more quickly search through them for a match, and then use that index over the entire Genome for finding a position. The time to process and accuracy of mapping varies between them as do various options they offer [23].

2.1.1 MapQ Scores

MapQ Scores, or Map Quality Scores, are a value represented by -10$\log_{10}$Pr, where Pr is the chance that the mapping position is wrong, rounded to the nearest integer [15]. For example, if the probability for a read to be in the correct position is 99 percent, then the MapQ score will be 20 : ($\log_{10}$(1 - 0.99)). 0.999 percent would be 30, and so forth. Generally, a user would self filter out poor scores for better results. Depending on which program is used, one may obtain different MapQ scores for the same read, presenting what the program perceives to be the accuracy of the read it's placing.

Unfortunately current scoring methods aren't very accurate. They generally limit their scores from 0 through 60 with some exceptions, and with vague to no proper reference as to how these are generated [23]. Maybe a score of 20 is as accurate as one of 30 in another tool, as consistency isn't seen throughout all mappers. This leaves the user in the dark, to have to make assumptions and decisions which add to processing time and illogical human errors. This is especially the case when a score of 60, which is meant to mean 99.9999 percent accurate, is really only 99.99 or so percent most of the time. For that reason, the MapQ Score accuracy is paramount and the whole reason to even have them. If a read is given a score equivalent to 40 percent accuracy, then if you sum up all the reads of that accuracy, you should obtain around 40 in 100 being hits, and 60 not. In the subsect of mapping however, a user wants to either confidently confirm it's a hit, or declare it as a miss. It's better to discard a read as a miss, than to have a false positive, and accidentally put a read where it's not meant to go. For that reason, reads of under a score of 20, or 99 percent accuracy, are generally thrown out, and the assumed misses are thrown in the sub 20 region leading to inaccuracy in the sub 20 region for an attempt at higher accuracy in that of the 20 and over [13].

Reads can be present in varying lengths depending on the size they're blasted into. The larger the read, the more base pairs can be matched, and there should therefore be higher accuracy. Consequently, shorter reads will have less accuracy. Mappers have to take this into account and their scores should consequently take a hit in accuracy when dealing with smaller ones, and be more confident when dealing with larger reads. As 150 bp Illumina read lengths are the most popular used [4], this project revolved around creating a model which manages read lengths of that size the most optimally. However the generated tools can and should be trained for read lengths of all sizes.

2.2 Machine Learning

Machine learning (ML) and artificial intelligence have been revolutionising all tech related fields. Google's pixel phones, self driving cars, and recommendations made by Amazon for your next purchase, are all examples of this giant step into the future [8, 24]. Google is so confident in its potential that they spent half a billion dollars purchasing a ML company, Deep Mind, in 2014 [21]. They aren't the only ones; Facebook, Microsoft, Apple, IBM, and so forth are all purchasing and investing in companies when they can [22]. The premise is simple: Give a computer a large enough sample size and you can determine optimal and accurate trends and patterns. In a world of social media and online interactions, there is so much data available that the technology has been able to see a huge growth [23].

Logistic Regression [14] (LR), Random Forests [25] (RF), XGBoost [26], Support Vector Machines [27] (SVM) and Neural Networks [28] (NN) are the 5 machine learning methods which were tested. The first four are some of the most popular methods and also considered simple for newcomers to the field which is why they were methods of choice to begin with [29]. Only once a base of knowledge was developed with some of those techniques were Neural Networks and Deep learning tested. Neural Networks were chosen due to their immense potential in all fields [30]. The bulk of script work was coded in Python [31]. The Logistic Regression model was set up in Octave [32] with skills obtained through the use of the Andrew Ng coursera course [33]. R [34] was used as the language of choice for Random Forests, XGBoost and SVM, as efficient libraries had already been written. Finally TensorFlow [35] was installed, and with the use of Keras [36], Neural Networks were trained in python. Logistic Regression and Random Forest methods were also coded in python, but for the sake of simplicity and time, the most efficient aforementioned ones were used to run everything.

2.2.1 Classification and Regression

Machine learning Algorithms can generally be divided into two types: classification and regression. Classification is done by grouping the results together, or classifying them, whereas regression is predicting the output through probability, aka the odds of it being one over another [37].

In the case of MapQ scores, although it would be ideal to be able to classify reads as 'hit' or 'miss', unfortunately it isn't as clear cut as separating the image of a human to that of a car would be. There aren't any humans who look identical to cars or vice versa, but there are perfectly good reads which are out of position due to unfortunate luck. For MapQ applications, regression is much more applicable to self sort high accuracy reads from low accuracy ones at certain cut offs, in this case, being around the 99 percent accuracy mark.

That isn't to say it wasn't attempted. Classification was tried and tested, both through Random Forests and SVM's, but the results were not anything promising to devote extra time towards. Instead, a better method is to make cut offs in Regression models to determine hits and misses, which is effectively self made classification.

2.2.2 Linear and Logistic Regression

Linear regression is the simple machine learning technique of sorting your data in a linear fashion. For example from smallest to largest, and assigning a score based on the position of the features on the scale. Logistic Regression just takes linear regression and sorts it between 0 and 1 using the Sigmoid Function; this allows probabilities to be easily discerned. So as linear regression is continuous, Logistic Regression is set between a range making it ideal for probability measurements [38].

Once all the values are placed in some order, the model has to be tested. The cost function quantifies the amount of error between what your point is and the correct answer, whereas Gradient descant is the learning algorithm that attempts to minimise the error. So given many points, one would want gradient descent, using the cost, to find the point where there's the least cost overall out of all possibilities, as that is where the value is closest to ideal for your overall sample. This will result in an output theta for each parameter.

The sigmoid function, with formula:

$$s(x) = 1/(1 + e^{-x})$$

is what is used to place the linear regression results between the ranges of 0 and 1, or a probability of 0 to 100 percent [39]

2.2.3 Random Forests

Random decision forests are another type of supervised classification algorithms. They can be used for both classification and regression tasks. These work by creating multiple decision trees, and then merge them together to obtain a stable precision, in a process called bagging [40]. A decision tree is effectively a subset of learned samples. It attempts to decide whether the read it is viewing is a hit, or a miss. By taking multiple random samples from the learning set, and taking the option (hit, or miss) which appears more, it is capable of accurately determining whether a certain sample is in fact a hit or a miss. In the case of Random Forests, these are fully grown decision trees which have low bias, but high variance. They attempt to minimise error by reducing variance. The trees made are unrelated, so by creating many and averaging out the best outcomes, you obtain a tree which should in theory function well for your sample. By re-sampling the data constantly for each sample, you obtain different classifiers which over-fit the data in their own way, and then you average out the results to make up for the over-fitting [41].

2.2.4 XGBoost

This relatively newer and extremely popular method has been winning competitions left and right [41]. It is similar to random forests in that it uses decision trees. However, unlike random forest it uses boosting, which is based on weak learners. These are shallow trees, which get extended upon through each iteration. It is effectively creating its own fork in the road at each decision tree which a sample has to pass through to arrive at the next sample. Classifiers are added one at a time, so the next classifier should be improving the current one, instead of making separate ones like they would be in Random Forests. This results in high bias, but low variance [43].

2.2.5 Support Vector Machines

Support Vector Machines (SVM) are also a supervised machine learning algorithm which have the ability to be used in both classification and regression [44]. These work by drawing a fine line between hits and misses for all samples. Effectively placing them into finely separated groups. It then attempts to find the best separating line, instead of finding a line through the data [45]. This algorithm is better for smaller data sets, but it isn't suited for larger ones, such as for a large number of reads, as the training time can be very high. This is because it is stored in a NxN kernel matrix, where N is your number of samples. If you do not store the kernel matrix, then you would have to recompute the values repeatedly making it extremely slow [46,47].

2.2.6 Neural Networks - Deep Learning

Neural networks aim to emulate how a human brain would work by taking a lot of decision trees to act as neurons, which 'speak' to one another. A way to imagine this is a human being asked a question, which is the input to the first neuron. This then sends the information to multiple neurons which perform calculations. One could be factoring the words used, one the tone used, one the language, and so forth. These calculations are then combined to form some output or 'answer'. Each Neuron is effectively performing logistic regression or some other statistical method, but with different weights and possibly different inputs, before they all output one cohesive response [28].

Although there's no clear definition of when a neural network is in fact 'deep learning', it is generally considered deep when it has at least 2 hidden layers [48, 49]. This is the 'magic tool' which is making great advancements in the field of artificial intelligence. Self driving cars, photo recognition, drug design and so on have seen great success through the use of neural networks [8]. It only seemed appropriate to give it a go at trying to improve MapQ Scores.

2.3 Data Handling and Evaluation

2.3.1 Data

The lander/Waterman equation is used by Illumina to estimate how many reads one needs for full coverage of a genome. Ideally a model should be touching, or learning from, every piece of a genome so it can confidently have learned from all of it [50]. This is represented by the equation

$$C = LN/G$$

Where C is the coverage, G is the haploid genome length, L is the sequence read length and N is the number of sequence reads. Generally, a minimum of 30x coverage is recommended in order to obtain optimal input read depth to missing calls [50, 51]. That is, generally 30x the length of the human genome is recommended to be mapped for efficient accuracy.

Other than making sure enough samples are used to train a model, another problem which arises when making any tool is to make sure it's not over specific for your sample set (over fit), or too broad resulting in poor results (under fit). Both will result in inconsistencies when measuring your tool on other samples. Over-fit can be thought of has high bias. It's when the tool is considerate only for the sample it has been trained on and will have difficulty seeing similarities with other samples. Under fitting can be thought of as high variance, where the tool may be too aloof in its judging resulting in too broad of assumptions being made when evaluating other samples [52].

Regularization is a method which can be used to further reduce error by reducing over fitting. This is done by attempting to reduce the bias mentioned before, by introducing variance. A value, dubbed lambda, is added to the cost function which changes how the gradient function works, making it better at generalising as it will no longer be optimised solely for the given data set [53, 54].

Normalisation of Data is another issue. Data sets contents won't be identical in range. For example the weight in kilograms of a fossil as one parameter with a range of 40 to 100 kg, will have a much different range than the age of the fossil. Having inputs be normalised leads to faster training and lower chance of getting stuck in a local optima, among other benefits [55].

2.3.2 Methods of performance measurement

There are many ways of measuring the performance of the varying tools. In machine learning this is generally done through the use of Accuracy, Precision, Recall and the F1 Score. Using ROC Curves and measuring the area under the curve (AUC), performance can also be measured [56, 67].

From most data sets one is left with fourth groups of values. True positives (TP) are the correctly predicted positive values (Hits). True negatives (TN) are the correctly predicted negative values (Misses). False positives (FP) are the falsely identified as positive values (These are misses, but the tool thought were hits). False negatives are the falsely identified as negative values (These are hits, but the tool identified them as misses).

I Accuracy

Accuracy is the first calculated model, that is, the simple ratio of correctly predicted results to total results. This is possibly the most important metric when comparing tools, as false positives are generally considered very undesirable.

II Precision

Precision, this is the ratio of correctly predicted positive observations to the total predicted positive observation. It can be presented as

$$TP/(TP + FP)$$

Precision shows of those which were claimed to be hits, exclusively, what percent were in fact hits.

III Recall

Recall, also called sensitivity, is the ratio of correctly predicted positive results. Presented by

$$TP/(TP + FN)$$

This is also very important and usually goes hand in hand with Accuracy. Recall shows how many of the total number of possible positives were actually obtained.

IV F1 Score

Finally, the F1 Score is the weighted average of precision and recall. This takes into account both false positive and negatives and is generally considered a good metric for comparing total performance. This is presented as:

$$F1Score = 2 * (Recall * Precision)/(Recall + Precision)$$

2.3.3 Performance Goals

There's a balance to be had in regards to performance. To give an example: when scanning for cancer from images, a doctor prefers a more sensitive tool which has a high number of false positives, aka is more sensitive, in order to not miss anyone who actually may have the disease.

They would rather say someone has cancer and have them go through the tests than miss it for someone and have them die.

In the case of spam mail, the email client wants the tool to be less sensitive so that the user will receive all real mail. This results in false negatives from time to time, aka spam in their email box. It's less sensitive, but still able to filter out some spam, without ever affecting any real emails.

In the case of reads, somewhere in-between is preferred, with a preference to the higher accuracy case as in the doctor example. It's preferable to let in those which are as close to 100 percent accurate as possible in order to avoid false positives, but not at the expense of 99 percent of your samples. At the same time you would rather have significantly less than 0.1 percent in false positives, as in an entire human genome of 3 billion base pairs, that would result in 2 million misplaced positions with 150 bp reads.

Chapter 3

Methods

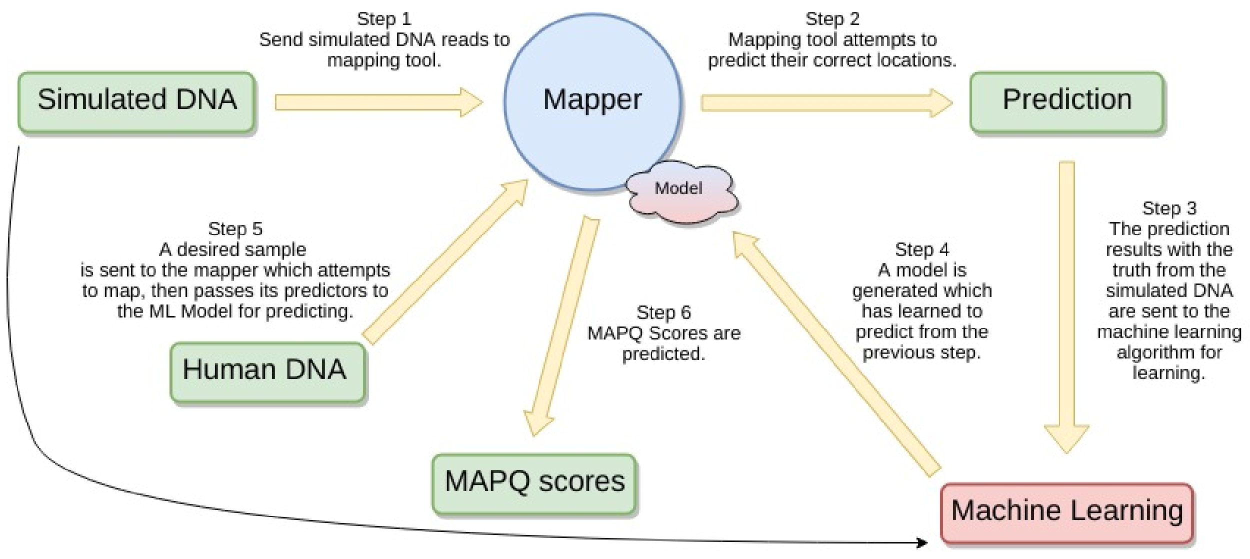
Step 1
Send simulated DNA reads to mapping tool.
Step 2
Mapping tool attempts to predict their correct locations.
Simulated DNA
Mapper
Prediction
Model
Step 5
A desired sample is sent to the mapper which attempts to map, then passes its predictors to the ML Model for predicting.
Step 4
A model is generated which has learned to predict from the previous step.
Step 3
The prediction results with the truth from the simulated DNA are sent to the machine learning algorithm for learning.
Human DNA
Step 6
MAPQ Scores are predicted.
MAPQ scores
Machine Learning

Figure 3.1: Pipeline overview of mapping to model to prediction process.

3.1 Scripts

The minimalist version of all scripts and bash code can be found at:

https://github.com/roushrsh/BioInformaticsMapQMachineLearningMethods

These scripts are referred to as minimalist as certain variables have to be changed for different test cases. All in all, close to 500 unique scripts of varying models were generated for Logistic Regression, Random Forest, XGboost and SupportVectors, and over 2000 unique Neural Networks each with their own saved script (for reproducibility purposes). The XGBoost model and two preferential Neural Network models are available in the github account for all to use.

3.2 Methodology - MapQ Score Generation

Predictors, currently obtained from the Gem3 tool, were taken and normalised to an approximate 0 to 1 range through division. This was based on their maximum found value in a sample range of 100 million reads for the specific length of the reads.

For the logistic regression model, these were multiplied by the obtained theta's to obtain percent accuracy, and then converted to MapQ scores using the accuracy to MapQ formula as seen in the background section. This results in a score for each read using the Logistic Regression model. Logistic Regression was tested with different sample sizes, gradient descent algorithms, with and without regularization, as well as in Python vs in Octave before settling on the two current models. Random Forest, XGBoost and SVM perform their own classification methods as explained in the background using the same training data set (albeit smaller due to the individual ram limitations of these methods). XGBoost scores were also combined with logistic regression scores in certain ranges, as seen in the results, in order to obtain the best of both methods, i.e. parts which each method excels at, in order to obtain the highest possible accuracy. Deep Learning had many models run with varying sample sizes, seeds, and variables. Ultimately those with the best results were filtered out.

The file format in which all reads are presented is known as the SAM format. SAM stands for Sequence Alignment/Map Format. It has become the standard for representing alignments and reads, though it's basically an organised text format. The coordinates can be found in the references. [57]

3.3 DNA Read Generation : Mason

Before any DNA can be mapped, it has to first be generated with an already known position, and then be mapped so to be compared with the truth to see if mapping was successful.

The mason tool does this [12]. By passing it the genome, it quickly generates randomly formed 'cuts' which it exports along with their locations to a '.sam' file. Mason2 was used for Chromosome 1 DNA generation, but mason 0.1.2 had to be used for the human genome due to bugs with ram handling. The Illumina method was used for all samples for time and consistency [58].

3.3.1 Generating samples for the Logistic Regression Model

A minimum 1x coverage in the human genome was desired for the logistic regression learning algorithm. So the Lander/Waterman equation was considered [50]. The human genome has roughly 3 billion base pairs. Assuming 150 base pair reads, 20 million reads (N) would be required to reach 1x coverage:

$$1 = 150 * N/3000000000$$

When testing chromosome 1, it was quickly made evident that having more samples resulted in better and more consistent accuracy. As mentioned prior, this is attributed as the reason for the growth in machine learning, as there's a lot of data in the hands of users now, and models are more accurate as a result [23]. This became noticeable during tests when increasing the sample size at a scale of 10. 100,000 samples to 1 million to 10 million and finally to 100 million all resulted in better results, albeit at diminishing returns. So it made sense to try to maximise the available ram when possible, which just happened to be around 100 million samples.

Due to a ram limitation of 64 gb's, the 100 million samples couldn't be generated all at once by mason using the human genome, even if the Logistic Regression model was capable of processing that many. Instead, 80 million unique samples were generated with base settings (no seed), followed by another 20 million with a seed of 1, and these two were merged in order to obtained a 100 million sample file for teaching the Logistic Regression model. This file was taken apart to teach other tools, ex: 10 million of it used to teach XGBoost. As the samples to teach will invariably have bias on the testing sample, a separate 10 million read (10MR) sample was generated using a seed of 2 on which all the mappers and ML methods were tested upon. Later, seeds of 100, 200, 300, 400 and 500 were used to generate 5 more 10 million samples for further bias verification on for promising models.

As mason allows the user to choose the length of reads, it was also used to generate 100 million 25bp, 50bp and 100bp reads for further testing. A fair point is that the smaller reads take less room and therefore require less ram. For the sake of time and simplicity, only 100 million were generated for those as well. This gave over 1x coverage for the 50bp and 100 bp samples, whereas 25 bp would have required only 20 million more, which didn't appear would be too drastic of an improvement given the relatively slight improvement in performance observed when increasing the sample size from 10 million to 100 million.

Gem3 [10], BWA [59], MiniMap2 [60], SNAP [61] and BowTie2 [62] were the mappers used as a benchmark to compare to the generated machine learning model results. They each made their own indexes and then attempted to map the mason generated reads using their own algorithms and methods. It is important to note that some tools such as SNAP couldn't map smaller read lengths, or in the case of BWA had to use a different set of commands, which can be found in their manuals. It is also important to note that Gem3 in particular is the foundation the machine learning models were made from. The generated models are built in order to improve Gems own scores (above its own and other tools) using its own generated predictors. This is possible as Gem3 has the option of outputting the predictors for each of the reads it maps. These being the values pertinent to the specific read, such as position in genome, size relative to genome, etc, which can be obtained and used to train statistical models such as machine learning ones.

3.4 Benchmarking the tools

3.4.1 Mapper evaluation

After the mapping tools were run and scores were given by the tools, their perceived locations by the mapper was compared to the ground truth given by mason in order to obtain Hits and Misses for each MapQ score given.

Script 1 takes the position given by mason for each read, and compares it to that of the mappers estimated position in the genome (both .sam files). This is done with a 10 base pair leeway in each direction (up and downstream). This is because mild errors in mapping can occur from time to time from repetitive DNA strings. Different ranges of leeway were tested. A leeway of 2 base pairs still resulted in some mild error for some tools, so the range was increased to 10, as there wasn't a significant, if any, difference when further increasing it (for the tested samples). The file generated here is dubbed "oneZeroWithLeeway.txt".

3.4.2 Benchmarking

Script 9 is the evaluation script. It takes the MapQ Scores given by the tools, and compiles them against the actual (hit or miss) given by script 1, oneZeroWithLeeway.txt. It then outputs a myriad of information, most importantly the number of hits and misses for each MapQ Score, which allows for the benchmarking of each tool. It is important to note that some tools, such as SNAP and BWA, either won't have their output scores in the order as they appear in the mason file, or can have multiple entries for a single read. The former case requires sorting, and the latter was dealt by taking the read with the highest score which was always the first position in the .sam file for that read. Bash [63] commands were used to quickly handle both issues as seen in the github.

3.5 Generating Data for ML and running the scripts

3.5.1 Data Preparation - Bash

Some data has be extract before proceeding to any machine learning. First, the mason .sam file as mentioned prior has to be sorted in order. From these files egrep was used to cut out the location of each read, as perceived by the mapping tools, to a file dubbed "mappedOut.txt". Then egrep was used to cut out the given MapQ scores to a different file dubbed "mapScores.txt". Finally, it was used to cut out the true locations from the mason file to a file named "truth.txt". These are generated for self comparison, lower ram use, and bug analysis rather than scrolling through the large, sam files separately. A length.txt file is also generated for convenience to remember the length of reads being used and to allow the scripts more flexibility. As mentioned prior, Gem3 was used to obtain predictors for each read that it could, which were then used to both teach and test the machine learning algorithms. Gem3 outputs 21 predictors total for each read which it was capable of mapping. Some of which are completely useless, others which are relatively less useful than others. In order to determine which predictors to use, each column of predictors was scrolled analysed once again with bash commands.

The results can be seen in table 8.23. Quite a few predictors gave only a value of 1 and were excluded. Everything else was included in the Logistic Regression model, as after some tests with 12, 13, 14 and 15 samples, omitting those with few variables (such as column 14), it was found that including all 15 led to the most accurate results in the case of logistic regression by a small margin. Those being columns 2, 3, 4, 5, 6, 11, 12, 13, 14, 15, 16, 17, 19, 20 and 22 (Figure 7.23). Those 15 rows were cut from the predictors file separately and used to train the ML model for Logistic Regression. The same columns were used to train the XGBoost model. However Random Forest, Support Vector Machines and Neural Networks are far more ram intensive and didn't see as much of a gain from having 15 rows, and were further reduced to 13 columns to allow for more samples. Those being columns 14 and 19 due to the lack of performance gain, but thanks to the huge ram and computing time gain from having fewer parameters.

Each row of predictors was also normalised by dividing them by the largest found value (from a sample of 100 million reads) in order to obtain a rough range of 0 through 1 for each column for the benefits normalisation brings, as mentioned in the background. This is performed for the predictors in the training example, as well as for the predictors in the test sample.

A file named "theNumbersOfThePredictors.txt" was generated from the predictors file to be used in Script 2. Script 2 then generates a file with the positions of all the missing predictors for the training file. This is needed as Gem3, as well as other mappers, aren't capable of mapping all reads to a genome, and therefore predictors aren't generated for the un-mapped positions. In order to teach a ML algorithm, the predictors are required, along with the outcome of Gem3 mapping. This is for it to learn and discover which combination of predictors Gem3 can map, and which it struggles with. Therefore only the truth which matches a predictor needs to be merged with the predictors file for learning. Script two therefore goes through the position of the predictors and generates a file containing the position of the ones which are missing from the truth. This file can also be useful for troubleshooting bugs later. It uses the length file to know how many items there are and how far to go. The file it prints is dubbed "missingPredictors.txt" which contains as its name implies, the position of the missing Predictors.

As the missing predictors don't have predictor values, they can't be used to train any models. There's nothing there to teach. Script three takes the position of the missing predictors, and removes the should be 0's (misses) in the positions where there are no predictors from the "oneZeroWithLeeway.txt" file, generating a file named "oneZeroWithoutMissingPredictors.txt". The predictors which have been normalised from 0 to the number of predictors are now had, as well as their corresponding hits and misses from using Gem3. These are all that's needed to train a model, but first these will be merged into one file. This is because some Machine Learning tools separate the values from the truth on their own. Script 4 just takes the predictors and their corresponding ones and zeros from the oneZeroWihoutMissingPredictors.txt file and concatenates them together into one file called “mergedPredAndZeros.txt”.

3.6 Running the ML methods

3.6.1 Running Logistic Regression

The data preparation step of separating and dividing predictors had to also be performed on the test 10 million reads (10MR). This is because each ML method predicts using predictors, so the predictors have to be taken from the 10MR, predicted upon, and later analysed.

There were two Logistic Regression models run. One with regularization and one without. The LogisticRegressionTool.m is the one without. It takes in the mergedPredAndZeros.txt file generated in script 4. In the code it uses the first 15 columns as the parameters, and the last column as the truth (the ones and zeros as hits and misses). It requires costFunction.m for the cost function, and sigmoid.m to perform logistic calculations. It outputs a theta’s file which is used to matrix multiply into the 10MR predictors file to obtain MapQ scores for each file. The Regularization method has similar steps. The file used is “logisticRegressionToolWithRegularization.m” But it requires a different cost function, “costFunctionReg.m”. This is similar to the previous costfunction except the code has been edited to include a regularization parameter. It also outputs thetas which can later be multiplied in.

Scripts 6 is what takes the 10MR predictors and proceeds to perform vector multiplication against the obtained thetas. This provides a linear regression result, which then has to go through the sigmoid function, and then the mapScore function to obtain the Logistic Regressions Scores for each predictor [38]. This file is called “mapScoresWithoutMissing.txt” as it does not contain the missing reads which gem was incapable of mapping.

3.6.2 Running Random Forests

For RandomForests, the R randomForest library was used [64]. After fiddling with settings, for the data set, the final model settings were 500 trees with an mtry of 3. RandomForests had a much higher memory requirement than Logistic Regression, so only the first 1 million from LR’s 100 million mergedPredAndZeros.txt was used to train the model.

For this library, the predictors and truth have to be given titles. So a header file with only “a,b,c,d,e,f,g,h,i,j,k,l,m,HitOrMiss” was added to the top of the data set. Consequently, to 10MR “a,b,c,d,e,f,g,h,i,j,k,l,m” was also added. After running Random Forest, each predictor is presented as a percent accuracy which has to be converted to map Scores. Script 5 does a quick calculation to convert percentages to a MapQ Score, this filled will also be dubbed “mapScoresWithoutMissing.txt” as was the case in Logistic Regression.

3.6.3 Running XGBoost

For XGBoost, the XGBoost library in R was utilised [65]. After some experimentation, a max depth of 4, eta of 1, and 100 rounds was used for the obtained results. The model is saved and available in github. Similar to R, XGBoost requires the parameters for the 10MR file to have labels. However due to the lower memory requirements, all 15 parameters were used, and therefore the added label was “a,b,c,d,e,f,g,h,i,j,k,l,m,n,o” for the XGboost file, and “a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,HirOrMiss” for the 10MR predictors file, which also had two more columns than the previous R sample. As XGBoost was a bit more flexible with its memory constraints, it allowed just over 20 million samples to be run of the 100 million used in LR. Ultimately 10 million were used as very little, if any, improvement was seen in scoring from going from

10 to 20 million samples, but the shorter learning time was worth staying with 10 million for the optimisation.

XGBoost does also require its training data to be inputted slightly differently. It takes in two files, one which is the parameters with the labels, and the other which is just the truth with its own label. So the 10MR merged predictors file had to be re-split with the labels into two files. The original predictors only, now with the labels, named TrainP1.csv and the ones and zeros column with the HitOrMiss label, named TrainP2.csv. Similar to R, XGBoost gave its results in percentages of accuracy, therefore script 5 was run in order to obtain the corresponding "mapScoresWithoutMissing.txt" file.

3.6.4 Running Support Vector Machines

For SVM, a SVM library e1071 in R was utilised [66]. Only a few models were tested, but ultimately a simple linear method was utilised for sake of time. Data Prep for SVM was the exact same as R. It required one file with the labels ending with the HitOrMiss. SVM's were by far the most memory demanding method used, and so only 50000 of the 100 million predictors were able to be utilised. Only hits or misses were therefore obtained for each read. Once again Script 5 could be run and a "mapScoresWithoutMissing.txt" file obtained, but the results are distributed differently. All of the hits and misses end up at a score of 60 and 0 respectively, 60 being the max confidence allowed.

3.6.5 Neural Networks

Neural Networks were run in python with the tensorflow framework and keras library [35,36]. Many networks were generated. Different layers, epochs, optimisers, batch sizes, etc. Ultimately the best results came from a neural network with 3 hidden layers of 9 nodes each, using the Adam optimiser, with only 1 epoch and a batch size of 8 or 9. The two presented models 1204 and 1209 are available in the github and can be loaded and run on any relevant sample.

Data Prep for Neural Networks was rather simple. Similar to logistic Regression there was no need to prepare any labels as the code separates out the predictors from the truth. This also goes for the 10MR predictors which it will predict on as is, with no labels required. Having more layers resulted in a bit more running time, as did more epochs and more samples. Memory requirements weren't too drastic due to the efficiency of tensorflow. However the options available for the networks are vast, and running too large of a network presented too little time to test others for what may or may not be a gain in performance, so the parameters tested also took into consideration time restraints. Samples were run with 50 million reads down to samples with only a few thousand. Ultimately it was discovered that the initial random seed (weights) presented the largest variance for model generation. So tests were performed in order to find an optimal seed by running hundreds of separate seeds with only 1 million samples. Neural Network outputs its results as a percent hit, therefore like the previous methods, Script 5 can be used to generate map Scores for those methods as a "mapScoresWithoutMissing.txt" file.

An issue to remark upon which which arises when training Neural Networks is the reproducibility of the results. Setting a seed alone isn't enough, as seen in the keras manuals, three different seeds have to be set prior to loading the keras library. The code has to also be run on a single thread to remove randomness from multi-thread use [36].

3.7 Analysing the ML results

For all the tools, once the map Scores are obtained, these set of scripts can be run to obtain benchmark-able results, as was previously done for the mapping tools.

Script 7 has to be run first. It takes the mapScoresWithoutMissing.txt generated by each ML tool, and the missingPredictors file which contains the position where there isn't a predictor, and proceeds to add 0's in places where there wasn't a predictor. This results in a whole 10 million test file which contains the predicted scores, as well as the misses which Gem3 couldn't place. This file currently contains scores which can theoretically be much higher than 60, so it is dubbed "finalWithOver60.txt". This is because the tools give a score of confidence and a score of 60 is 99.9999 percent confident. If for whatever reason they're more confident than that, the score can consequently be higher. Now the range can be adjusted to match the range of other popular tools, namely Gem3 which the models are based on. Script 8 goes through the "finalWithOver60.txt" file and makes all scores over 60 into 60 to be more consistent with other tools. Other than SNAP, the other tools tested all max at 60 at most, as accuracy beyond that point, and I'd argue even at that point, is hard to quantify. This generates "finalResultsMax60.txt" Now script 9 can be run again, only this time on the predicted MapQ Scores rather than the mappers self generated score.

3.8 XGBoost merged Models

Merged models of XGBoost with Logistic Regression were generated in an attempt to obtain better scores than either tool could offer on their own. As later seen in the results, Logistic Regression with Regluarization (LRwR) only hits well at a score of 60 with good accuracy, and XGBoost does well in all other ranges, along with having an extremely smooth curve. This cross validating the MapQ scores generated by XGBoost with those of LRwR allows for a super model to be generated. The merging is simple. First the script checks the scores in both tools. If the XGBoost score is over 21, which is where XGBoost scores most confidently, it writes the LRwR score as LRwR has a higher range of scores than XGBoost, allow for a better spread of results. Otherwise, if that criteria isn't fulfilled, it writes its own score. This way a smooth curve is still generated with a high accuracy. Other models were also tested by merging XGBoost with LR, Gem3 and so forth, but for the sake of time these were abandoned for neural network improvements

3.9 Training other Models

As the machine learning scripts are all provided, a user can input their own predictors and truths, modify the parameters and train their own model for their own purposes. Only a proper input is required to train the models which can then be used to improve a mappers scores. This is useful in that specific models can be generated for different species and even organisms. The models can be generated in any of the tested machine learning methods, however Neural Networks are recommended as the method of choice. Number of input parameters has to be altered depending on the sample size, and it is recommended to adjust parameters as needed. The base parameters present are were a par of the best found when training with 1 million reads of the human genome, using the specified 13 parameters.

Chapter 4

Results

4.1 Read hit Accuracy Improvements

Method	TP 20 to 60	FP 20 to 60	Percent Accuracy
LRwR	9257446	667335	0.932761
SNAP*	8810864	26893	0.996957
BWA	9040397	2751	0.999697
BowTie2	8508834	2478	0.999709
Random Forest	8635614	1719	0.999801
GEM3	9217282	1612	0.999825
Logistic Regression	7399193	904	0.999878
XGBoost	9201165	976	0.999895
MiniMapper2	8714557	692	0.999921
XGBoost + LRwR	8713879	429	0.999951
NN 1204	8788038	65	0.999992
NN 1209	8374355	55	0.999993

Table 4.1: Percent accuracy, Number of True Positives (TP) and number of False Positives (FP) in the 20 to 60 range, or the ‘desired region’ using various tools and Machine Learning methods. Columns are sorted by Percent accuracy scores. Gem3, BWA, MiniMapper2, XGBoost and neural network models have been averaged from seeds 2, 100, 200, 300, 400 and 500. The rest are using a seed of 2. *Snap is till a score of 70. Data is available in the appendix.

This is the desirable map score region of 20 through 60 region that is generally used for its confidence. 20 being the lower end of 99 percent, and 60 being the upper end of 99.9999 percent accuracy. This region is desirable as some will opt to throw out the ranges of 0 through 20 in order to not risk having false positives in their samples [13]. If accuracy is too low, uncertain regions won’t be predicted upon for risk of false conclusions. BWA, GEM3, XGBoost, MiniMapper2 and the Neural Networks were averaged over all the tested seeds in order to obtain a less biased data set. When it comes to accuracy, the only tools with double digit results are the Neural Network Models. The closest non ML generated method, MiniMapper2, is already accumulating close to 700 false positives on average, while having less True Positives than Neural Network 1204.

Method	TP	FP	Percent Accuracy
Support Vector Machine	8942055	105192	0.9883730
BWA	8687646	2296	0.9997368
Random Forest	8055968	749	0.9999070
SNAP	8267170	676	0.9999182
MiniMapper2	8061615	563	0.9999308
Gem3	8928359	54	0.9999940
LR with Regularization	8015468	46	0.9999943
XGBoost + LRwR	7876320	28	0.9999965
NN 1204	7798136	3	0.9999996
NN 1209	6345552	1	0.9999999

Table 4.2: The best possible obtainable percent accuracy for each tool. Gem3, BWA, MiniMapper2, XGBoost and Neural Network Models have been averaged from samples with seeds 2, 100, 200, 300, 400 and 500. True Positive (TP) and False Positive (FP) values have been rounded to the nearest integer. Data available in the appendix.

This is the highest possible accuracy obtainable for for each tool. For all tools, other than the Neural Networks, the range used is solely the highest score the tool can output. For SVM, BWA, RF, MiniMapper2, Gem3, LRwR and XGBoost+LRwR this means a score of 60, and a score of 70 for SNAP. For Neural Networks it's a range of 21 and up. BWA, GEM3, XGBoost, MiniMapper2 and the Neural Networks were averaged over all the tested seeds in order to obtain a less biased data set, data available in the appendix. The Neural Networks are achieving single digit false positives. NN 1209 in particular has an average of 0.8, sub one FP, with its 6.3 million TP's. NN 1204 is achieving a notable 8 million TP's with only 3 FP's as well. There is a sacrifice of some True positives for this accuracy, but as seen in the appendix results, the range can be increased to up the true positive numbers. These two models are currently the only way to obtain single digit false positives with hits in the 60-80 percent of reads range.

4.2 MapQ score Accuracy improvement

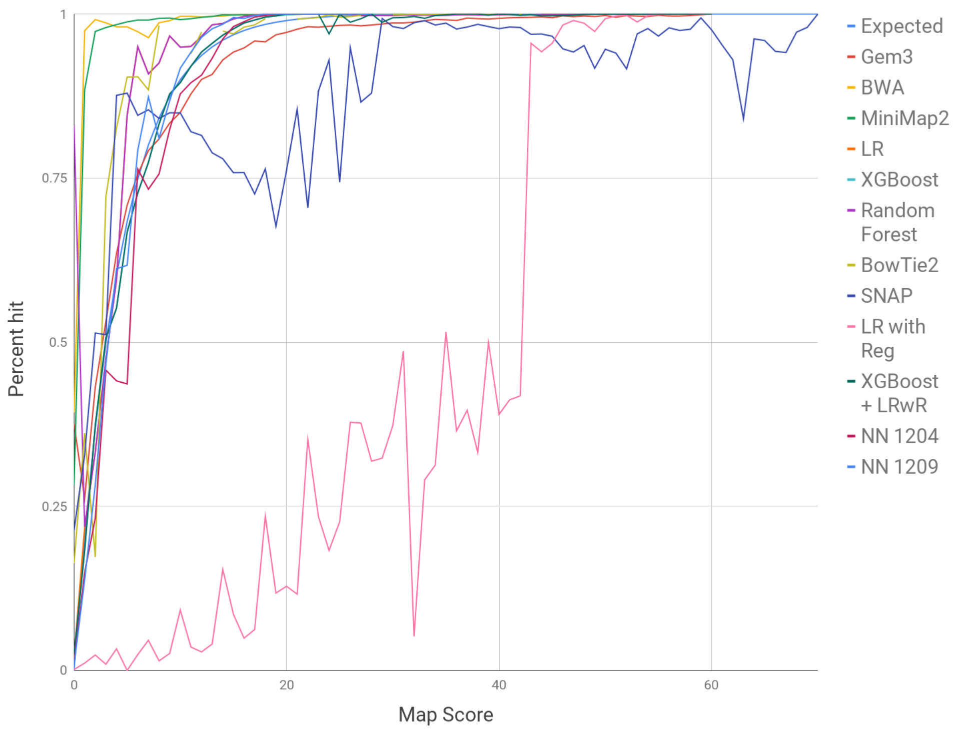

Figure 4.1: MapQ score accuracy for tested mapping tools and machine learning models. Map score is plotted against its respective accuracy for each individual tool. All values are available in the Appendix, as are each individual figure for better visual discretion.

Model	AUC Model vs Expected Seed 2	AUC Model vs Expected Seed 300
LR with Regularization	29.81349	n/a
BowTie2	9.05749	n/a
SNAP	5.71882	n/a
NN 1204v14	4.67863	n/a
BWA	3.85854	3.60438
MiniMapper2	3.74596	3.33947
Random Forest	1.83424	n/a
Gem3	1.11743	1.182407
LR	1.04164	n/a
NN 1204	1.02618	1.025203
NN 1209	0.56146	0.958621
XGBoost + LRwR	0.286947	n/a
XGBoost	0.24996	0.948367

Table 4.3: Area Under the Curve (AUC) difference between the expected and various models MapQ score Accuracies. Sorted from most deviated to least. All calculated with a seed of 2. Data available in the appendix.

From Figure 4.1 and Table 4.3, it is possible to analyse the capability of each tool to output accurate MapQ Scores. These results are shown on the sets of 10MR generated with seeds 2 and 300. Seeds 100, 200, 400 and 500 performed similarly to Seed 300 in regards to scoring accuracy. On the seed 2 dataset, XGBoost models perform by and far the best with a near 1:1 on the Seed 2 data set, followed closely by the Neural Network Models and then the Logistic Regression model. Gem3, which also uses a regression based scoring system, appears to perform above the average. In the appendix the individual graphs can be seen, and from those of BWA and MiniMapper2 it is evident that both place uncertain reads into the 0 through 20 Map Score region. This results in a loss of True Positives from the 20:60 range, into the lower 0:20 range, and damages their overall MapQ score accuracy. This is possibly done as a way to up the accuracy of later scores by being more sensitive. In the Seed 300, the results are similar as before in the order of which tools performed better than others. It is worth noting that XGBoost took a huge hit on all seeds other than 2, specifically a slightly high true positive region in the scores of 2, 3 and 4. Regardless, it still outperforms other models, with the Neural Networks's continuing to be quite accurate.

4.3 Recall, Precision and F1 Score

Model	TP	FP	FN	Accuracy	Precision	Recall	F1 Score
Random Forest	8409599	568183	1022218	0.858343	0.936712	0.891620	0.913610
MiniMapper2	9344965	655036	859291	0.977317	0.998764	0.913973	0.954361
Snap	9181727	429292	388980	0.948769	0.955333	0.959357	0.957341
BowTie2	9220455	722945	56600	0.926781	0.927294	0.993899	0.959442
XGBoost + LRwR	9254563	598448	146989	0.939813	0.939262	0.984365	0.961285
LR	9255631	625497	118872	0.937215	0.936698	0.987320	0.961343
LRwR	9257895	673382	68723	0.932653	0.932196	0.992632	0.961465
NN 1204v14	9257987	681007	61006	0.931899	0.931481	0.993454	0.961470
BWA	9529593	470407	727722	0.973362	0.999512	0.927196	0.961868
Gem3	9675271	324729	702596	0.959546	0.998563	0.929645	0.962752
XGBoost	9634493	365507	103532	0.973097	0.968736	0.989771	0.979089
NN Model 1204	9675271	324729	83367	0.979401	0.975521	0.991423	0.983346
NN Model 1209	9675271	324729	80310	0.979056	0.975272	0.991735	0.983372

Table 4.4: Recall, Precision and F1 of tools sorted from best to worst F1 of tested methods. Gem3, BWA, MiniMapper2, XGBoost and the Neural Networks have been averaged from seeds 2, 100, 200, 300, 400 and 500. Data available in the appendix.

F1 score is considered an important metric in how well a tool performs [67]. It's meant to represent an ideal balance between Precision and Recall. The Neural network Models 1204 and 1209 outperform all tools with their very high F1 Scores. Other tools will sacrifice Recall in favour of precision, or vice versa, but only the Neural Network and XGBoost models appear to have achieved a balance. The neural network models also happen to have the highest overall accuracy out of all models, from all reads with a score of 0 through 60 that is. This is while also achieving the highest number of true positives, thanks to Gem3 based. This can be an advantage for users who want the highest recall possible.

4.4 Discussion

4.4.1 Improving MapQ Score accuracy

Initial Logistic Regression tests were run solely on Chromosome 1. This was to save time due to the limitations of available memory at the time, as well as to prove that such a model could even work. It also presented a good opportunity of learning parameters and becoming more at ease in the use of machine learning techniques, before moving onto the longer to run and process human genome. Through serendipity, it demonstrated the potential of using machine learning techniques to further optimise scores for specific samples. This was because Gem3 gave more accurate MapQ scores on the whole genome than on chromosome 1, and using the logistic regression model made for chromosome 1, worse results were obtained than when using a whole genome specific model. The difference wasn't colossal, but when a 100th of a percent gain is coveted, every small boost can matter. It is very fast to train a model using mason and the genome, or using a specific chromosome of the desired animal, and a scientist would be remiss not to.

To re-iterate MapQ scores: A 0 should be a miss, a 20 should be 99 percent accurate, and a 60 should be more or less 99.9999 percent accurate. In theory, if one were to take 100 scores of 10, which is 90 percent, they would end up with approximately 90 hits, and around 10 misses. This is currently not the case with the tested tools. The tools are confident in their scoring abilities, so they place most of their hits into a score of 60. This wouldn't be a problem were they to achieve 99.9999 percent accuracy, but they end up with many misses which takes them out of being anywhere close to a true 99.9999 percent accuracy. This also ends up damaging other scores leading up to 60, and it appears some, seemingly haphazardly, place their less certain misses into the 0 through 20 region. This region which is deemed less important appears to be used to increase the scores of later regions. So not only is the 60 not accurate due to lack of sensitivity, but scores leading up to 60 can get damaged. The 0 through 20 region ends up with an overly large positive accuracy, and all these combined nullify any reason to even have a scoring system, especially not one till 60.

To demonstrate the capability of having scores from 0 through 60 with more accurate map scores, a logistic regression model was generated. The results can be seen in the appendix for both Chr1 and the whole human genome. The logistic regression model was closer to the expected curve than all external tools, both for Chromosome 1, and the whole genome. 10 million read samples (10MR) were used for testing in order to obtain a smooth curve, reduce the odds of anomalies in data, while taking into consideration time and memory.

XGBoost and Deep Learning Methods were found to provide near perfect representations of what the data should look like. Notably XGBoost, had an almost 1 to 1 to expected curve output.

In order to quantify the difference between expected MapQ scores and those obtained, an Area Under the Curve (AUC) was performed between each method and the desired curve. The results, as seen in Table 4.3, re-iterate what can be visually seen on the graphs. The XGBoost model was almost 1:1 with the graph, providing the most accurate results, followed closely by the Deep Learning Neural Network Models, Logistic Regression and finally Gem3. Other models struggled, most important of which, BWA and MiniMapper2, due to their poor 0 through 20 region.

It should be noted that in places where there was no score, an average was taken between the farthest two points to predict what the score would be. XGBoost also becomes much more uncertain in the 2 to 4 MapQ region on subsequent samples, though it still maintained its smoothness in the range of 5 through to the end. This is bizarre as it is so perfectly smooth throughout the original sample, and also later mostly smooth when performed on samples of 50bp and 100bp [Appendix 2]. Regardless, it would still places amongst the best, and the model has been made available on github.

4.4.2 Reducing False Positives

MapQ score isn't as high of a priority for some tools, because obtaining a low number of false positives is deemed more important [13]. This is the reason why some scientists and users discarding anything under a score of 20, and sometimes even 30, in order to have more accurate results. Thinking like a physician explains the conundrum. They obtain saliva from their patient, it goes through quick PCR and next gen sequencing, and they are left with all these reads which are assembled to generate what should be the persons exact DNA sequence. With a lot of false positives, they will end up with the wrong read in the wrong spot. What could be a completely normal sequence could end up being a cancer marker, or worse, what could be a cancer marker would end up appearing normal. Fortunately, low false positives and high MapQ Score accuracy aren't mutually exclusive.

With Logistic Regression, the best MapQ Score accuracy was obtained, but the 20 to 60 region has more false positives than MiniMapper2 (Table 4.1). A regression model was also generated which had an undesirable 0:45 region, but its 60 score performed very accurately. Random Forests gave decent results, but nowhere near the other tools, and Support Vector Machines were doomed to fail from the start due to their strength being in classification and not regression. XGBoost however was both extremely accurate, and had a low false positive model (as evident by figure 14 and Table 4.1). Not only that, but it obtained on the upper end of true positives as well. However it did still fall slightly short of MiniMapper2's false positives. To combat this, the Logistic Regression with regularization model was merged with the XGBoost results in order to obtain superior results. This not only resulted in a model with extremely high accuracy, but thanks to its XGBoost base it also had the highest accuracy for the 20:60 region of MapQ Scores[Table 4.1], as well as the highest possible MapQ Score (Table 4.2).

The only models, which were capable of achieving a true 60, that is 99.9999 percent accuracy, are the Neural Network models. These models were capable of the highest seen accuracy, with still extremely high MapQ score accuracy.

4.4.3 Assessing results

ROC curves, comparing the number of False positive to the percent True positive is a popular way of comparing tools. The area under the curve (AUC) can be calculated to obtain a quantifiable result [10, 20, 41]. It is to be noted that a direct one to one comparison using AUC is difficult for two reasons. One, a tool with more false positives, which is an undesirable value, ends up by default with a larger AUC. This forces a comparison between identical total FP numbers, aka judging tools amongst themselves, or for a visual approximation to be performed. Once identical FP numbers are being compared, the data cannot be split into ranges. This is because, once again, the range with the most False Positives will perform the best. If, for example, a range of 20 to 60 is taken, a tool such as NN Model 1204 with only 5 false positives (seed 2) in the entire range will obtain a lower AUC, whereas a tool with much higher false positives in that range will end up with a much higher area under the curve in the range. Fortunately, GEM3 has been shown to outperform other tools in this criteria [10, 20, 41]. With the generated models being Gem Based, they happen to have the same number of False positives, so it becomes just just a matter of beating Gem3 in this method with a model that has an identical number of total False Positives. The Neural networks 1204 and 1209 do. Results can be seen in Appendix 1 Table 7.1. Left-hand underestimates as the curve is increasing, whereas Right hand overestimates. Therefore a mixed method is used to minimise bias. When this method is used, most methods perform very close to one another. XGBoost and the Neural networks perform the best of all methods, further justifying their use. Although this AUC method appears to be generally good for making quick overall comparisons, it's too general and doesn't take into consideration the way scores can be distributed in a set leading to biased results. For tools which are treading such a thin line of 'best possible' performance, there needs to be more mathematically rigorous methods for determining the best. Random Forest which is very clearly worse than Gem3 has a higher average AUC, and so these results can't be taken on their own, more parameters are needed to judge how well a tool performs.

Recall, Precision, F1 and Accuracy, are other good metrics for measuring how well a tool performs [56, 67]. As previously discussed for Table 4.4, the Neural Networks are a league above other tools in regards to F1 Score. However it's very important to show why F1 score shouldn't be the be all end all for the purposes of mapping. NN 1204v14 is included for this purpose. This is NN model 1204, but with some regularization introduced. It has a higher F1 score than many other mappers and methods, but as a scoring model it is very poor. The score distribution can be seen in the Appendix, Table 7.13. This model places all its scores between a Map Score of 10 and 11, effectively becoming a weak classification model. It has relatively poor precision as a result, but an extremely high recall, resulting in the high F1 Score. It doesn't score till 20, and its overall accuracy is extremely poor at 93 percent. For all intents and purposes, this model should never be used. All factors should be considered when judging a tool as for reads mapping scores are regression based, and so there isn't a black and white hit or miss to be considered. A high percent overall accuracy, a high accuracy in the 20 through 60, and a high maximum possible accuracy should accompany the high F1 score.

Improved accuracy and Map Scores were also attempted for varying read lengths. The arbitrarily chosen lengths were 25bp, 50bp and 100bp. With less base pairs per read, there are less base pairs to match, and therefore it becomes more difficult to place a read in a genome. Some tools just could not perform on the lower base pair reads, and all methods struggled at the 25 bp mark. For 25bp, BWA had to be run in aln mode as mem doesn't work for such small reads. Gem3 was the only other tested mapper which attempted to place reads of that length. Both Gem3 and BWA only succeeded to hit 68 percent of the reads they were presented with, showcasing the difficulties in dealing with such a small read length. Most surprising of all was how poorly the bagging methods, Random Forests and XGBoost performed. XGBoost in particular couldn't be at all accurate in the 25bp region, which is surprising given its success in other regions. Gem3 couldn't even place a million reads into the 20:60 region. Random Forests placed just over 6.5 million, but the accuracy was horrendous at under 90 percent. Logistic Regression, however, performed relatively well. 6 million hit at 99.65 percent in the 20 to 60 region. It is worth noting that if the 10 through 60 region is taken, Gem3 hits 5.7 million at 99.64 percent. That's almost as high of an accuracy with almost as many hits, albeit with less confidence.

50bp reads had much more stable results. BWA continued to perform the worst, but now XGBoost out performed all tools in its 18:60 region by having the best accuracy and the most number of hits. Here Gem3 proceeds to place many of its confident hits at a score of 60. It sacrifices some True positives, but proceeds to only have two False Positives in that range. By taking Gem3's 60, and placing them as the de facto 60 score for all tools, and having the rest be filled by a merged XGBoost + LR Model, all the benefits of each tool can be had. High accuracy in the 20:60 range, as well as the highest possible at 60.

For 100bp, BWA could be run in normal mem mode. However Gem3 far outperforms all tools when only the highest possible score of 60 is considered. The same trick as done for 50bp can be performed for 100bp to further enhance the results; Combining Gem3 with XGBoost and LR. Running the usual XGBoost, LR and XGB + LR merged methods presented better results than both BWA and Gem3 in the desirable 20:60 region. By making the Gem3 60's be taken by default, with a merged XGBoost + LR Model model being responsible for the rest, one can obtain both high hits and high accuracy not present in any other tool.

MapQ accuracy graphs for the 25bp, 50bp and 100bp read lengths can be seen in Appendix 2 With the evident gains of using Logistic Regression and XGBoost on the samples, the inevitable conclusion can be drawn that a specifically trained Neural Network can out perform all tools, as done for the 150bp samples.

In order to demonstrate the simplicity of neural network training. The base settings of Neural

Network 1209 were used to train a model for the 100bp. Only the input file was altered from 1 million 150bp reads to 1.5 million 100bp reads. No optimization was performed for the sample such as finding optimal seeds, layers, batch size, epoch, etc. Regardless, the Neural Network model outperformed all base models in the 20 through 60 region, while having a higher F1 score than the Gem3 it learned from (Appendix 2). That isn't to say this model can't do with any tuning. It is outperformed by the merged XGBoost models, and as it hasn't been optimized, it cannot hit as many as Gem3 at its best confidence level. Gem3's near 8 million with only 10 misses is more practical than the Neural Networks 70000 at 100 percent in regards to mapping. These problems can be bypassed in two ways. One is to take Gem3's 60, and use the Neural Network for the remainder so that a high 20 through 60 can also be obtained, or to spend time tuning the learning to obtain a model that does it all by itself such as 1204 and 1209 do for 150 bp reads.

4.5 Performance restrictions per method

The time taken to generate relevant results can be found in Table 7.29, and the number of samples which would be used to train each machine learning model are available in Table 7.30. Both of these tables are approximate and shouldn't be used as strict benchmark results. Random Forests and Support Vector Machines suffered from their high memory requirements in the number of samples they could learn from. Their parameters also had to be reduced for the sake of memory.In regards to time, Gem3 performed the fastest of all tools. This is relevant for the machine learning models, as they are trained off of Gem3 predictors, and therefore are only run after Gem3. Logistic Regression would require no extra processing time, as the thetas have already been generated, and Gem3 already performs the matrix multiplication in its own analysis. XGBoost being run after would still net it at under half the time it takes the next closest tool to run, MiniMapper2. Neural Networks are another story, they alone took nearly 25 minutes to run on 10 million reads. This isn't too much of a difference, and a far cry from other tested tools such as BWA, SNAP and BowTie2. It is also important to note that the generated XGBoost and Neural Network models spend some percent of time loading the model, which becomes much less pronounced when dealing with larger sample sizes, relatively.

4.6 How to further improve the scores

The most important variable in obtaining better scores would be having more, and better, parameters. Even adding something mundane such as percent GC in the read can potentially lead to improvements. For a scoring system that is working in the range of 5 decimal points, any incremental improvement can matter immensely. If it doesn't, it can quickly be discarded to save performance, but having access to more parameters is never a negative.

Another would be having access to a large amount of ram. More training examples can be run, and ram limited models could be further improved for model generation.

Having more positions to choose from would almost definitely be a major missing piece in obtaining even better results. Some tools such as BWA already output multiple positions which a read can match, however it's likely that allowing the algorithms itself choose between positions based on their predictions can greatly improve the results. It also wouldn't be too hard or too much more taxing on the hardware.

In the case of Neural Networks, the randomly generated weights per node, or the chosen 'Seeds', greatly affected the results. Being able to generates ones own weights, or having the time to test millions, would allow for a more fully realised model.

Lastly is more time to optimise and test. All tested models had more obscure parameters which could have been tested and potentially led to better results. This isn't as limiting as having more parameters or positions would be, but it is still an important step.

Making a NN for each relevantly used read length, tuned with proper seeds, (as well as other parameters), would results in the best possible method. With a mix of reads, one can call each network for each read and deal with many reads of varying lengths. This would be the most optimal method currently possible. Alternatively a parameter can be added to the neural network as the read length, but the accuracy of this most likely won't be as good as having specific networks for specific read lengths due to the broad assumptions the network would have to make. Presenting more positions for the network to decide between (base on highest Map Score according to the network) would result in the most optimal results.

From all models currently obtained, the presented 1204/1209 Neural Networks have presented the best possible accuracy at 150bp reads, however other models have been generated with more hits, but less accuracy, while still having better accuracy than other tools. A user could be presented with options. For example, a high accuracy model such as NN1209, a medium accuracy higher hits, such as NN1204, and finally a lower accuracy high hit model (of which plenty have been generated).

A simple interface could have ones generated by 1209 above the threshold just be referred to as 'hits' with the rest being given scores. The user can choose to use the rest, or as Gem3 is performing its own scoring regardless, they can proceed to see the Gem3 predicted scores for comparison as well.

Chapter 5

Conclusion

In this thesis, a range of machine learning approaches were explored in their ability to improve MapQ scores beyond those of currently available methods. Logistic Regression, Random Forests, XGBoost, Support Vector Machines and Neural Networks were all evaluated, and the best results were used to procure the best feasible models. Previous methods of assigning scores to mapped reads (e.g. Bowtie2, BWA-MEM or SNAP) resulted in relatively inaccurate scores, with relatively high false positives both in the desired 20 through 60 MapQ score range, as well as in their best case. The methods had a tendency of overestimating the MapQ quality of reads, all at different inconsistent extents. This leads to inconsistencies in overall score predictions, as well as leading to more false positives, which in turn results in many more errors in the genome that is being generated. This damages the subsequent and final step of the genome re-construction pipeline , where the generated DNA is sent to be analysed.

Ultimately, a neural network was developed to predict MapQ scores for reads of 150bp – without loss of generality – based on predictors obtained from the Gem3 mapper. This model had better accuracy in the best case – 20 through 60 region – and the overall. It also has more accurate MapQ scores, and the highest F1 score, thus outperforming previous methods in all desired criteria. In some tested samples, an accuracy of 100 percent was obtained with over 60 percent of a set of tested samples – unseen in any other tools. Given more and better predictors, the tool can theoretically be further improved to more consistently hit a 100 percent accuracy, with a wider net of true positives. Similarly, XGBoost showed immense promise with the highest MapQ score accuracy, but with more false positives. This can mostly be mitigated by combining the scores with the logistic regression results that Gem3 calculates. XGBoost is faster, and models are much more easily trained than neural networks, providing users with a fast and easy way to improve their map scores.

Although the generated models outperform current methods for the test case of 150bp, better models can still be trained. To start with, these would require more, and better, predictors to be generated, so that the models can find better patterns to discern between. Next, having the Gem3 mapper run longer so that it can output more mapping locations, as BWA does, and sending those to be chosen by the model could result in even better accuracy. Finally, the parameters used were optimised given the time available as well as the 150bp read goal. More parameter testing for possible small improvements, as well as generating models for each used base pair size, would allow for a powerful tool that is also flexible.

As a result of this work, all the code and tools to train models is publicly available. It can be used to generate models which adaptively calibrate MapQ scores to any genome of any species.Many sequencing-data analysis pipelines based on non-model organism can see a potential improvement in having custom MapQ models tailored for them. This will allow for improved accuracy and specificity flexible around the experiment goals, read lengths, and samples. The model can also learn from, and become specific to, reads of any specific length; providing an increase in accuracy for a wide range of tools and mapping methods used today. Furthermore, the computational cost of generating MapQ scores using these models is negligible and does not add any significant processing time when combined with any pre-existing analysis pipeline.

I've thoroughly enjoyed my work on this project and hope I have contributed, however small, to the field. I anticipate seeing more machine learning being introduced to the fields of bioinformatics and medicine, and I wish to continue to expand my knowledge and expertise so that I can be a pioneer as these fields continue to grow.

Chapter 6

References

1. Wetterstrand, K.A. (2018) *DNA Sequencing Costs: Data from the NHGRI Genome Sequencing Program (GSP)* Retrieved from www.genome.gov/sequencingcostsdata

2. Breastcancer.org. (2018). *Genetics: Breast Cancer Risk Factors.* Retrieved from https://www.breastcancer.org/risk/factors/genetics

3. National Human Genome Research Institute (NHGRI). (2010). *The Human Genome Project Completion: Frequently Asked Questions.* Retrieved from https://www.genome.gov/11006943/human-genome-project-completion-frequently-asked-questions/

4. Ekblom, R. & Wolf, J.B.W. (2014). A field guide to whole-genome sequencing, assembly and annotation. *Evolutionary Applications*, 7(9), 1026-1042. doi: 10.1111/eva.12178.

5. Lischer, H. E. L., & Shimizu, K. K. (2017). Reference-guided de novo assembly approach improves genome reconstruction for related species. *BMC Bioinformatics* , 18, 474. doi: http://doi.org/10.1186/s12859-017-1911-6

6. Keats, B.J.B. & Sherman, S.L. (2013) Emery and Rimoin's Principles and Practice of Medical Genetics. *Academic Press* , 13, 1-12. doi: https://doi.org/10.1016/B978-0-12-383834-6.00015-X.

7. Roser, M. & Ortiz-Ospina, E. (2017). World Population Growth. *Our World in Data* . Retrieved from https://ourworldindata.org/world-population-growth

8. Heath, N. (2018). What is machine learning? Everything you need to know. *ZDNet.* Retrieved from: https://www.zdnet.com/article/what-is-machine-learning-everything-you-need-to-know/

9. Watson, J.D. & Crick, F.H. (1953). Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *Nature.* 1974, 248-765.

10. Marco-Sola, S., Sammeth, M., Guigó, R., & Ribeca, P. (2012). The GEM mapper: fast, accurate and versatile alignment by filtration. *Nature Methods* , 9(12), 1185–1188. doi: http://doi.org/10.1038/nmeth.2221

11. Deng, N., Zhou, H., Fan, H., & Yuan, Y. (2017). Single nucleotide polymorphisms and cancer susceptibility. *Oncotarget* , 8(66), 110635–110649. doi: http://doi.org/10.18632/oncotarget.22372

12. Holtgrewe, M. (2010). Mason – a read simulator for second generation sequencing data.*Technical Report TR-B-10-06.* Institut für Mathematik und Informatik, Freie Universität Berlin.

13. Van der Auwera, G. A., Carneiro, M. O., Hartl, C., Poplin, R., del Angel, G., Levy-Moonshine, A., Jordan, T., Shakir, K., Roazen, D., Thibault, J., Banks, E., Garimella, K.V., Altshuler, D., Gabriel, S. & DePristo, M. A. (2013). From FastQ data to high confidence variant calls: the Genome Analysis Toolkit best practices pipeline. *Current Protocols in Bioinformatics*, 11(1110), 11.10.1–11.10.33. doi: http://doi.org/10.1002/0471250953.bi1110s43

14. Brownlee, J. (2018). Logistic Regression for Machine Learning. *Machine Learning Mastery* . Retrieved from https://machinelearningmastery.com/logistic-regression-for-machine-learning/

15. Bradnam, K. (2018). Understanding MAPQ scores in SAM files: does 37 = 42?. *ACGT* . Retrieved from http://www.acgt.me/blog/2014/12/16/understanding-mapq-scores-in-sam-files-does-37-42

16. Liu, L., Li, Yinhu., Li, Siliang., , Yinhu Li, Siliang Li, (2012). Comparison of Next-Generation Sequencing Systems. *Journal of Biomedicine and Biotechnology.* 2012. doi: https://doi.org/10.1155/2012/251364.

17. Quail, M.A., Smith, M., Coupland, P., Otto, T.D., Harris, S.R., Connor, T.R., Bertoni, A., Swerdlow, H.P. & Gu, Y. (2012) A tale of three next generation sequencing platforms: comparison of Ion Torrent, Pacific Biosciences and Illumina MiSeq sequencers.*BMC Genomics.* 13-341. doi: 10.1186/1471-2164-13-341.

18. Brown, T.A. (2002). *Genomes 2nd edition.* Chapter 5, Mapping Genomes. Retrieved from: https://www.ncbi.nlm.nih.gov/books/NBK21116/

19. Ye, H., Meehan, J., Tong, W. & Hong, H. (2015). Alignment of Short Reads: A Crucial Step for Application of Next-Generation Sequencing Data in Precision Medicine. *Ed. Pharmaceutics.* 7(4), 523-541. doi: 10.3390/pharmaceutics7040523.

20. Lee, H., Lee, K., Lee, T., Park, D., Chung, J., Lee, C., Park, W. & Son, D. (2018) Performance evaluation method for read mapping tool in clinical panel sequencing. *Genes & genomics.* 40(2). 189-197.

21. Healthcare IT News. (2018).*Microsoft vs. Google: Hot start to 2018 for investments in machine learning and precision medicine startups* . [online] Retrieved from http://www.healthcareitnews.com/news/microsoft-vs-google-hot-start-2018-investments-machine-learning-and-precision-medicine-startups

22. Mercer, C. (2018). 10 tech giants investing big in artificial intelligence. *Techworld* . Retrieved from https://www.techworld.com/picture-gallery/data/tech-giants-investing-in-artificial-intelligence-3629737/

23. Andrews, S. (2018). MAPQ values are really useful but their implementation is a mess. *Sequencing.qcfail.com.* Retrieved from https://sequencing.qcfail.com/articles/mapq-values-are-really-useful-but-their-implementation-is-a-mess/

24. Gibbs, S. (2018). Google's Pixel 2 and Pixel 2 XL: an AI-infused challenge to the iPhone. *The Guardian* . Retrieved from https://www.theguardian.com/technology/2017/oct/04/pixel-2-pixel-2-xl-google-apple-iphone-ai-smartphone-64gb-storage-speakers-12-megapixel-cameras

25. Donges, N. (2018). The Random Forest Algorithm – Towards Data Science. *Towards Data Science.* Retrieved from
https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd

26. Chen, T. & Guestrion, C. (2016). XGBoost: A Scalable Tree Boosting System. *In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16).* 785-794. doi: https://doi.org/10.1145/2939672.2939785

27. Hearts, A. M. (1998). Support Vector Machines. *IEEE Intelligent Systems* . 13(4), 18-28. doi: http://dx.doi.org/10.1109/5254.708428

28. Nielsen, M. (2015). Neural Networks and Deep Learning. *Determination Press.* Retrieved from http://neuralnetworksanddeeplearning.com/

29. Le, J. (2018). A Tour of The Top 10 Algorithms for Machine Learning Newbies. *Towards Data Science.* Retrieved from https://towardsdatascience.com/a-tour-of-the-top-10-algorithms-for-machine-learning-newbies-dde4edffae11

30. Morales, M. (2018). How Is Deep Learning Revolutionizing Artificial Intelligence. *Channels.theinnovationenterprise.com* Retrieved from:
https://channels.theinnovationenterprise.com/articles/how-is-deep-learning-revolutionizing-artificial-intelligence

31. Van Rossum, G. (1995) Python Reference Version 2.7, Technical Report CS-R9526. *Centrum voor Wiskunde en Informatica (CWI).* Retrieved from http://www.python.org

32. Eaton J.W., Bateman D., Hauberg S., Wehbring R. (2014). GNU Octave version 3.8.1 manual: a high-level interactive language for numerical computations. *CreateSpace Independent Publishing Platform.* Retrieved from http://www.gnu.org/software/octave/doc/interpreter/

33. Ng, A. (2018). Machine Learning — Coursera. *Coursera.* Retrieved from
https://www.coursera.org/learn/machine-learning

34. RStudio (2012). RStudio: Integrated development environment for R . *[Computer software]* . Retrieved from http://www.rstudio.org/

35. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jozefowicz, R., Jia, Y., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Schuster, M., Monga, R., Moore, S., Murray, D., Olah, C., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y. & Zheng, X. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems.* Retrieved from tensorflow.org.

36. Keras.io. (2018). *FAQ - Keras Documentation.* Retrieved from https://keras.io/getting-started/faq/#how-can-i-obtain-reproducible-results-using-keras-during-development

37. Brownlee, J. (2018). Difference Between Classification and Regression in Machine Learning. *Machine Learning Mastery.* Retrieved from
https://machinelearningmastery.com/classification-versus-regression-in-machine-learning/

38. Tech Differences. (2018). Difference Between Linear and Logistic Regression (with Comparison Chart). *Tech Differences.* Retrieved from https://techdifferences.com/difference-between-linear-and-logistic-regression.html.

39. Ng, A. (2018). Classification - Logistic Regression — Coursera. *Coursera.* Retrieved from https://www.coursera.org/lecture/machine-learning/classification-wlPeP

40. Nagpal, A. (2017). Decision Tree Ensembles- Bagging and Boosting. *Towards Data Science.* Retrieved from https://towardsdatascience.com/decision-tree-ensembles-bagging-and-boosting-266a8ba60fd9

41. Laurie, S., Fernandez-Callejo, M., Marco-Sola, S., Trotta, J., Camps, J., Chacón, A., Espinosa, A., Gut, M., Heath, S., & Beltran, S. (2016). From Wet-Lab to Variations: Concordance and Speed of Bioinformatics Pipelines for Whole Genome and Whole Exome Sequencing. *Human Mutation.* 37(12), 1263–1271. http://doi.org/10.1002/humu.23114

42. Jin, Y. (2018). Tree Boosting With XGBoost — Why Does XGBoost Win "Every" Machine Learning Competition?. *Medium.* Retrieved from https://medium.com/syncedreview/tree-boosting-with-xgboost-why-does-xgboost-win-every-machine-learning-competition-ca8034c0b283

43. Z, Z. (2018). What is better: gradient-boosted trees, or a random forest? - FastML. *Fastml.com.* Retrieved from http://fastml.com/what-is-better-gradient-boosted-trees-or-random-forest/

44. Yu, H. & Kim, S. (n.d.). SVM Tutorial: Classification, Regression, and Ranking. *Pdfs.semanticscholar.org* Retrieved from https://pdfs.semanticscholar.org/cbc3/d8b04d37b2d4155f081cd423380220a91f13.pdf

45. Ibm.com. (2012). *How SVM works.* Retrieved from https://www.ibm.com /support/knowledgecenter/en/SS3RA7_15.0.0/com.ibm.spss.modeler.help/svm_howwork.htm

46. Krishna A. M. (2018). *Large-Scale Support Vector Machines: Algorithms and Theory.* Retrieved from https://www.researchgate.net/publication/251892233_Large-Scale_Support_Vector_Machines_Algorithms_and_Theory

47. Maji, S., Berg, A.C. & Malik, J. (2008). *Classification using Intersection Kernel Support Vector Machines is Efficient.* Retrieved from https://people.cs.umass.edu/ smaji/papers/iksvm-cvpr08.pdf

48. Heaton, J. (2008). Introduction to Neural Networks for Java, 2nd Edition. *Heaton Research, Inc.* isbn : 1604390085, 9781604390087

49. Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. *Neural Networks.* 61, 85–117. doi: 10.1016/j.neunet.2014.09.003.

50. Illumina.com. (2018). *Estimating Sequencing Coverage.* Retrieved from https://www.illumina.com/documents/products/technotes/technote_coverage_calculation.pdf

51. Illumina.com. (2018). *Coverage depth recommendations.* Retrieved from https://www.illumina.com/science/education/sequencing-coverage.html

52. Cai, E. (2014). Machine Learning Lesson of the Day – Overfitting and Underfitting. *The Chemical Statistician.* Retrieved from https://chemicalstatistician.wordpress.com/2014/03/19/machine-learning-lesson-of-the-day-overfitting-and-underfitting/

53. Domke, J. (2008). Why does regularization work?. *JustinDomke.* Retrieved from https://justindomke.wordpress.com/2008/12/12/why-does-regularization-work/

54. Soren, D. (2017). How regularization can improve your machine learning algorithms - Practical Artificial Intelligence. *Practicalai.* Retrieved from https://www.practicalai.io/how-regularization-can-improve-your-machine-learning-algorithms/

55. S. Sarle, W. (2002). comp.ai.neural-nets FAQ Part 2 of 7: Learning. *Faqs.org* . Retrieved from http://www.faqs.org/faqs/ai-faq/neural-nets/part2/

56. Joshi, R. (2016). Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures. *exsilio.* Retrieved from http://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/

57. The SAM/BAM Format Specification Working Group (2018). *Sequence Alignment/Map Format Specification.* Retrieved from https://samtools.github.io/hts-specs/SAMv1.pdf

58. Illumina.com. (2018).*Introduction to NGS.* Retrieved from https://www.illumina.com/science/technology/next-generation-sequencing.html

59. Li, H. (2013) Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. *Genomics.* arXiv: 1303.3997.

60. Li, H. (2018) Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics.* doi: https://doi.org/10.1093/bioinformatics/bty191

61. Zaharia, M., Bolosky, W.J., Curtis, K., Fox, A., Patterson, D., Shenker, S., Sotica, I., Karp, R.M., & Stittler, T. (2011). Faster and More Accurate Sequence Alignment with SNAP. *Genomics.* arXiv : 1111.5572

62. Langmead, B. & Salzberg, S.L. (2012). Fast gapped-read alignment with Bowtie 2. *Nature methods.* 9(4), 357-359 doi: 10.1038/nmeth.1923.

63. Free Software Foundation (2007). Bash (3.2.48). *[Unix shell program].* Retrieved from http://ftp.gnu.org/gnu/bash/bash-3.2.48.tar.gz

64. Breiman, L., Liaw, A. & Wiener, M. (2018). *Breiman and Cutler's Random Forests for Classification and Regression.* Retrieved from https://cran.r-project.org/web/packages/randomForest/randomForest.pdf

65. Chen T., Tong, H., Benesty, M. & Khotilovich, V. (2018). *Extreme Gradient Boosting* . Retrieved from https://cran.r-project.org/web/packages/xgboost/xgboost.pdf

66. Meyer, D. (2017). *Support Vector Machines The Interface to libsvm in package e1071.* Retrieved from https://cran.r-project.org/web/packages/e1071/vignettes/svmdoc.pdf

67. Mishra, A. (2018). Metrics to Evaluate your Machine Learning Algorithm. *Towards Data Science* Retrieved from https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

Chapter 7

Appendix 1

7.1 150 bp Reads

7.1.1 Chromosome 1 Results

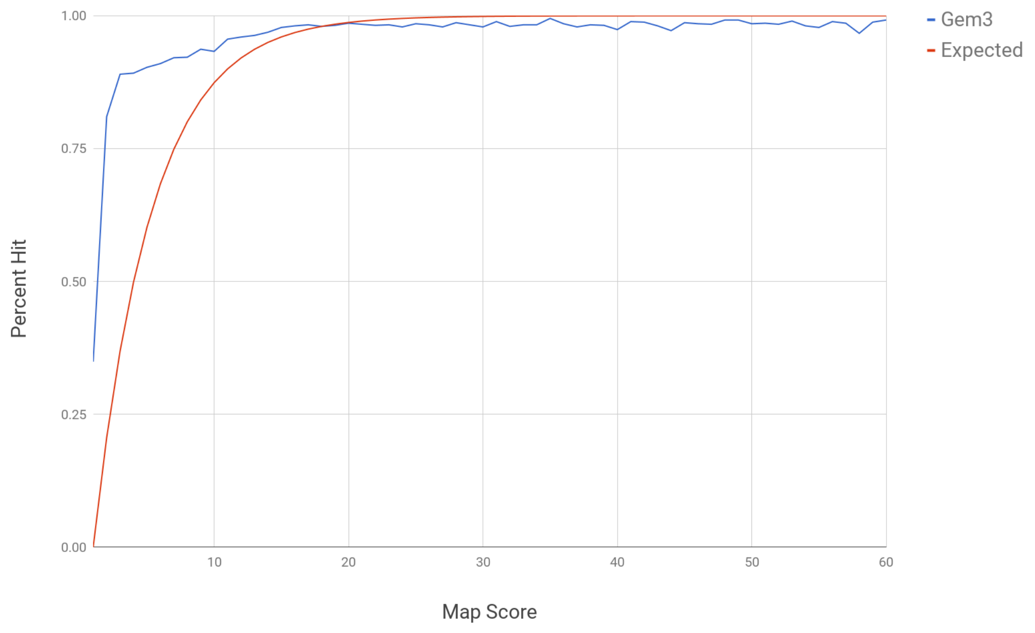

Figure 7.1: Resulting percent hit per map score for Gem3 mapping 10 million 150bp chromosome 1 reads alongside the expected percent hit. No base pair leeway was given. No seed was used for this test set.

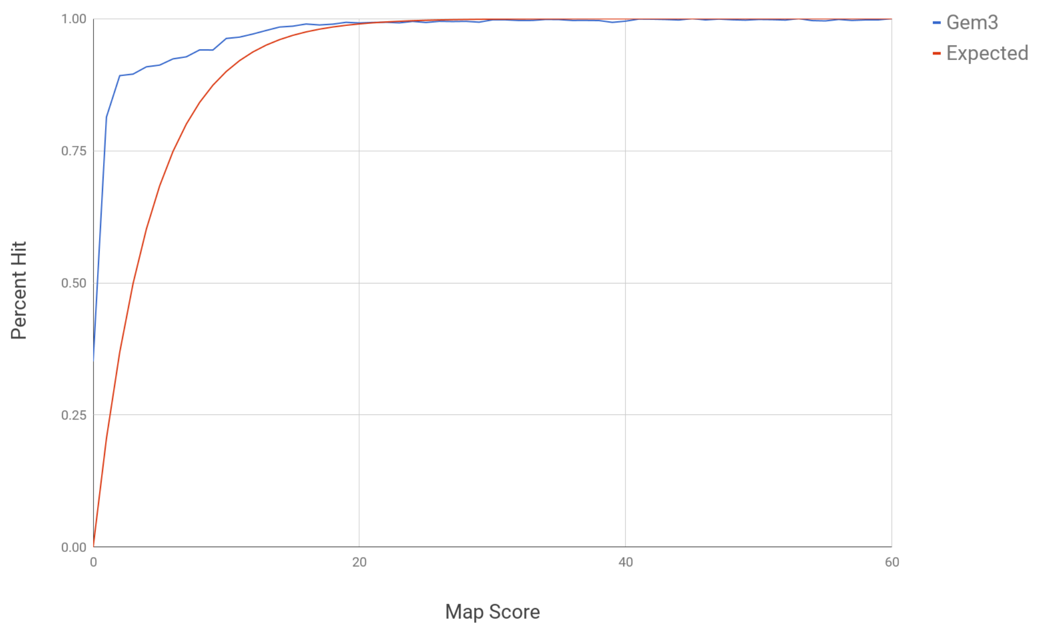

Figure 7.2: Resulting percent hit per map score for Gem3 mapping 10 million 150bp chromosome 1 reads alongside the expected percent hit. Mapping was given a 10 base pair leeway. No seed was used for this test set.

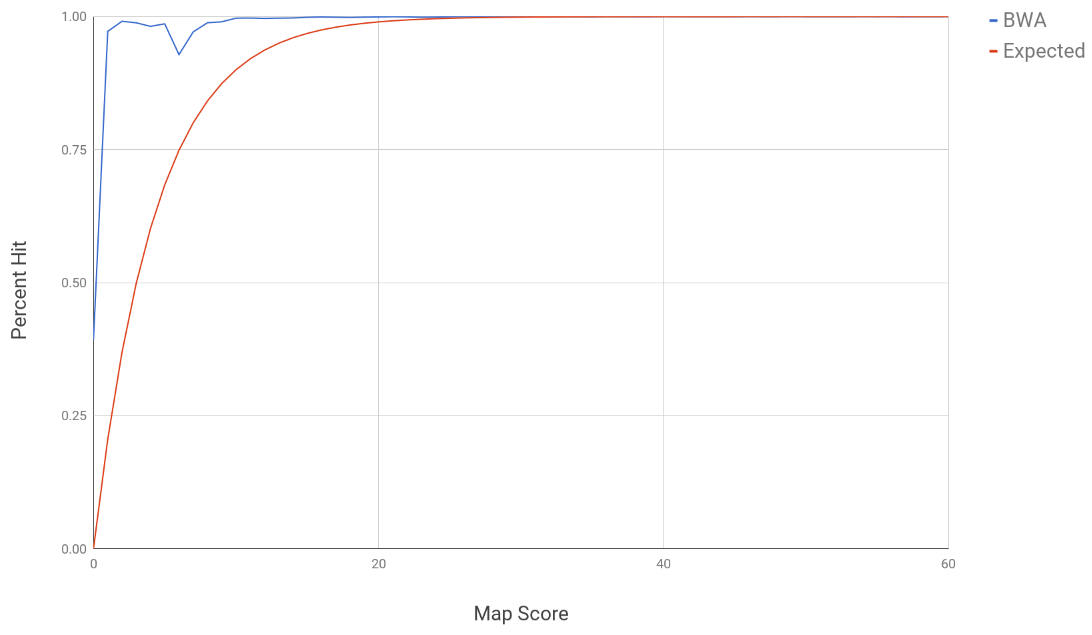

Figure 7.3: Resulting percent hit per map score for bwa-mem mapping 10 million 150 bp chromosome 1 reads alongside the expected percent hit. Mapping was given a 10 base pair leeway. No seed was used for this test set.

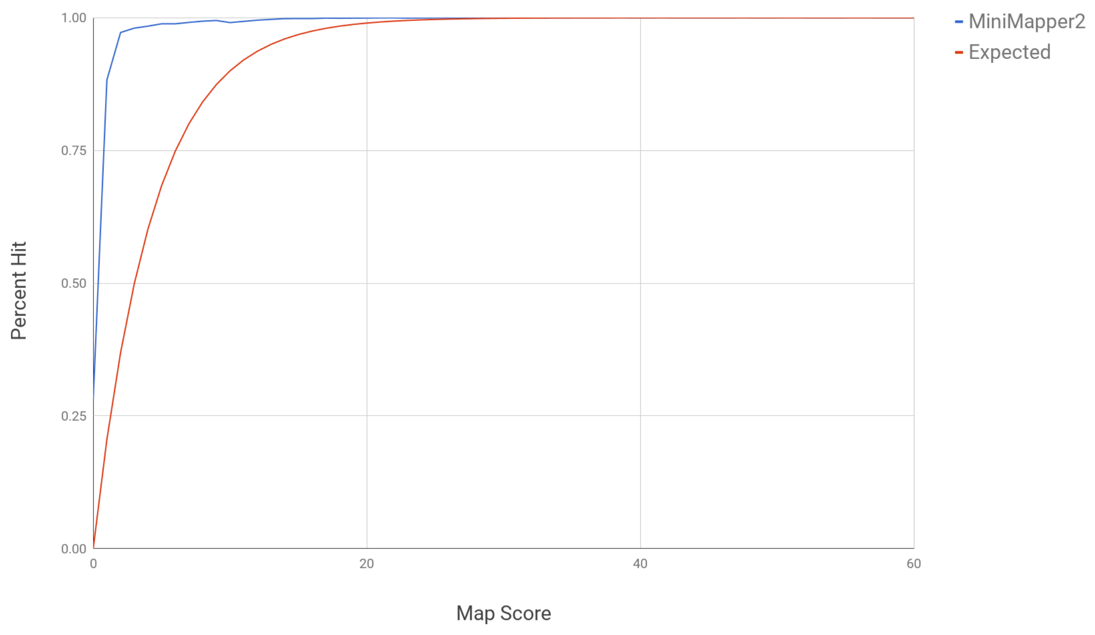

Figure 7.4: Resulting percent hit per map score for MiniMapper2 mapping 10 million 150 bp chromosome 1 reads alongside the expected percent hit. Mapping was given a 10 base pair leeway. No seed was used for this test set.

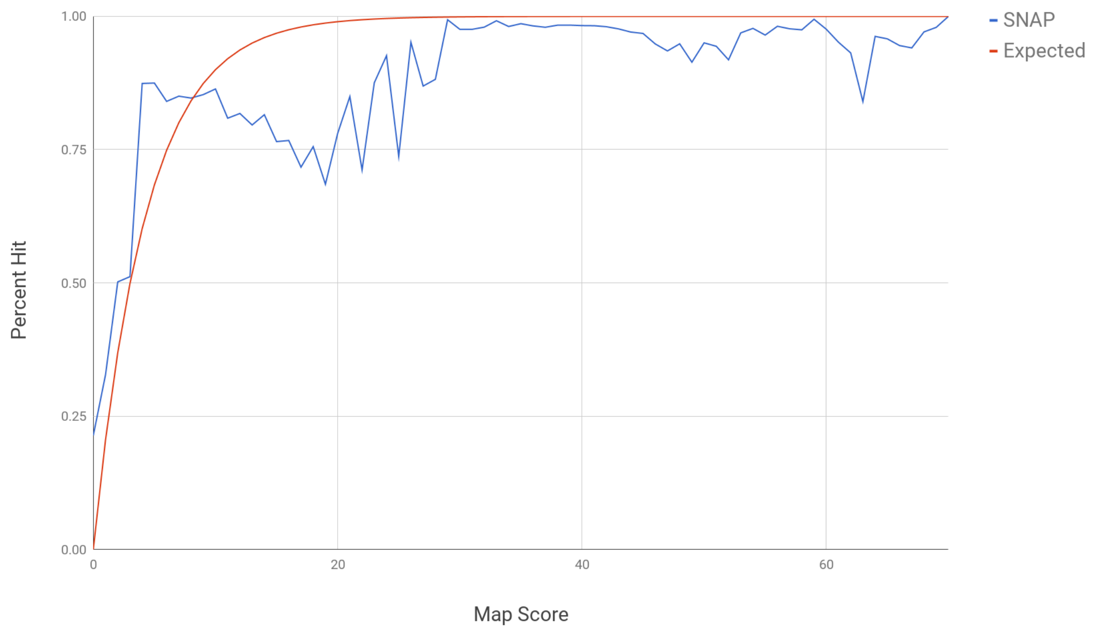

Figure 7.5: Resulting percent hit per map score for SNAP mapping 10 million 150 bp chromosome 1 reads alongside the expected percent hit. Mapping was given a 10 base pair leeway. No seed was used for this test set.

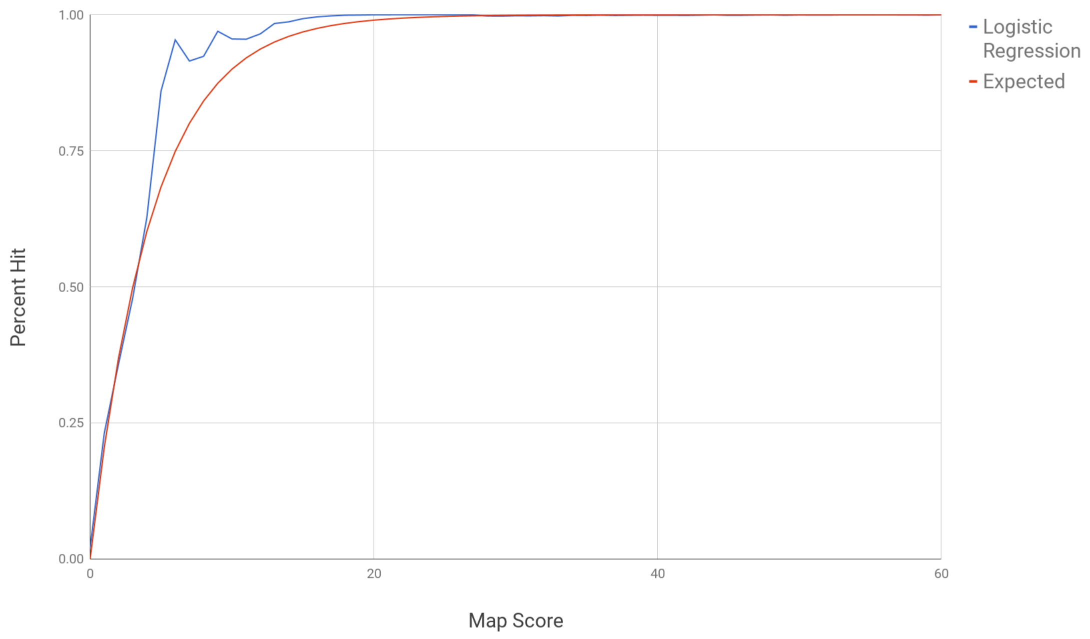

Figure 7.6: Resulting percent hit per map score for a generated Logistic Regression model utilising predictors obtained from a separate 80 million Gem3 mapped result, mapping 10 million 150 bp chromosome 1 reads versus the expected percent hit. Mapping was given a 10 base pair leeway. No seed was used for this test set.

7.1.2 Full Human Genome Results

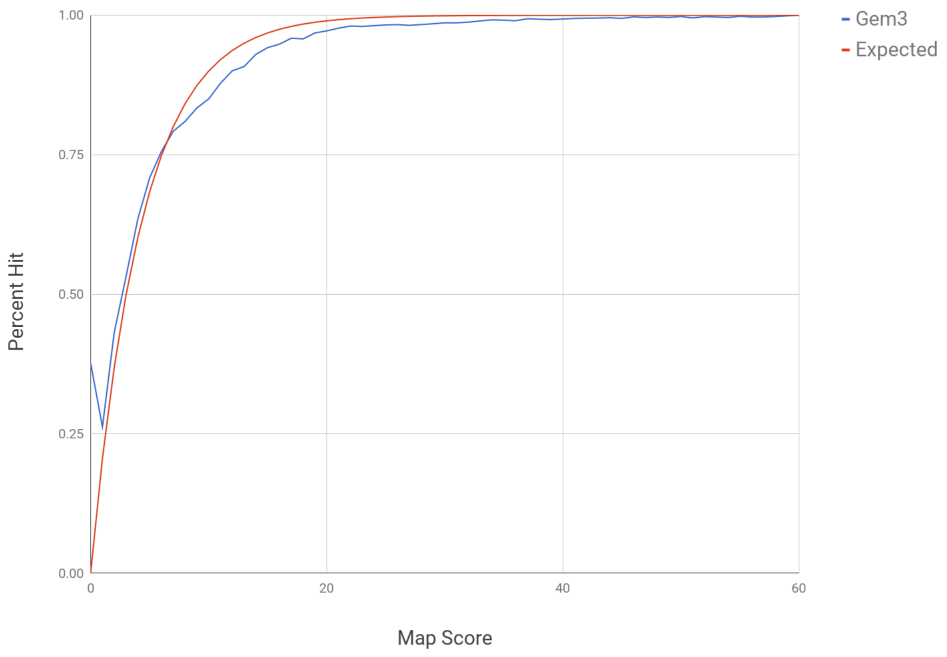

Figure 7.7: Resulting percent hit per map score for Gem3 mapping 10 million 150 bp HG38.12 reads, generated using mason with a seed of 2, alongside the expected percent hit with a 10 base pair leeway. Data in Table 7.2.

Table 7.1: Area Under Curve of Methods using data available in Appendix 1. Sorted by smallest to largest Mixed Method Score using no seed on all.

	Area Under the Curve for Figure 7.2		
	Left Hand Method	Right Hand Method	Averaged Method
Gem3	654309	686207	670258
LR w Reg	667803	678258	673030
LR	669265	679805	674535
Random Forest	673854	679189	676522
NN 1204	669848	683280	676564
NN 1209	670424	682948	676686
XGBoost	673557	682073	677815

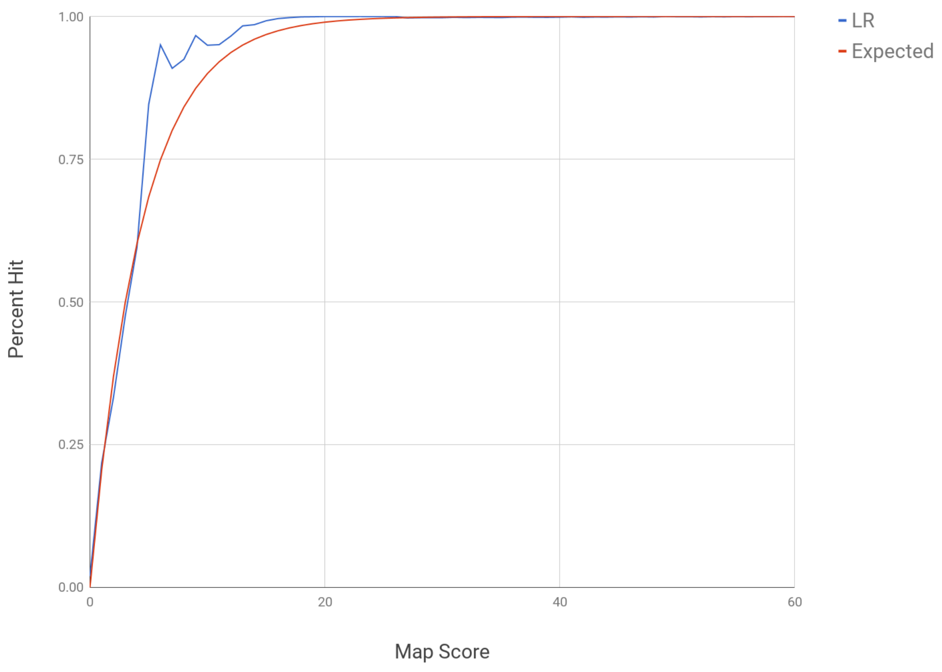

Figure 7.8: Resulting percent hit per map score for a Logistic Regression Model generated using 100 million HG38.12 mason generated reads. It was used to estimate on 10 million 150 bp HG38.12 reads, generated using mason with a seed of 2, graphed alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Table 7.7

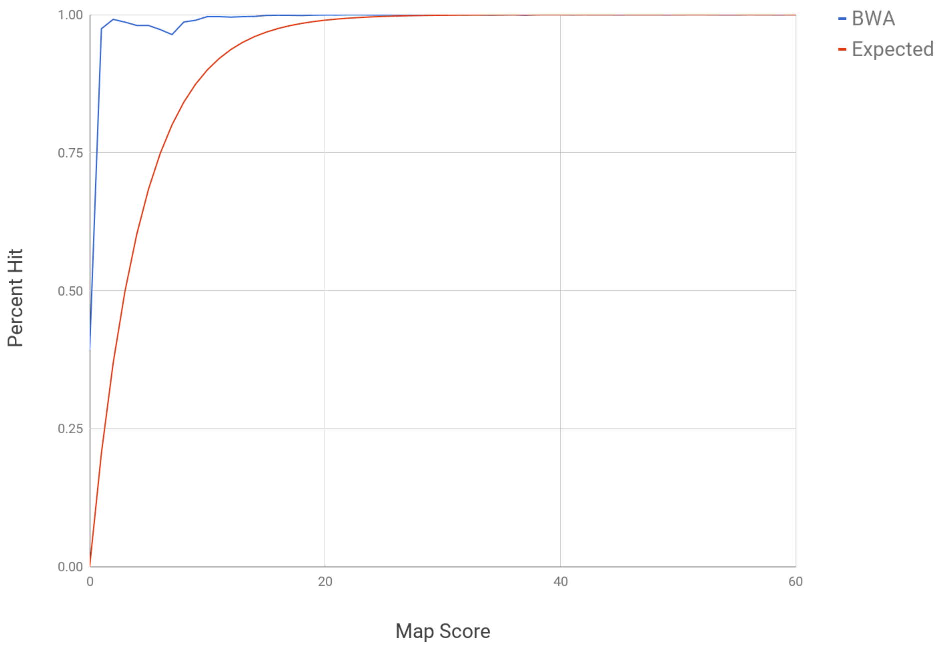

Figure 7.9: Resulting percent hit per map score for BWA mapping 10 million 150 bp HG38 reads, generated using mason with a seed of 2, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Table 7.3

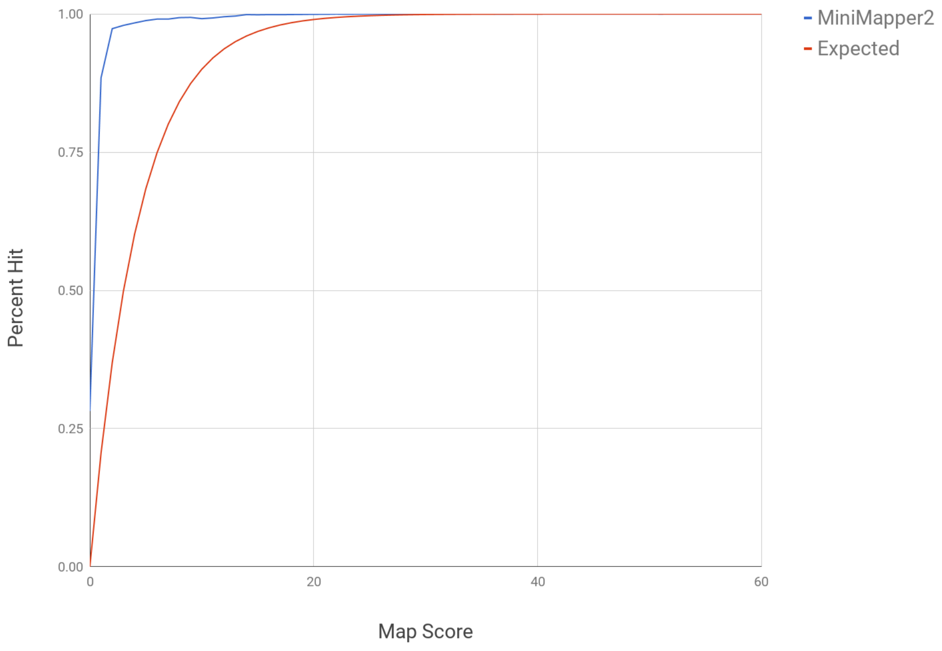

Figure 7.10: Resulting percent hit per map score for MiniMap2 mapping 10 million 150 bp HG38 reads, generated using mason with a seed of 2, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Table 7.4

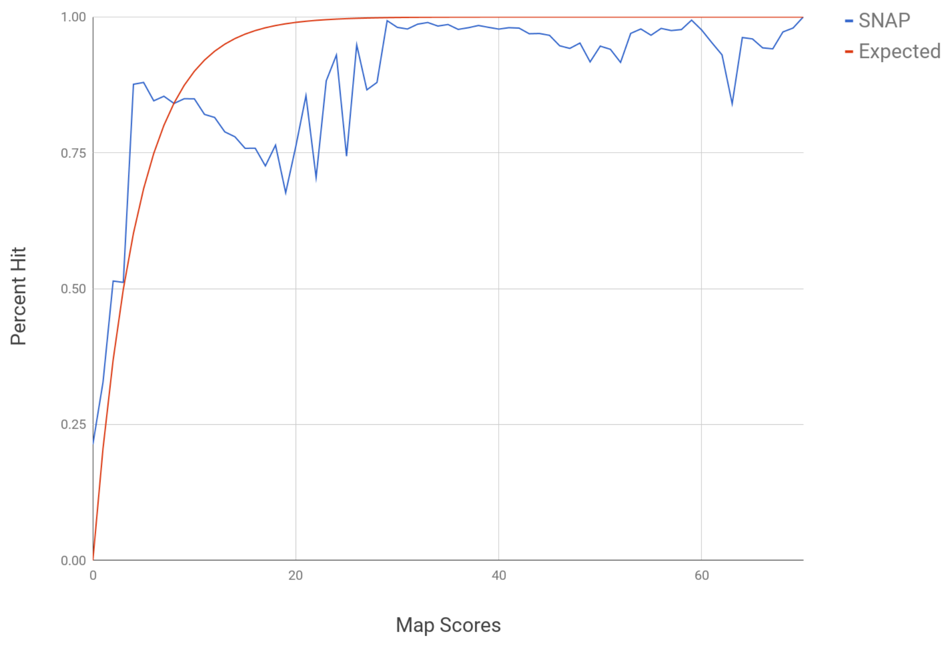

Figure 7.11: Resulting percent hit per map score for SNAP mapping 10 million 150 bp HG38.12 reads, generated using mason with a seed of 1, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Table 7.5

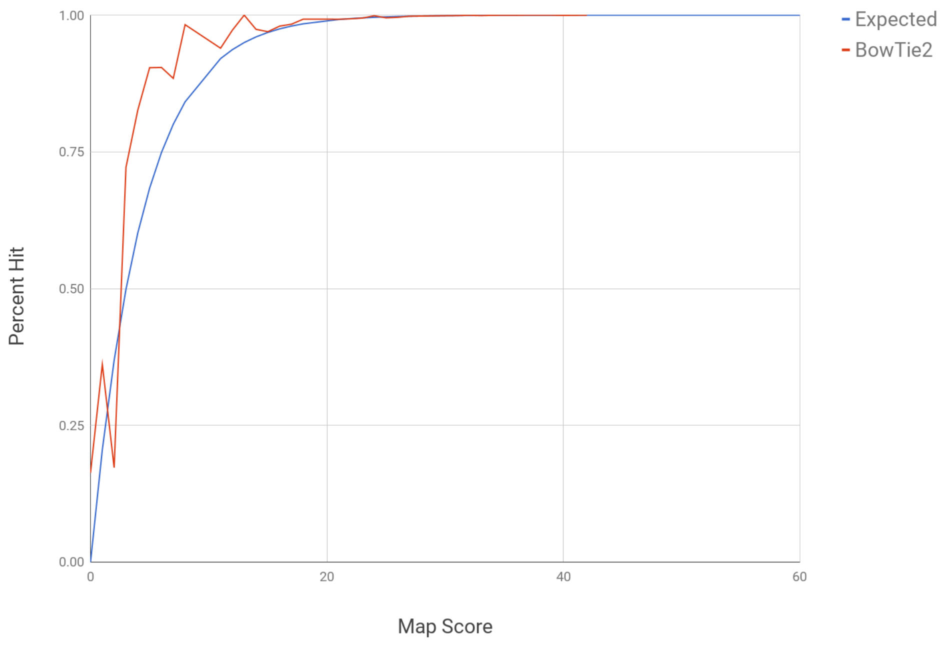

Figure 7.12: Resulting percent hit per map score for BowTie2 mapping 10 million 150 bp HG38.12 reads, generated using mason with a seed of 2, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Table 7.6

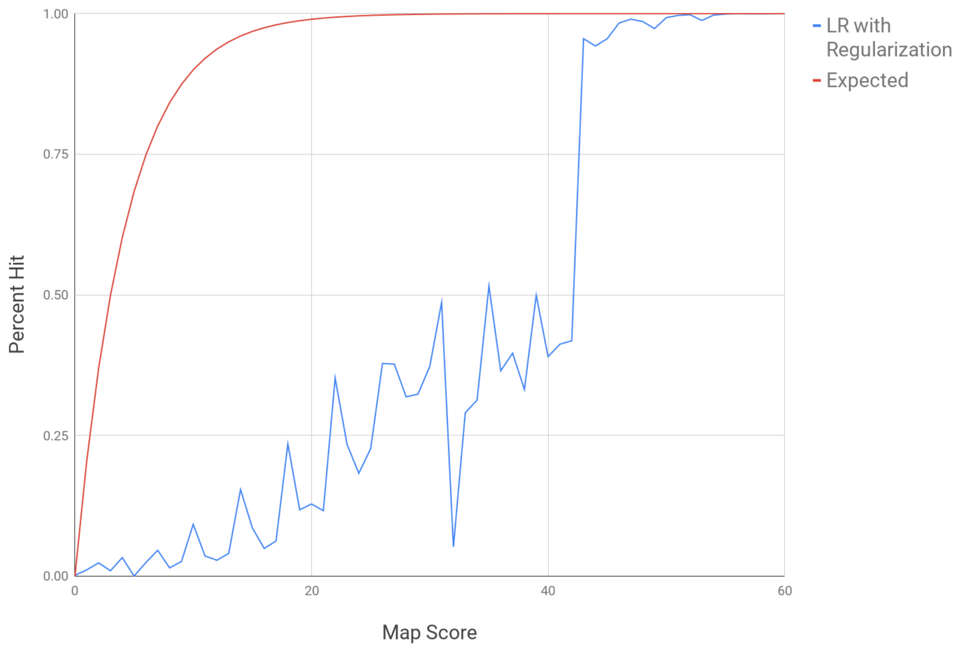

Figure 7.13: Resulting percent hit per map score for a Logistic Regression Model with Regularization generated using 75 million HG38.12 mason generated reads with base seed used to train the model. It was used to estimate on 10 million 150 bp HG38.12 reads, generated using mason with a seed of 2, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Table 7.10

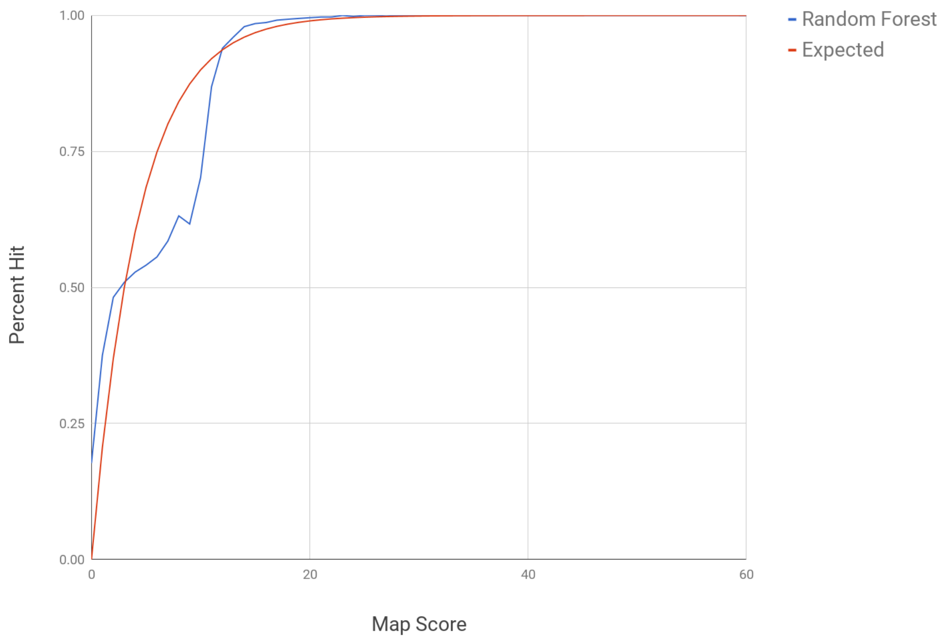

Figure 7.14: Resulting percent hit per map score for a Random Forest Model generated using 1 million HG38.12 mason generated reads with base seed, used to train the model. It was used to estimate on 10 million 150 bp HG38.12 reads, generated using mason with a seed of 2, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Table 7.8

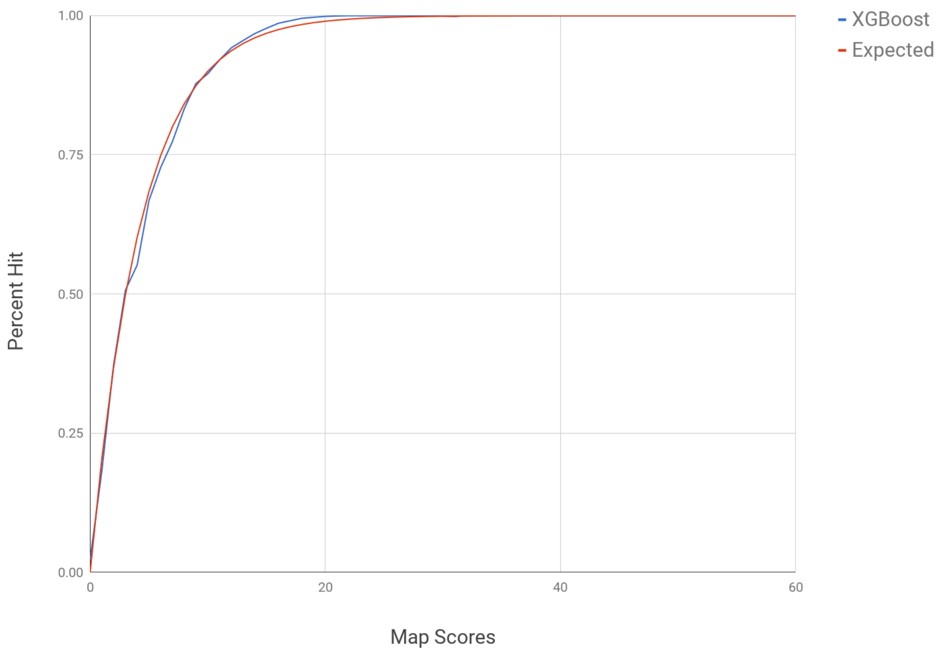

Figure 7.15: Resulting percent hit per map score for a XGBoost Model created using 10 million HG38.12 mason generated reads with base seed, used to train the model. It was used to estimate on 10 million 150 bp HG38.12 reads, generated using mason with a seed of 2, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Table 7.9

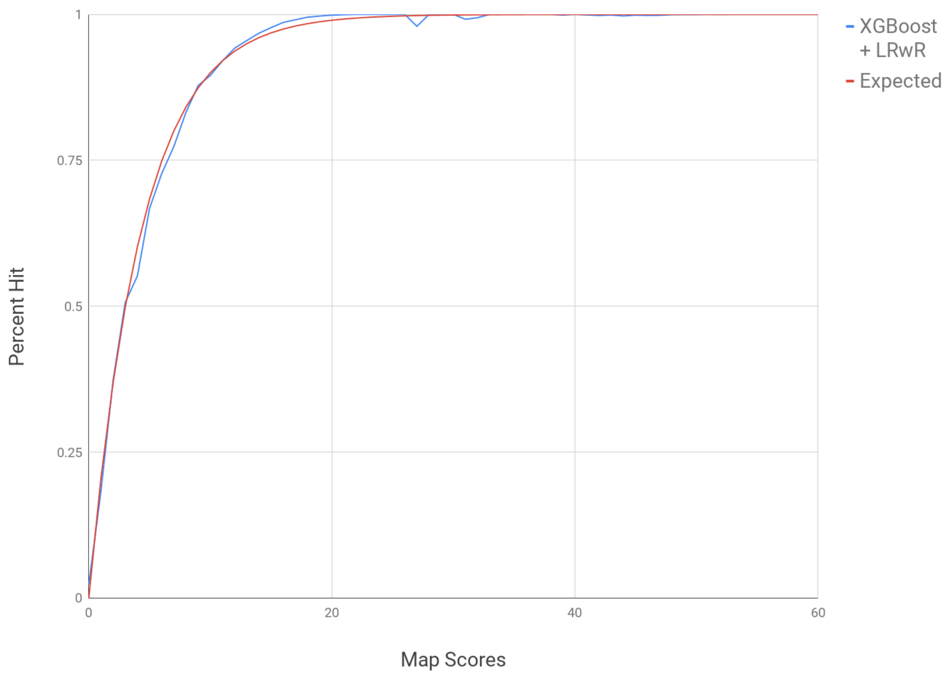

Figure 7.16: Resulting percent hit per map score for a combined Model of XGBoost with Logistic Regression with Regularization, created using 10 million HG38.12 mason generated reads with base seed, used to train the model. It was used to estimate on 10 million 150 bp HG38.12 reads, generated using mason with a seed of 2, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data available in table 7.14

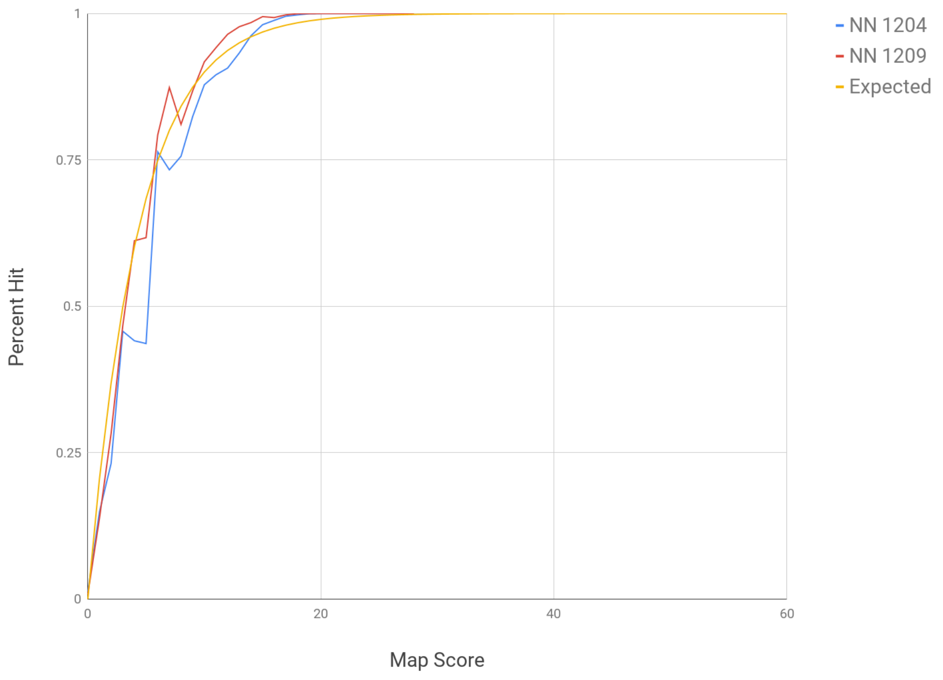

Figure 7.17: Resulting percent hit per map score for the Neural Networks 1204 and 1209, created using 1 million HG38.12 mason generated reads with base seed, and a seed of 66 used to train the model. It was used to estimate on 10 million 150 bp HG38.12 reads, generated using mason with a seed of 2, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Tables 7.11 and 7.12

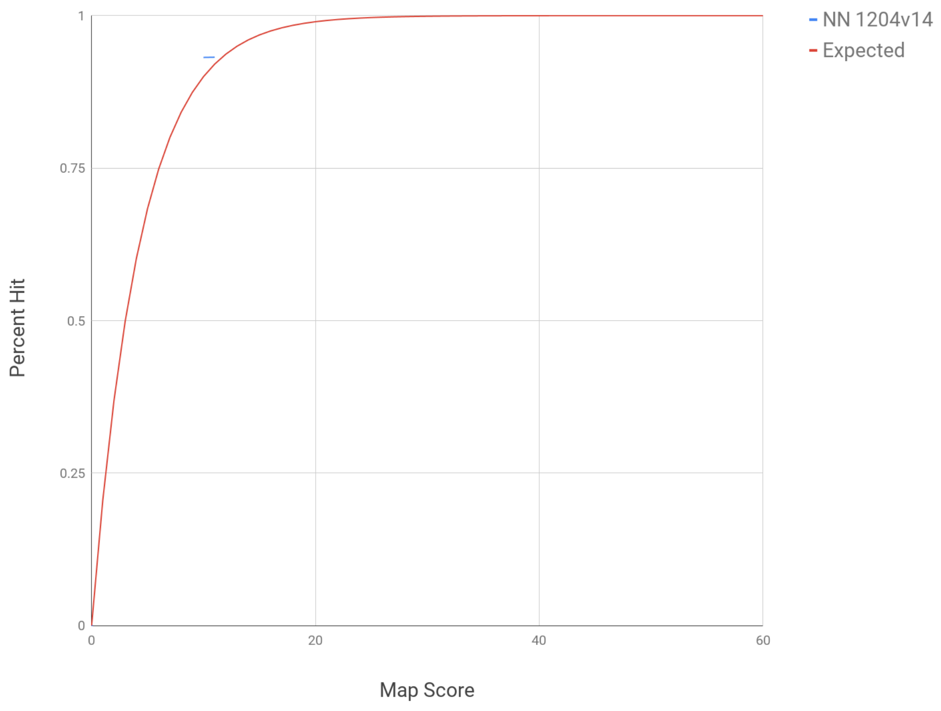

Figure 7.18: Resulting percent hit per map score for the Neural Networks 1204 with regularization version 7. It was generated using 1 million HG38.12 mason generated reads with base seed, and a seed of 66 used to train the model. The model was used to estimate on 10 million 150 bp HG38.12 reads, generated using mason with a seed of 2, alongside the expected percent hit. Mapping was given a 10 base pair leeway. Data in Tables 7.13.

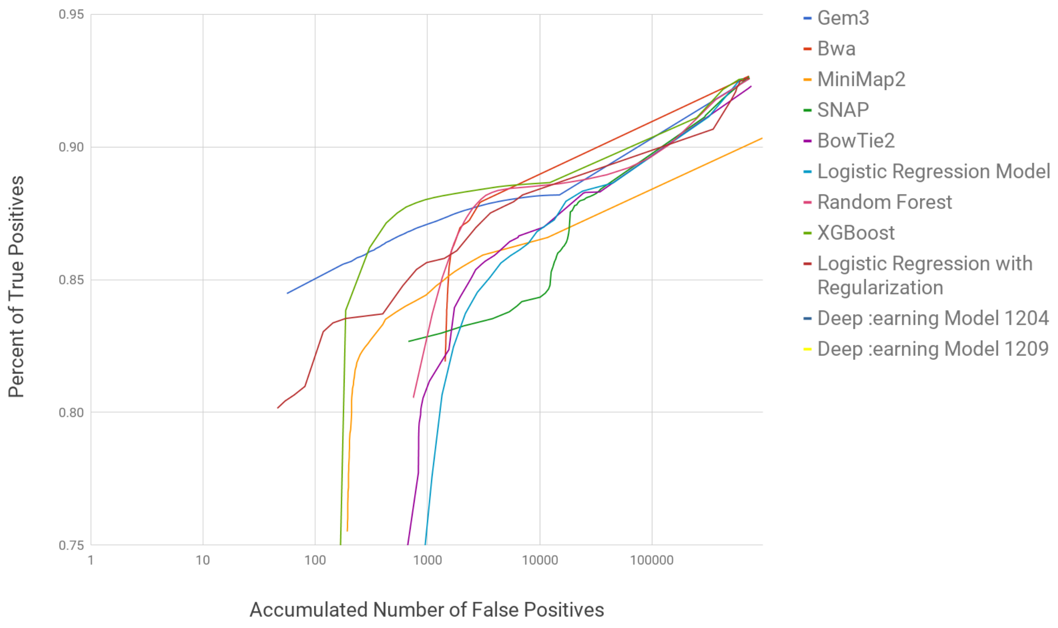

Figure 7.19: Total percent of True positives versus the total number of False positives for multiple tested tools and machine learning methods using 150bp length reads.

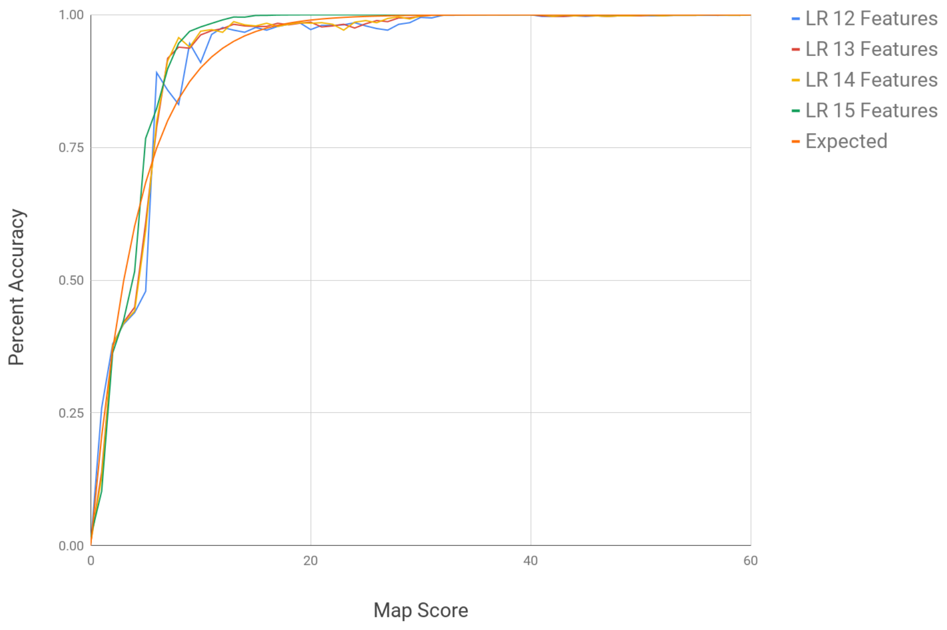

Figure 7.20: Visual representation of Accuracy per Map Score running LR on a sample Chromosome 1 data set of 10 million samples with varying numbers of parameters: 12, 13, 14 and 15.

Table 7.2: Gem3 results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	Gem3% Accuracy	Expected	Sum of FP's	TP Percent
60	8447912	56	0.999993	0.999999	56	0.844791
59	109290	119	0.998912	0.999999	175	0.855720
58	2558	6	0.997660	0.999998	181	0.855976
57	3551	11	0.996912	0.999998	192	0.856331
56	5822	18	0.996918	0.999997	210	0.856913
55	13278	26	0.998046	0.999997	236	0.858241
54	5597	22	0.996085	0.999996	258	0.858801
53	5931	19	0.996807	0.999995	277	0.859394
52	12687	32	0.997484	0.999994	309	0.860663
51	5307	26	0.995125	0.999992	335	0.861193
50	4824	11	0.997725	0.999990	346	0.861676
49	8042	31	0.996160	0.999987	377	0.862480
48	5792	18	0.996902	0.999984	395	0.863059
47	4468	18	0.995988	0.999980	413	0.863506
46	5206	15	0.997127	0.999975	428	0.864027
45	5453	31	0.994347	0.999968	459	0.864572
44	4234	19	0.995533	0.999960	478	0.864995
43	3949	20	0.994961	0.999950	498	0.865390
42	4641	25	0.994642	0.999937	523	0.865854
41	4677	27	0.994260	0.999921	550	0.866322
40	3876	26	0.993337	0.999900	576	0.866710
39	4004	31	0.992317	0.999874	607	0.867110
38	4468	32	0.992889	0.999842	639	0.867557
37	3663	23	0.993760	0.999800	662	0.867923
36	3717	37	0.990144	0.999749	699	0.868295
35	4126	37	0.991112	0.999684	736	0.868707
34	4199	35	0.991734	0.999602	771	0.869127
33	3713	38	0.989869	0.999499	809	0.869499
32	3905	49	0.987607	0.999369	858	0.869889
31	4137	57	0.986409	0.999206	915	0.870303
30	3782	52	0.986437	0.999000	967	0.870681
29	4066	63	0.984742	0.998741	1030	0.871088
28	4288	73	0.983261	0.998415	1103	0.871516
27	4295	79	0.981939	0.998005	1182	0.871946
26	4331	74	0.983201	0.997488	1256	0.872379
25	4588	81	0.982652	0.996838	1337	0.872838
24	4825	92	0.981289	0.996019	1429	0.873320
23	4829	99	0.979911	0.994988	1528	0.873803
22	5269	104	0.980644	0.993690	1632	0.874330
21	5520	131	0.976818	0.992057	1763	0.874882
20	5612	162	0.971943	0.990000	1925	0.875443
19	6096	200	0.968234	0.987411	2125	0.876053
18	5983	265	0.957586	0.984151	2390	0.876651
17	5923	254	0.958880	0.980047	2644	0.877243
16	5592	305	0.948279	0.974881	2949	0.877803
15	5128	316	0.941954	0.968377	3265	0.878315
14	4676	352	0.929992	0.960189	3617	0.878783
13	3977	403	0.907991	0.949881	4020	0.879181
12	3679	407	0.900392	0.936904	4427	0.879549
11	3293	457	0.878133	0.920567	4884	0.879878
10	3163	558	0.850040	0.900000	5442	0.880194
9	2788	556	0.833732	0.874107	5998	0.880473
8	2700	635	0.809595	0.841511	6633	0.880743
7	2529	664	0.792045	0.800474	7297	0.880996
6	2364	763	0.755996	0.748811	8060	0.881232
5	2085	857	0.708702	0.683772	8917	0.881441
4	1623	932	0.635225	0.601893	9849	0.881603
3	1230	1084	0.531547	0.498813	10933	0.881726
2	1094	1441	0.431558	0.369043	12374	0.881836
1	939	2645	0.261998	0.205672	15019	0.881929
0	438693	726994	0.376339	0.000000	742013	0.925799

Table 7.3: BWA results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	BWA % Accuracy	Expected	Sum of FP's	TP Percent
60	8191443	1439	0.999824	0.999999	1439	0.819144
59	9878	3	0.999696	0.999999	1442	0.820132
58	9805	3	0.999694	0.999998	1445	0.821113
57	10004	2	0.999800	0.999998	1447	0.822113
56	9668	1	0.999897	0.999997	1448	0.823080
55	9542	4	0.999581	0.999997	1452	0.824034
54	9422	4	0.999576	0.999996	1456	0.824976
53	9439	2	0.999788	0.999995	1458	0.825920
52	9148	1	0.999891	0.999994	1459	0.826835
51	8809	1	0.999886	0.999992	1460	0.827716
50	9031	2	0.999779	0.999990	1462	0.828619
49	8650	3	0.999653	0.999987	1465	0.829484
48	8956	2	0.999777	0.999984	1467	0.830380
47	9020	2	0.999778	0.999980	1469	0.831282
46	8119	2	0.999754	0.999975	1471	0.832093
45	8612	3	0.999652	0.999968	1474	0.832955
44	7539	2	0.999735	0.999960	1476	0.833709
43	8103	2	0.999753	0.999950	1478	0.834519
42	8413	1	0.999881	0.999937	1479	0.835360
41	8577	3	0.999650	0.999921	1482	0.836218
40	8597	1	0.999884	0.999900	1483	0.837078
39	7750	1	0.999871	0.999874	1484	0.837853
38	7698	0	1.000000	0.999842	1484	0.838622
37	8416	8	0.999050	0.999800	1492	0.839464
36	8036	1	0.999876	0.999749	1493	0.840268
35	8900	3	0.999663	0.999684	1496	0.841158
34	7134	5	0.999300	0.999602	1501	0.841871
33	7652	2	0.999739	0.999499	1503	0.842636
32	8793	3	0.999659	0.999369	1506	0.843515
31	8335	2	0.999760	0.999206	1508	0.844349
30	9433	6	0.999364	0.999000	1514	0.845292
29	6701	2	0.999702	0.998741	1516	0.845962
28	7095	2	0.999718	0.998415	1518	0.846672
27	8298	5	0.999398	0.998005	1523	0.847502
26	10352	5	0.999517	0.997488	1528	0.848537
25	11079	7	0.999369	0.996838	1535	0.849645
24	6639	4	0.999398	0.996019	1539	0.850309
23	7218	2	0.999723	0.994988	1541	0.851030
22	8707	2	0.999770	0.993690	1543	0.851901
21	12885	8	0.999380	0.992057	1551	0.853190
20	14365	8	0.999443	0.990000	1559	0.854626
19	6635	6	0.999097	0.987411	1565	0.855290
18	7202	11	0.998475	0.984151	1576	0.856010
17	8486	10	0.998823	0.980047	1586	0.856858
16	18344	19	0.998965	0.974881	1605	0.858693
15	20757	28	0.998653	0.968377	1633	0.860769
14	6830	22	0.996789	0.960189	1655	0.861452
13	7753	28	0.996401	0.949881	1683	0.862227
12	7814	35	0.995541	0.936904	1718	0.863008
11	21931	78	0.996456	0.920567	1796	0.865201
10	43035	152	0.996480	0.900000	1948	0.869505
9	6362	65	0.989886	0.874107	2013	0.870141
8	7867	106	0.986705	0.841511	2119	0.870928
7	187	7	0.963918	0.800474	2126	0.870946
6	36	1	0.972973	0.748811	2127	0.870950
5	1002	20	0.980431	0.683772	2147	0.871050
4	5078	101	0.980498	0.601893	2248	0.871558
3	8472	116	0.986493	0.498813	2364	0.872405
2	66170	567	0.991504	0.369043	2931	0.879022
1	4677	122	0.974578	0.205672	3053	0.879490
0	472252	729796	0.392873	0.000000	732849	0.926715

Table 7.4: MiniMap2 results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	MiniMapper2% Accuracy	Expected	Sum of FP's	TP Percent
60	7551074	193	0.999974	0.999999	193	0.755107
59	17869	1	0.999944	0.999999	194	0.756894
58	16908	0	1.000000	0.999998	194	0.758585
57	16148	1	0.999938	0.999998	195	0.760200
56	16191	0	1.000000	0.999997	195	0.761819
55	15996	0	1.000000	0.999997	195	0.763419
54	23839	0	1.000000	0.999996	195	0.765803
53	24833	1	0.999960	0.999995	196	0.768286
52	17215	0	1.000000	0.999994	196	0.770007
51	17362	2	0.999885	0.999992	198	0.771744
50	16323	0	1.000000	0.999990	198	0.773376
49	16165	0	1.000000	0.999987	198	0.774992
48	17208	0	1.000000	0.999984	198	0.776713
47	16189	1	0.999938	0.999980	199	0.778332
46	16226	0	1.000000	0.999975	199	0.779955
45	15770	1	0.999937	0.999968	200	0.781532
44	15368	1	0.999935	0.999960	201	0.783068
43	15034	0	1.000000	0.999950	201	0.784572
42	17143	1	0.999942	0.999937	202	0.786286
41	14809	0	1.000000	0.999921	202	0.787767
40	14707	0	1.000000	0.999900	202	0.789238
39	14377	1	0.999930	0.999874	203	0.790675
38	14504	1	0.999931	0.999842	204	0.792126
37	14303	3	0.999790	0.999800	207	0.793556
36	41404	3	0.999928	0.999749	210	0.797697
35	21977	1	0.999954	0.999684	211	0.799894
34	16032	0	1.000000	0.999602	211	0.801497
33	15493	0	1.000000	0.999499	211	0.803047
32	14600	0	1.000000	0.999369	211	0.804507
31	14700	1	0.999932	0.999206	212	0.805977
30	15418	2	0.999870	0.999000	214	0.807519
29	14780	1	0.999932	0.998741	215	0.808997
28	14589	4	0.999726	0.998415	219	0.810455
27	13958	1	0.999928	0.998005	220	0.811851
26	13831	3	0.999783	0.997488	223	0.813234
25	13903	2	0.999856	0.996838	225	0.814625
24	13986	2	0.999857	0.996019	227	0.816023
23	13531	5	0.999631	0.994988	232	0.817376
22	13880	3	0.999784	0.993690	235	0.818764
21	15035	8	0.999468	0.992057	243	0.820268
20	14487	8	0.999448	0.990000	251	0.821717
19	14183	12	0.999155	0.987411	263	0.823135
18	14362	15	0.998957	0.984151	278	0.824571
17	13958	18	0.998712	0.980047	296	0.825967
16	14393	17	0.998820	0.974881	313	0.827406
15	57122	87	0.998479	0.968377	400	0.833118
14	19592	23	0.998827	0.960189	423	0.835078
13	26840	101	0.996251	0.949881	524	0.837762
12	22019	111	0.994984	0.936904	635	0.839963
11	20217	146	0.992830	0.920567	781	0.841985
10	22205	191	0.991472	0.900000	972	0.844206
9	17105	107	0.993783	0.874107	1079	0.845916
8	17536	117	0.993372	0.841511	1196	0.847670
7	17801	167	0.990706	0.800474	1363	0.849450
6	17510	164	0.990721	0.748811	1527	0.851201
5	18937	230	0.988000	0.683772	1757	0.853095
4	18232	300	0.983812	0.601893	2057	0.854918
3	20916	441	0.979351	0.498813	2498	0.857009
2	22707	621	0.973380	0.369043	3119	0.859280
1	66015	8621	0.884493	0.205672	11740	0.865882
0	374686	954759	0.281836	0.000000	966499	0.903350

Table 7.5: SNAP results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12. *Snap has 1 less read than the rest as it skips over mason read 0.

Map Score	Hit	Miss	SNAP% Accuracy	Expected	Sum of FP's	TP Percent
70	8267170	676	0.999918	1.000000	676	0.826717
69	31258	651	0.979598	1.000000	1327	0.829843
68	27498	781	0.972382	1.000000	2108	0.832593
67	27008	1685	0.941275	1.000000	3793	0.835293
66	25967	1566	0.943123	1.000000	5359	0.837890
65	21392	902	0.959541	1.000000	6261	0.840029
64	17040	671	0.962114	1.000000	6932	0.841733
63	16593	3152	0.840365	0.999999	10084	0.843393
62	15146	1136	0.930230	0.999999	11220	0.844907
61	15784	786	0.952565	0.999999	12006	0.846486
60	14325	355	0.975817	0.999999	12361	0.847918
59	49956	297	0.994090	0.999999	12658	0.852914
58	9549	227	0.976780	0.999998	12885	0.853869
57	9054	233	0.974911	0.999998	13118	0.854774
56	19399	418	0.978907	0.999997	13536	0.856714
55	10844	378	0.966316	0.999997	13914	0.857798
54	12376	280	0.977876	0.999996	14194	0.859036
53	10016	313	0.969697	0.999995	14507	0.860038
52	8010	731	0.916371	0.999994	15238	0.860839
51	7623	483	0.940415	0.999992	15721	0.861601
50	6729	382	0.946280	0.999990	16103	0.862274
49	4814	434	0.917302	0.999987	16537	0.862755
48	4555	230	0.951933	0.999984	16767	0.863211
47	4165	256	0.942095	0.999980	17023	0.863627
46	4158	233	0.946937	0.999975	17256	0.864043
45	5477	192	0.966132	0.999968	17448	0.864591
44	4201	132	0.969536	0.999960	17580	0.865011
43	3634	116	0.969067	0.999950	17696	0.865374
42	3240	68	0.979444	0.999937	17764	0.865698
41	2922	59	0.980208	0.999921	17823	0.865990
40	2882	66	0.977612	0.999900	17889	0.866279
39	3890	76	0.980837	0.999874	17965	0.866668
38	2314	37	0.984262	0.999842	18002	0.866899
37	1975	40	0.980149	0.999800	18042	0.867096
36	2218	52	0.977093	0.999749	18094	0.867318
35	1912	27	0.986075	0.999684	18121	0.867509
34	2050	35	0.983213	0.999602	18156	0.867714
33	3971	41	0.989781	0.999499	18197	0.868112
32	1776	24	0.986667	0.999369	18221	0.868289
31	1276	29	0.977778	0.999206	18250	0.868417
30	1534	30	0.980818	0.999000	18280	0.868570
29	69570	481	0.993134	0.998741	18761	0.875527
28	2438	334	0.879509	0.998415	19095	0.875771
27	2459	381	0.865845	0.998005	19476	0.876017
26	19109	1036	0.948573	0.997488	20512	0.877928
25	2417	833	0.743692	0.996838	21345	0.878169
24	9170	692	0.929832	0.996019	22037	0.879086
23	6355	846	0.882516	0.994988	22883	0.879722
22	4905	2056	0.704640	0.993690	24939	0.880212
21	5438	922	0.855031	0.992057	25861	0.880756
20	3302	1032	0.761883	0.990000	26893	0.881086
19	2629	1257	0.676531	0.987411	28150	0.881349
18	2471	763	0.764069	0.984151	28913	0.881596
17	2364	893	0.725821	0.980047	29806	0.881833
16	2433	775	0.758416	0.974881	30581	0.882076
15	1982	632	0.758225	0.968377	31213	0.882274
14	1799	509	0.779463	0.960189	31722	0.882454
13	1883	505	0.788526	0.949881	32227	0.882643
12	1908	433	0.815036	0.936904	32660	0.882833
11	1890	413	0.820669	0.920567	33073	0.883022
10	1787	317	0.849335	0.900000	33390	0.883201
9	1456	258	0.849475	0.874107	33648	0.883347
8	1336	253	0.840780	0.841511	33901	0.883480
7	1457	249	0.854045	0.800474	34150	0.883626
6	1386	253	0.845638	0.748811	34403	0.883765
5	1357	186	0.879456	0.683772	34589	0.883900
4	1520	215	0.876081	0.601893	34804	0.884052
3	271893	259660	0.511507	0.498813	294464	0.911242
2	6636	6275	0.513980	0.369043	300739	0.911905
1	62676	128553	0.327754	0.205672	429292	0.918173
0	83019	305961	0.213427	0.000000	735253	0.926475

Table 7.6: BowTie2 results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	BowTie2% Accuracy	Expected	Sum of FP's	TP Percent
42	7037704	461	0.999934	0.999937	461	0.703770
41	0	0	n/a	0.999921	461	0.703770
40	734013	369	0.999498	0.999900	830	0.777172
39	67106	3	0.999955	0.999874	833	0.783882
38	34460	2	0.999942	0.999842	835	0.787328
37	31733	0	1.000000	0.999800	835	0.790502
36	18576	2	0.999892	0.999749	837	0.792359
35	22492	3	0.999867	0.999684	840	0.794608
34	15103	5	0.999669	0.999602	845	0.796119
33	29633	24	0.999191	0.999499	869	0.799082
32	21478	5	0.999767	0.999369	874	0.801230
31	13435	11	0.999182	0.999206	885	0.802573
30	28377	29	0.998979	0.999000	914	0.805411
29	0	0	n/a	0.998741	914	0.805411
28	0	0	n/a	0.998415	914	0.805411
27	63406	127	0.998001	0.998005	1041	0.811752
26	78287	311	0.996043	0.997488	1352	0.819580
25	40722	201	0.995088	0.996838	1553	0.823653
24	158176	187	0.998819	0.996019	1740	0.839470
23	40670	220	0.994620	0.994988	1960	0.843537
22	46912	319	0.993246	0.993690	2279	0.848228
21	26551	199	0.992561	0.992057	2478	0.850883
20	0	0	n/a	0.990000	2478	0.850883
19	0	0	n/a	0.987411	2478	0.850883
18	28677	211	0.992696	0.984151	2689	0.853751
17	20553	346	0.983444	0.980047	3035	0.855806
16	11670	239	0.979931	0.974881	3274	0.856973
15	20980	655	0.969725	0.968377	3929	0.859071
14	6948	185	0.974064	0.960189	4114	0.859766
13	0	0	n/a	0.949881	4114	0.859766
12	45913	1299	0.972486	0.936904	5413	0.864358
11	15389	990	0.939557	0.920567	6403	0.865896
10	0	0	n/a	0.900000	6403	0.865896
9	0	0	n/a	0.874107	6403	0.865896
8	5745	102	0.982555	0.841511	6505	0.866471
7	33327	4357	0.884381	0.800474	10862	0.869804
6	124999	13219	0.904361	0.748811	24081	0.882304
5	1863	198	0.903930	0.683772	24279	0.882490
4	1161	244	0.826335	0.601893	24523	0.882606
3	2618	1009	0.721809	0.498813	25532	0.882868
2	1846	8840	0.172749	0.369043	34372	0.883052
1	389932	688573	0.361549	0.205672	722945	0.922046
0	9245	47355	0.163339	0.000000	770300	0.922970

Table 7.7: Logistic Regression Model results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	LR Model% Accuracy	Expected	Sum of FP's	TP Percent
60	65025	19	0.999708	0.999999	19	0.006503
59	11277	0	1.000000	0.999999	19	0.007630
58	12201	5	0.999590	0.999998	24	0.008850
57	11991	5	0.999583	0.999998	29	0.010049
56	13663	7	0.999488	0.999997	36	0.011416
55	14884	5	0.999664	0.999997	41	0.012904
54	13785	9	0.999348	0.999996	50	0.014283
53	14214	6	0.999578	0.999995	56	0.015704
52	15755	13	0.999176	0.999994	69	0.017280
51	16086	8	0.999503	0.999992	77	0.018888
50	14821	8	0.999461	0.999990	85	0.020370
49	15388	2	0.999870	0.999987	87	0.021909
48	17290	16	0.999075	0.999984	103	0.023638
47	16450	10	0.999392	0.999980	113	0.025283
46	15065	15	0.999005	0.999975	128	0.026790
45	17065	13	0.999239	0.999968	141	0.028496
44	17269	19	0.998901	0.999960	160	0.030223
43	15718	16	0.998983	0.999950	176	0.031795
42	15539	22	0.998586	0.999937	198	0.033349
41	16575	12	0.999277	0.999921	210	0.035006
40	16439	20	0.998785	0.999900	230	0.036650
39	15232	23	0.998492	0.999874	253	0.038173
38	15808	23	0.998547	0.999842	276	0.039754
37	16284	18	0.998896	0.999800	294	0.041382
36	15333	19	0.998762	0.999749	313	0.042916
35	16021	31	0.998069	0.999684	344	0.044518
34	15438	26	0.998319	0.999602	370	0.046062
33	15028	24	0.998406	0.999499	394	0.047564
32	14081	27	0.998086	0.999369	421	0.048973
31	16168	24	0.998518	0.999206	445	0.050589
30	15519	33	0.997878	0.999000	478	0.052141
29	14356	31	0.997845	0.998741	509	0.053577
28	12983	29	0.997771	0.998415	538	0.054875
27	13604	35	0.997434	0.998005	573	0.056236
26	179700	30	0.999833	0.997488	603	0.074206
25	1756152	35	0.999980	0.996838	638	0.249821
24	575292	33	0.999943	0.996019	671	0.307350
23	1930411	50	0.999974	0.994988	721	0.500391
22	949331	55	0.999942	0.993690	776	0.595324
21	854153	61	0.999929	0.992057	837	0.680739
20	590895	67	0.999887	0.990000	904	0.739829
19	359327	194	0.999460	0.987411	1098	0.775762
18	308370	251	0.999187	0.984151	1349	0.806599
17	180877	357	0.998030	0.980047	1706	0.824686
16	125570	471	0.996263	0.974881	2177	0.837243
15	79948	607	0.992465	0.968377	2784	0.845238
14	55996	818	0.985602	0.960189	3602	0.850838
13	54219	907	0.983547	0.949881	4509	0.856260
12	28147	1000	0.965691	0.936904	5509	0.859074
11	21748	1130	0.950608	0.920567	6639	0.861249
10	24022	1274	0.949636	0.900000	7913	0.863651
9	44316	1533	0.966564	0.874107	9446	0.868083
8	21118	1713	0.924970	0.841511	11159	0.870195
7	23212	2320	0.909134	0.800474	13479	0.872516
6	70310	3676	0.950315	0.748811	17155	0.879547
5	38089	6941	0.845858	0.683772	24096	0.883356
4	30834	20972	0.595182	0.601893	45068	0.886439
3	249688	276565	0.474464	0.498813	321633	0.911408
2	128923	258628	0.332661	0.369043	580261	0.924300
1	12628	45236	0.218236	0.205672	625497	0.925563
0	2356	116516	0.019820	0.000000	742013	0.925799

Table 7.8: Random Forest Model results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	LR Model% Accuracy	Expected	Sum of FP's	TP Percent
60	59104	19	0.999679	0.999999	19	0.005910
59	10233	0	1.000000	0.999999	19	0.006934
58	11066	4	0.999639	0.999998	23	0.008040
57	10902	4	0.999633	0.999998	27	0.009131
56	12422	7	0.999437	0.999997	34	0.010373
55	13455	5	0.999629	0.999997	39	0.011718
54	12538	8	0.999362	0.999996	47	0.012972
53	12917	6	0.999536	0.999995	53	0.014264
52	14353	12	0.999165	0.999994	65	0.015699
51	14610	7	0.999521	0.999992	72	0.017160
50	13482	7	0.999481	0.999990	79	0.018508
49	13975	2	0.999857	0.999987	81	0.019906
48	15722	13	0.999174	0.999984	94	0.021478
47	14994	10	0.999334	0.999980	104	0.022977
46	13686	12	0.999124	0.999975	116	0.024346
45	15534	13	0.999164	0.999968	129	0.025899
44	15717	14	0.999110	0.999960	143	0.027471
43	14220	16	0.998876	0.999950	159	0.028893
42	14128	20	0.998586	0.999937	179	0.030306
41	15097	12	0.999206	0.999921	191	0.031816
40	14964	17	0.998865	0.999900	208	0.033312
39	13830	22	0.998412	0.999874	230	0.034695
38	14327	21	0.998536	0.999842	251	0.036128
37	14770	15	0.998985	0.999800	266	0.037605
36	13970	19	0.998642	0.999749	285	0.039002
35	14530	27	0.998145	0.999684	312	0.040455
34	14008	25	0.998218	0.999602	337	0.041855
33	13628	22	0.998388	0.999499	359	0.043218
32	12794	19	0.998517	0.999369	378	0.044498
31	14690	23	0.998437	0.999206	401	0.045967
30	14096	29	0.997947	0.999000	430	0.047376
29	13033	28	0.997856	0.998741	458	0.048680
28	11742	27	0.997706	0.998415	485	0.049854
27	12355	32	0.997417	0.998005	517	0.051089
26	163431	27	0.999835	0.997488	544	0.067432
25	1595989	30	0.999981	0.996838	574	0.227031
24	522399	29	0.999944	0.996019	603	0.279271
23	1754654	45	0.999974	0.994988	648	0.454737
22	862191	48	0.999944	0.993690	696	0.540956
21	776007	57	0.999927	0.992057	753	0.618556
20	537156	60	0.999888	0.990000	813	0.672272
19	326059	178	0.999454	0.987411	991	0.704878
18	280437	227	0.999191	0.984151	1218	0.732922
17	164231	330	0.997995	0.980047	1548	0.749345
16	114203	434	0.996214	0.974881	1982	0.760765
15	72633	557	0.992390	0.968377	2539	0.768028
14	50978	746	0.985577	0.960189	3285	0.773126
13	49137	826	0.983468	0.949881	4111	0.778040
12	25496	897	0.966014	0.936904	5008	0.780589
11	19793	1030	0.950535	0.920567	6038	0.782569
10	21712	1156	0.949449	0.900000	7194	0.784740
9	40190	1392	0.966524	0.874107	8586	0.788759
8	19258	1551	0.925465	0.841511	10137	0.790685
7	21114	2122	0.908676	0.800474	12259	0.792796
6	63864	3357	0.950060	0.748811	15616	0.799182
5	34662	6311	0.845972	0.683772	21927	0.802649
4	28010	19027	0.595489	0.601893	40954	0.805450
3	226682	251171	0.474376	0.498813	292125	0.828118
2	116937	235034	0.332235	0.369043	527159	0.839812
1	11484	41024	0.218710	0.205672	568183	0.840960
0	848388	173830	0.829948	0.000000	742013	0.925799

Table 7.9: XGBoost Model results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	XGBoost Model% Accuracy	Expected	Sum of FP's	TP Percent
44	1	0	1.000000	0.999960	0	0.000000
43	1	0	1.000000	0.999950	0	0.000000
42	1	0	1.000000	0.999937	0	0.000000
41	3	0	1.000000	0.999921	0	0.000001
40	11	0	1.000000	0.999900	0	0.000002
39	20	0	1.000000	0.999874	0	0.000004
38	32	0	1.000000	0.999842	0	0.000007
37	56	0	1.000000	0.999800	0	0.000013
36	96	0	1.000000	0.999749	0	0.000022
35	127	0	1.000000	0.999684	0	0.000035
34	189	0	1.000000	0.999602	0	0.000054
33	269	0	1.000000	0.999499	0	0.000081
32	422	0	1.000000	0.999369	0	0.000123
31	665	1	0.998498	0.999206	1	0.000189
30	1391	1	0.999282	0.999000	2	0.000328
29	2268	0	1.000000	0.998741	2	0.000555
28	4473	0	1.000000	0.998415	2	0.001003
27	10040	3	0.999701	0.998005	5	0.002007
26	22035	0	1.000000	0.997488	5	0.004210
25	45245	3	0.999934	0.996838	8	0.008735
24	119165	7	0.999941	0.996019	15	0.020651
23	413454	59	0.999857	0.994988	74	0.061996
22	7764443	113	0.999985	0.993690	187	0.838441
21	235701	118	0.999500	0.992057	305	0.862011
20	93357	124	0.998674	0.990000	429	0.871347
19	37588	109	0.997109	0.987411	538	0.875105
18	22637	108	0.995252	0.984151	646	0.877369
17	15754	146	0.990818	0.980047	792	0.878944
16	11626	162	0.986257	0.974881	954	0.880107
15	8583	199	0.977340	0.968377	1153	0.880965
14	6366	212	0.967771	0.960189	1365	0.881602
13	7400	347	0.955208	0.949881	1712	0.882342
12	6832	420	0.942085	0.936904	2132	0.883025
11	8762	759	0.920281	0.920567	2891	0.883901
10	6843	799	0.895446	0.900000	3690	0.884586
9	3588	498	0.878120	0.874107	4188	0.884944
8	3206	646	0.832295	0.841511	4834	0.885265
7	2878	841	0.773864	0.800474	5675	0.885553
6	2008	753	0.727273	0.748811	6428	0.885754
5	2519	1254	0.667638	0.683772	7682	0.886006
4	5707	4633	0.551934	0.601893	12315	0.886576
3	244769	238314	0.506681	0.498813	250629	0.911053
2	106021	177995	0.373292	0.369043	428624	0.921655
1	38011	169824	0.182890	0.205672	598448	0.925456
0	3424	143565	0.023294	0.000000	742013	0.925799

Table 7.10: Logistic Regression with Regularization Model results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	LR w/ Reg Model% Accuracy	Expected	Sum of FP's	TP Percent
60	8015422	46	0.999994	0.999999	46	0.801542
59	27708	8	0.999711	0.999999	54	0.804313
58	22574	11	0.999513	0.999998	65	0.806570
57	32202	16	0.999503	0.999998	81	0.809791
56	206202	37	0.999821	0.999997	118	0.830411
55	33067	26	0.999214	0.999997	144	0.833718
54	16652	43	0.997424	0.999996	187	0.835383
53	17305	213	0.987841	0.999995	400	0.837113
52	106731	205	0.998083	0.999994	605	0.847786
51	60570	194	0.996807	0.999992	799	0.853843
50	25942	185	0.992919	0.999990	984	0.856438
49	15739	430	0.973406	0.999987	1414	0.858011
48	29230	415	0.986001	0.999984	1829	0.860934
47	86968	871	0.990084	0.999980	2700	0.869631
46	54974	941	0.983171	0.999975	3641	0.875129
45	25639	1199	0.955325	0.999968	4840	0.877693
44	16325	997	0.942443	0.999960	5837	0.879325
43	26187	1220	0.955486	0.999950	7057	0.881944
42	247690	344380	0.418346	0.999937	351437	0.906713
41	108297	154407	0.412240	0.999921	505844	0.917542
40	32670	51089	0.390048	0.999900	556933	0.920809
39	16951	16993	0.499381	0.999874	573926	0.922505
38	6657	13401	0.331888	0.999842	587327	0.923170
37	3300	5027	0.396301	0.999800	592354	0.923500
36	3821	6650	0.364913	0.999749	599004	0.923882
35	5939	5582	0.515493	0.999684	604586	0.924476
34	1929	4238	0.312794	0.999602	608824	0.924669
33	1720	4203	0.290393	0.999499	613027	0.924841
32	2308	42184	0.051874	0.999369	655211	0.925072
31	2424	2558	0.486552	0.999206	657769	0.925314
30	691	1162	0.372909	0.999000	658931	0.925383
29	487	1019	0.323373	0.998741	659950	0.925432
28	553	1182	0.318732	0.998415	661132	0.925487
27	988	1634	0.376812	0.998005	662766	0.925586
26	522	859	0.377987	0.997488	663625	0.925638
25	209	714	0.226436	0.996838	664339	0.925659
24	153	684	0.182796	0.996019	665023	0.925675
23	274	895	0.234388	0.994988	665918	0.925702
22	307	565	0.352064	0.993690	666483	0.925733
21	53	403	0.116228	0.992057	666886	0.925738
20	66	449	0.128155	0.990000	667335	0.925745
19	78	584	0.117825	0.987411	667919	0.925752
18	131	426	0.235189	0.984151	668345	0.925766
17	18	271	0.062284	0.980047	668616	0.925767
16	17	329	0.049133	0.974881	668945	0.925769
15	38	406	0.085586	0.968377	669351	0.925773
14	56	308	0.153846	0.960189	669659	0.925778
13	8	191	0.040201	0.949881	669850	0.925779
12	8	276	0.028169	0.936904	670126	0.925780
11	12	325	0.035608	0.920567	670451	0.925781
10	26	257	0.091873	0.900000	670708	0.925784
9	4	151	0.025806	0.874107	670859	0.925784
8	4	270	0.014599	0.841511	671129	0.925785
7	15	312	0.045872	0.800474	671441	0.925786
6	4	162	0.024096	0.748811	671603	0.925787
5	0	204	0.000000	0.683772	671807	0.925787
4	11	324	0.032836	0.601893	672131	0.925788
3	2	211	0.009390	0.498813	672342	0.925788
2	10	416	0.023474	0.369043	672758	0.925789
1	7	624	0.011094	0.205672	673382	0.925790
0	92	68631	0.001339	0.000000	742013	0.925799

Table 7.11: NN Model 1204 results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	NN Model 1204% Accuracy	Expected	Sum of FP's	TP Percent
27	1	0	1.000000	0.998005	0	0.000000
26	7	0	1.000000	0.997488	0	0.000001
25	36	0	1.000000	0.996838	0	0.000004
24	452	0	1.000000	0.996019	0	0.000050
23	3482	0	1.000000	0.994988	0	0.000398
22	1615782	1	0.999999	0.993690	1	0.161976
21	5707061	4	0.999999	0.992057	5	0.732682
20	968032	63	0.999935	0.990000	68	0.829485
19	269695	101	0.999626	0.987411	169	0.856455
18	82226	198	0.997598	0.984151	367	0.864677
17	90895	428	0.995313	0.980047	795	0.873767
16	33849	401	0.988292	0.974881	1196	0.877152
15	27419	536	0.980826	0.968377	1732	0.879894
14	13333	528	0.961908	0.960189	2260	0.881227
13	7973	577	0.932515	0.949881	2837	0.882024
12	7218	742	0.906784	0.936904	3579	0.882746
11	5524	646	0.895300	0.920567	4225	0.883299
10	6912	958	0.878272	0.900000	5183	0.883990
9	4196	896	0.824038	0.874107	6079	0.884409
8	4107	1324	0.756214	0.841511	7403	0.884820
7	4064	1480	0.733045	0.800474	8883	0.885226
6	7249	2242	0.763776	0.748811	11125	0.885951
5	2881	3721	0.436383	0.683772	14846	0.886239
4	11144	14121	0.441085	0.601893	28967	0.887354
3	321780	381723	0.457397	0.498813	410690	0.919532
2	57050	189172	0.231701	0.369043	599862	0.925237
1	4082	23311	0.149016	0.205672	623173	0.925645
0	1537	118840	0.012768	0.000000	742013	0.925799

Table 7.12: NN Model 1209 results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	NN Model 1209 %Accuracy	Expected	Sum FP's	TP%
28	1	0	1.000000	0.998415	0	1E-07
27	5	0	1.000000	0.998005	0	6E-07
26	15	0	1.000000	0.997488	0	2.1E-06
25	159	0	1.000000	0.996838	0	1.8E-05
24	910	0	1.000000	0.996019	0	0.000109
23	3214	0	1.000000	0.994988	0	0.0004304
22	149545	0	1.000000	0.993690	0	0.0153849
21	5765494	1	1.000000	0.992057	1	0.5919343
20	1953073	48	0.999975	0.990000	49	0.7872416
19	446178	74	0.999834	0.987411	123	0.8318594
18	236629	78	0.999670	0.984151	201	0.8555223
17	79902	182	0.997727	0.980047	383	0.8635125
16	35147	244	0.993106	0.974881	627	0.8670272
15	81141	436	0.994655	0.968377	1063	0.8751413
14	30747	487	0.984408	0.960189	1550	0.878216
13	22422	518	0.977419	0.949881	2068	0.8804582
12	15193	560	0.964451	0.936904	2628	0.8819775
11	8980	557	0.941596	0.920567	3185	0.8828755
10	5688	512	0.917419	0.900000	3697	0.8834443
9	4405	671	0.867809	0.874107	4368	0.8838848
8	3822	893	0.810604	0.841511	5261	0.884267
7	7788	1126	0.873682	0.800474	6387	0.8850458
6	3354	877	0.792720	0.748811	7264	0.8853812
5	2086	1294	0.617160	0.683772	8558	0.8855898
4	2525	1601	0.611973	0.601893	10159	0.8858423
3	301742	344910	0.466622	0.498813	355069	0.9160165
2	89834	227816	0.282808	0.369043	582885	0.9249999
1	7040	44027	0.137858	0.205672	626912	0.9257039
0	948	115101	0.008169	0.000000	742013	0.9257987

Table 7.13: NN Model 1204 with regularization version 14 results on 10 million reads generated by mason with a seed of 2 on Human Genome GR38.12

Map Score	Hit	Miss	NN M1204 Rv14 % Accuracy	Expected	Sum of FP's	TP%
11	644060	47139	0.931801	0.920567	47139	0.064406
10	8613927	633868	0.931457	0.900000	681007	0.064406
9	0	0	n/a	0.874107	681007	0.064406
8	0	0	n/a	0.841511	681007	0.064406
7	0	0	n/a	0.800474	681007	0.064406
6	0	0	n/a	0.748811	681007	0.064406
5	0	0	n/a	0.683772	681007	0.064406
4	0	0	n/a	0.601893	681007	0.064406
3	0	0	n/a	0.498813	681007	0.064406
2	0	0	n/a	0.369043	681007	0.064406
1	0	0	n/a	0.205672	681007	0.064406
0	0	61006	0.000000	0.000000	742013	0.064406

Table 7.14: Results of Merged XGBoost + LRwR

	XGBoost + LRwR	
Map Score	Hit	Miss
60	7876320	28
59	20105	0
58	13563	0
57	24452	0
56	166559	18
55	24270	1
54	12031	0
53	11618	1
52	86875	18
51	42486	2
50	16813	5
49	8522	3
48	8908	4
47	32420	50
46	12888	23
45	4572	6
44	2566	7
43	8336	9
42	1095	2
41	1058	1
40	663	0
39	4477	5
38	589	0
37	540	0
36	403	0
35	906	0
34	296	0
33	208	0
32	177	1
31	237	2
30	126	0
29	45	0
28	50	0
27	48	1
26	51	0
25	33	0
24	27	0
23	16	0
22	15	0
21	235717	118
20	93369	124
19	37591	109
18	22641	108
17	15755	146
16	11629	162
15	8583	199
14	6367	212
13	7401	347
12	6833	420
11	8762	759
10	6843	799
9	3588	498
8	3206	646
7	2878	841
6	2008	753
5	2519	1254
4	5707	4633
3	244770	238314
2	106021	177995
1	38011	169824
0	3424	143565

Table 7.15: Area Under Curve calculations for tools. Part 1.

Map Score Range	Gem3	BWA	MiniMap2	SNAP	BowTie2	LR	R	Expected AUC
0-1	0.319000	0.683725	0.583165	0.270591	0.262444	0.119028	0.276012	0.102836
1-2	0.347000	0.983041	0.928936	0.420867	0.267149	0.275448	0.428409	0.287357
2-3	0.482000	0.988998	0.976365	0.512744	0.447279	0.403562	0.495460	0.433928
3-4	0.583000	0.983495	0.981581	0.693794	0.774072	0.534823	0.518666	0.550353
4-5	0.672000	0.980464	0.985906	0.877768	0.865132	0.720520	0.534492	0.642833
5-6	0.732000	0.976702	0.989361	0.862547	0.904146	0.898087	0.548413	0.716292
6-7	0.774000	0.968445	0.990713	0.849841	0.894371	0.929724	0.570611	0.774643
7-8	0.801000	0.975311	0.992039	0.847412	0.933468	0.917052	0.608380	0.820992
8-9	0.822000	0.988296	0.993578	0.845128	n/a	0.945767	0.624079	0.857809
9-10	0.842000	0.993183	0.992628	0.849405	n/a	0.958100	0.659292	0.887054
10-11	0.864000	0.996468	0.992151	0.835002	2.883168	0.950122	0.785424	0.910284
11-12	0.889000	0.995998	0.993907	0.817853	0.956021	0.958149	0.904292	0.928736
12-13	0.904000	0.995971	0.995618	0.801781	0.986243	0.974619	0.949948	0.943393
13-14	0.919000	0.996595	0.997539	0.783994	0.987032	0.984574	0.969835	0.955035
14-15	0.936000	0.997721	0.998653	0.768844	0.971895	0.989033	0.982272	0.964283
15-16	0.945000	0.998809	0.998650	0.758321	0.974828	0.994364	0.986034	0.971629
16-17	0.954000	0.998894	0.998766	0.742119	0.981688	0.997147	0.989202	0.977464
17-18	0.958000	0.998649	0.998834	0.744945	0.988070	0.998608	0.992250	0.982099
18-19	0.963000	0.998786	0.999056	0.720300	2.977885	0.999324	0.993804	0.985781
19-20	0.970000	0.999270	0.999301	0.719207	n/a	0.999674	0.995214	0.988705
20-21	0.974000	0.999411	0.999458	0.808457	n/a	0.999908	0.996494	0.991028
21-22	0.979000	0.999575	0.999626	0.779836	0.992903	0.999935	0.997154	0.992874
22-23	0.980000	0.999747	0.999707	0.793578	0.993933	0.999958	0.998611	0.994339
23-24	0.981000	0.999560	0.999744	0.906174	0.996719	0.999958	0.999086	0.995504
24-25	0.982000	0.999383	0.999857	0.836762	0.996954	0.999961	0.999086	0.996428
25-26	0.983000	0.999443	0.999820	0.846133	0.995566	0.999907	1.000000	0.997163
26-27	0.983000	0.999458	0.999856	0.907209	0.997022	0.998633	0.999437	0.997746
27-28	0.983000	0.999558	0.999827	0.872677	2.995470	0.997603	0.999437	0.998210
28-29	0.984000	0.999710	0.999829	0.936321	n/a	0.997808	1.000000	0.998578
29-30	0.986000	0.999533	0.999901	0.986976	n/a	0.997862	1.000000	0.998871
30-31	0.986000	0.999562	0.999901	0.979298	0.999081	0.998198	1.000000	0.999103
31-32	0.987000	0.999710	0.999966	0.982222	0.999475	0.998302	1.000000	0.999287
32-33	0.989000	0.999699	1.000000	0.988224	0.999479	0.998246	1.000000	0.999434
33-34	0.991000	0.999519	1.000000	0.986497	0.999430	0.998362	1.000000	0.999550
34-35	0.991000	0.999481	0.999977	0.984644	0.999768	0.998194	1.000000	0.999643
35-36	0.991000	0.999769	0.999941	0.981584	0.999879	0.998416	1.000000	0.999716
36-37	0.992000	0.999463	0.999859	0.978621	1.999834	0.998829	1.000000	0.999775
37-38	0.993000	0.999525	0.999861	0.982205	n/a	0.998721	1.000000	0.999821
38-39	0.993000	0.999935	0.999931	0.982550	0.999949	0.998520	1.000000	0.999858
39-40	0.993000	0.999877	0.999965	0.979225	0.999726	0.998639	1.000000	0.999887
40-41	0.994000	0.999767	1.000000	0.978910	0.999716	0.999031	1.000000	0.999910
41-42	0.994000	0.999766	0.999971	0.979826	n/a	0.998931	1.000000	0.999929
42-43	0.995000	0.999817	0.999971	0.974255	n/a	0.998785	1.000000	0.999943
43-44	0.995000	0.999744	0.999967	0.969301	n/a	0.998942	1.000000	0.999955
44-45	0.995000	0.999693	0.999936	0.967834	n/a	0.999070	1.000000	0.999964
45-46	0.996000	0.999703	0.999968	0.956534	n/a	0.999122	1.000000	0.999972
46-47	0.997000	0.999766	0.999969	0.944516	n/a	0.999199	1.000000	0.999977
47-48	0.996000	0.999778	0.999969	0.947014	n/a	0.999234	1.000000	0.999982
48-49	0.997000	0.999715	1.000000	0.934617	n/a	0.999473	1.000000	0.999986
49-50	0.997000	0.999716	1.000000	0.931791	n/a	0.999665	1.000000	0.999989
50-51	0.996000	0.999833	0.999942	0.943347	n/a	0.999482	1.000000	0.999991
51-52	0.996000	0.999889	0.999942	0.928393	n/a	0.999339	1.000000	0.999993
52-53	0.997000	0.999839	0.999980	0.943034	n/a	0.999377	1.000000	0.999994
53-54	0.996000	0.999682	0.999980	0.973787	n/a	0.999463	1.000000	0.999996
54-55	0.997000	0.999578	1.000000	0.972096	n/a	0.999506	1.000000	0.999996
55-56	0.997000	0.999739	1.000000	0.972612	n/a	0.999576	1.000000	0.999997
56-57	0.997000	0.999848	0.999969	0.976909	n/a	0.999536	1.000000	0.999998
57-58	0.997000	0.999747	0.999969	0.975846	n/a	0.999587	1.000000	0.999998
58-59	0.998000	0.999695	0.999972	0.985435	n/a	0.999795	1.000000	0.999999
59-60	0.999000	0.999760	0.999959	0.984954	n/a	0.999854	0.999954	0.999999
60-61	n/a	n/a	n/a	0.964191	n/a	n/a	n/a	0.999999
61-62	n/a	n/a	n/a	0.941397	n/a	n/a	n/a	0.999999
62-63	n/a	n/a	n/a	0.885297	n/a	n/a	n/a	0.999999
63-64	n/a	n/a	n/a	0.901239	n/a	n/a	n/a	1.000000
64-65	n/a	n/a	n/a	0.960827	n/a	n/a	n/a	1.000000
65-66	n/a	n/a	n/a	0.951332	n/a	n/a	n/a	1.000000
66-67	n/a	n/a	n/a	0.942199	n/a	n/a	n/a	1.000000
67-68	n/a	n/a	n/a	0.956829	n/a	n/a	n/a	1.000000
68-69	n/a	n/a	n/a	0.975990	n/a	n/a	n/a	1.000000
69-70	n/a	n/a	n/a	0.989758	n/a	n/a	n/a	1.000000

Table 7.16: Area Under Curve calculations for tools. Part 2.

Map Score Range	XGBoost	LRwR	XGBoost + LrwR	NN 1204	NN 1209	NN 1204v14	Expected AUC
0-1	0.103092	0.006216	0.1030922668	0.080892	0.073014	4.657287	0.102836
1-2	0.278091	0.017284	0.2780913116	0.190359	0.210333	n/a	0.287357
2-3	0.439987	0.016432	0.4399872087	0.344549	0.374715	n/a	0.433928
3-4	0.529308	0.021113	0.5293081518	0.449241	0.539297	n/a	0.550353
4-5	0.609786	0.016418	0.60978636	0.438734	0.614566	n/a	0.642833
5-6	0.697456	0.012048	0.6974556056	0.600080	0.704940	n/a	0.716292
6-7	0.750568	0.034984	0.7505683346	0.748410	0.833201	n/a	0.774643
7-8	0.803079	0.030235	0.8030794268	0.744630	0.842143	n/a	0.820992
8-9	0.855208	0.020202	0.8552076614	0.790126	0.839207	n/a	0.857809
9-10	0.886783	0.058840	0.8867833147	0.851155	0.892614	n/a	0.887054
10-11	0.907864	0.063741	0.9078638507	0.886786	0.929508	0.931629	0.910284
11-12	0.931183	0.031889	0.9311872051	0.901042	0.953024	n/a	0.928736
12-13	0.948647	0.034185	0.948653588	0.919649	0.970935	n/a	0.943393
13-14	0.961490	0.097024	0.9614952533	0.947211	0.980914	n/a	0.955035
14-15	0.972556	0.119716	0.9725581357	0.971367	0.989532	n/a	0.964283
15-16	0.981799	0.067359	0.9818003605	0.984559	0.993880	n/a	0.971629
16-17	0.988537	0.055708	0.9885394474	0.991803	0.995416	n/a	0.977464
17-18	0.993035	0.148736	0.9930353631	0.996456	0.998699	n/a	0.982099
18-19	0.996180	0.176507	0.9961806459	0.998612	0.999752	n/a	0.985781
19-20	0.997891	0.122990	0.9978912254	0.999780	0.999905	n/a	0.988705
20-21	0.999087	0.122192	0.9990866738	0.999967	0.999988	n/a	0.991028
21-22	0.999743	0.234146	0.9997498251	0.999999	1.000000	n/a	0.992874
22-23	0.999921	0.293226	1	1.000000	1.000000	n/a	0.994339
23-24	0.999899	0.208592	1	1.000000	1.000000	n/a	0.995504
24-25	0.999937	0.204616	1	1.000000	1.000000	n/a	0.996428
25-26	0.999967	0.302211	1	1.000000	1.000000	n/a	0.997163
26-27	0.999851	0.377399	0.9897959184	1.000000	1.000000	n/a	0.997746
27-28	0.999851	0.347772	0.9897959184	n/a	1.000000	n/a	0.998210
28-29	1.000000	0.321053	1	n/a	n/a	n/a	0.998578
29-30	0.999641	0.348141	1	n/a	n/a	n/a	0.998871
30-31	0.998890	0.429730	0.9958158996	n/a	n/a	n/a	0.999103
31-32	0.999249	0.269213	0.9930069108	n/a	n/a	n/a	0.999287
32-33	1.000000	0.171134	0.9971910112	n/a	n/a	n/a	0.999434
33-34	1.000000	0.301594	1	n/a	n/a	n/a	0.999550
34-35	1.000000	0.414144	1	n/a	n/a	n/a	0.999643
35-36	1.000000	0.440203	1	n/a	n/a	n/a	0.999716
36-37	1.000000	0.380607	1	n/a	n/a	n/a	0.999775
37-38	1.000000	0.364094	1	n/a	n/a	n/a	0.999821
38-39	1.000000	0.415634	0.9994422133	n/a	n/a	n/a	0.999858
39-40	1.000000	0.444714	0.9994422133	n/a	n/a	n/a	0.999887
40-41	1.000000	0.401144	0.9995278565	n/a	n/a	n/a	0.999910
41-42	1.000000	0.415293	0.9986162794	n/a	n/a	n/a	0.999929
42-43	1.000000	0.686916	0.9985491779	n/a	n/a	n/a	0.999943
43-44	1.000000	0.948964	0.9981004751	n/a	n/a	n/a	0.999955
44-45	n/a	0.948884	0.9979844122	n/a	n/a	n/a	0.999964
45-46	n/a	0.969248	0.9984539787	n/a	n/a	n/a	0.999972
46-47	n/a	0.986628	0.9983393452	n/a	n/a	n/a	0.999977
47-48	n/a	0.988043	0.999005642	n/a	n/a	n/a	0.999982
48-49	n/a	0.979703	0.9995996304	n/a	n/a	n/a	0.999986
49-50	n/a	0.983163	0.9996753967	n/a	n/a	n/a	0.999989
50-51	n/a	0.994863	0.9998278137	n/a	n/a	n/a	0.999991
51-52	n/a	0.997445	0.9998728883	n/a	n/a	n/a	0.999993
52-53	n/a	0.992962	0.9998533914	n/a	n/a	n/a	0.999994
53-54	n/a	0.992633	0.999956967	n/a	n/a	n/a	0.999996
54-55	n/a	0.998319	0.9999793993	n/a	n/a	n/a	0.999996
55-56	n/a	0.999517	0.9999253702	n/a	n/a	n/a	0.999997
56-57	n/a	0.999662	0.9999459709	n/a	n/a	n/a	0.999998
57-58	n/a	0.999508	1	n/a	n/a	n/a	0.999998
58-59	n/a	0.999612	1	n/a	n/a	n/a	0.999999
59-60	n/a	0.999853	0.9999982225	n/a	n/a	n/a	0.999999
60-61	n/a	n/a	n/a	n/a	n/a	n/a	0.999999
61-62	n/a	n/a	n/a	n/a	n/a	n/a	0.999999
62-63	n/a	n/a	n/a	n/a	n/a	n/a	0.999999
63-64	n/a	n/a	n/a	n/a	n/a	n/a	1.000000
64-65	n/a	n/a	n/a	n/a	n/a	n/a	1.000000
65-66	n/a	n/a	n/a	n/a	n/a	n/a	1.000000
66-67	n/a	n/a	n/a	n/a	n/a	n/a	1.000000
67-68	n/a	n/a	n/a	n/a	n/a	n/a	1.000000
68-69	n/a	n/a	n/a	n/a	n/a	n/a	1.000000
69-70	n/a	n/a	n/a	n/a	n/a	n/a	1.000000

Table 7.17: Area Under Curve difference between tools and expected. Part 1.

Gem3	BWA	MiniMapper2	SNAP	BowTie2	LR	Random Forest
0.216332	0.580890	0.480329	0.167755	0.159608	0.016192	0.173176
0.059421	0.695684	0.641579	0.133510	0.020208	0.011909	0.141052
0.047625	0.555071	0.542438	0.078816	0.013351	0.030365	0.061532
0.033033	0.433143	0.431229	0.143441	0.223719	0.015530	0.031687
0.029131	0.337632	0.343073	0.234936	0.222300	0.077688	0.108341
0.016057	0.260410	0.273069	0.146255	0.187854	0.181795	0.167879
0.000622	0.193803	0.216071	0.075199	0.119728	0.155082	0.204032
0.020172	0.154319	0.171047	0.026420	0.112476	0.096060	0.212612
0.036145	0.130487	0.135769	0.012681	0.857809	0.087958	0.233730
0.045168	0.106130	0.105574	0.037649	0.887054	0.071047	0.227762
0.046197	0.086185	0.081867	0.075282	1.998119	0.039838	0.124860
0.039473	0.067263	0.065171	0.110883	0.027286	0.029414	0.024444
0.039202	0.052578	0.052225	0.141612	0.042850	0.031226	0.006555
0.036044	0.041560	0.042504	0.171041	0.031997	0.029539	0.014800
0.028310	0.033438	0.034370	0.195439	0.007611	0.024750	0.017989
0.026513	0.027180	0.027021	0.213308	0.003199	0.022735	0.014405
0.023885	0.021430	0.021302	0.235345	0.004223	0.019682	0.011738
0.023866	0.016550	0.016735	0.237154	0.005971	0.016509	0.010151
0.022871	0.013005	0.013275	0.265481	1.989380	0.013543	0.008023
0.018617	0.010565	0.010596	0.269498	n/a	0.010968	0.006509
0.016648	0.008383	0.008430	0.182571	n/a	0.008879	0.005465
0.014142	0.006701	0.006752	0.213038	0.000030	0.007062	0.004280
0.014062	0.005407	0.005368	0.200761	0.000406	0.005619	0.004271
0.014903	0.004057	0.004240	0.089330	0.001216	0.004455	0.003583
0.014458	0.002955	0.003428	0.159666	0.000525	0.003533	0.002658
0.014237	0.002280	0.002657	0.151030	0.001597	0.002744	0.002837
0.015177	0.001711	0.002109	0.090537	0.000724	0.000887	0.001691
0.015610	0.001348	0.001617	0.125533	1.996917	0.000607	0.001227
0.014577	0.001132	0.001251	0.062257	n/a	0.000770	0.001422
0.013281	0.000662	0.001031	0.011895	n/a	0.001009	0.001129
0.012680	0.000459	0.000798	0.019805	0.000022	0.000905	0.000897
0.012279	0.000422	0.000679	0.017065	0.000187	0.000985	0.000713
0.010696	0.000265	0.000566	0.011210	0.000045	0.001188	0.000566
0.008749	0.000031	0.000450	0.013053	0.000120	0.001188	0.000450
0.008220	0.000162	0.000334	0.014998	0.000125	0.001449	0.000357
0.009088	0.000053	0.000225	0.018132	0.000163	0.001301	0.000284
0.007823	0.000312	0.000084	0.021154	1.000036	0.000946	0.000225
0.006496	0.000296	0.000040	0.017616	n/a	0.001099	0.000179
0.007255	0.000078	0.000073	0.017308	0.000091	0.001338	0.000142
0.007060	0.000010	0.000078	0.020663	0.000161	0.001248	0.000113
0.006112	0.000143	0.000090	0.021000	0.000194	0.000880	0.000090
0.005478	0.000163	0.000042	0.020103	n/a	0.000997	0.000071
0.005142	0.000126	0.000027	0.025688	n/a	0.001159	0.000057
0.004708	0.000211	0.000012	0.030654	n/a	0.001013	0.000045
0.005024	0.000271	0.000029	0.032130	n/a	0.000894	0.000036
0.004235	0.000269	0.000003	0.043437	n/a	0.000850	0.000028
0.003420	0.000211	0.000008	0.055462	n/a	0.000779	0.000023
0.003537	0.000205	0.000013	0.052968	n/a	0.000748	0.000018
0.003455	0.000271	0.000014	0.065368	n/a	0.000513	0.000014
0.003046	0.000273	0.000011	0.068198	n/a	0.000323	0.000011
0.003566	0.000158	0.000049	0.056644	n/a	0.000509	0.000009
0.003688	0.000104	0.000050	0.071600	n/a	0.000654	0.000007
0.002849	0.000155	0.000014	0.056960	n/a	0.000618	0.000006
0.003550	0.000314	0.000016	0.026209	n/a	0.000533	0.000004
0.002931	0.000418	0.000004	0.027900	n/a	0.000491	0.000004
0.002515	0.000258	0.000003	0.027386	n/a	0.000421	0.000003
0.003083	0.000149	0.000029	0.023089	n/a	0.000462	0.000002
0.002712	0.000251	0.000029	0.024153	n/a	0.000411	0.000002
0.001712	0.000303	0.000027	0.014564	n/a	0.000203	0.000001
0.000546	0.000238	0.000040	0.015045	n/a	0.000145	0.000045
n/a	n/a	n/a	0.035808	n/a	n/a	n/a
n/a	n/a	n/a	0.058602	n/a	n/a	n/a
n/a	n/a	n/a	0.114702	n/a	n/a	n/a
n/a	n/a	n/a	0.098760	n/a	n/a	n/a
n/a	n/a	n/a	0.039172	n/a	n/a	n/a
n/a	n/a	n/a	0.048668	n/a	n/a	n/a
n/a	n/a	n/a	0.057801	n/a	n/a	n/a
n/a	n/a	n/a	0.043171	n/a	n/a	n/a
n/a	n/a	n/a	0.024010	n/a	n/a	n/a
n/a	n/a	n/a	0.010242	n/a	n/a	n/a

Table 7.18: Area Under Curve difference between tools and expected. Part 2.

XGBoost	LRwR	XGBoost + LRwR	NN 1204	NN 1209	NN 1204v14
0.000256	0.096620	0.000256	0.021944	0.029822	4.049877
0.009266	0.270073	0.009266	0.096998	0.077024	n/a
0.006059	0.417496	0.006059	0.089379	0.059213	n/a
0.021045	0.529240	0.021045	0.101112	0.011055	n/a
0.033046	0.626415	0.033046	0.204099	0.028266	n/a
0.018836	0.704244	0.018836	0.116212	0.011352	n/a
0.024074	0.739659	0.024074	0.026232	0.058559	n/a
0.017913	0.790757	0.017913	0.076363	0.021151	n/a
0.002601	0.837607	0.002601	0.067683	0.018602	n/a
0.000270	0.828214	0.000270	0.035899	0.005561	n/a
0.002420	0.846543	0.002420	0.023498	0.019224	0.021346
0.002447	0.896847	0.002451	0.027694	0.024288	n/a
0.005254	0.909208	0.005261	0.023744	0.027543	n/a
0.006455	0.858012	0.006460	0.007824	0.025878	n/a
0.008272	0.844567	0.008275	0.007084	0.025248	n/a
0.010169	0.904270	0.010171	0.012930	0.022251	n/a
0.011073	0.921756	0.011075	0.014338	0.017952	n/a
0.010935	0.833363	0.010936	0.014356	0.016600	n/a
0.010399	0.809274	0.010400	0.012831	0.013971	n/a
0.009186	0.865715	0.009186	0.011075	0.011199	n/a
0.008058	0.868837	0.008058	0.008939	0.008959	n/a
0.006869	0.758727	0.006876	0.007126	0.007126	n/a
0.005582	0.701113	0.005661	0.005660	0.005661	n/a
0.004396	0.786911	0.004496	0.004496	0.004496	n/a
0.003509	0.791813	0.003572	0.003572	0.003572	n/a
0.002804	0.694952	0.002837	0.002837	0.002837	n/a
0.002104	0.620347	0.007951	0.002254	0.002254	n/a
0.001641	0.650438	0.008414	n/a	0.001790	n/a
0.001422	0.677526	0.001422	n/a	n/a	n/a
0.000770	0.650730	0.001129	n/a	n/a	n/a
0.000213	0.569373	0.003287	n/a	n/a	n/a
0.000038	0.730074	0.006280	n/a	n/a	n/a
0.000566	0.828300	0.002243	n/a	n/a	n/a
0.000450	0.697957	0.000450	n/a	n/a	n/a
0.000357	0.585499	0.000357	n/a	n/a	n/a
0.000284	0.559513	0.000284	n/a	n/a	n/a
0.000225	0.619168	0.000225	n/a	n/a	n/a
0.000179	0.635727	0.000179	n/a	n/a	n/a
0.000142	0.584223	0.000416	n/a	n/a	n/a
0.000113	0.555173	0.000445	n/a	n/a	n/a
0.000090	0.598767	0.000382	n/a	n/a	n/a
0.000071	0.584636	0.001312	n/a	n/a	n/a
0.000057	0.313028	0.001394	n/a	n/a	n/a
0.000045	0.050991	0.001855	n/a	n/a	n/a
n/a	0.051080	0.001980	n/a	n/a	n/a
n/a	0.030724	0.001518	n/a	n/a	n/a
n/a	0.013350	0.001638	n/a	n/a	n/a
n/a	0.011940	0.000976	n/a	n/a	n/a
n/a	0.020282	0.000386	n/a	n/a	n/a
n/a	0.016826	0.000313	n/a	n/a	n/a
n/a	0.005128	0.000163	n/a	n/a	n/a
n/a	0.002548	0.000120	n/a	n/a	n/a
n/a	0.007032	0.000141	n/a	n/a	n/a
n/a	0.007363	0.000039	n/a	n/a	n/a
n/a	0.001677	0.000017	n/a	n/a	n/a
n/a	0.000480	0.000072	n/a	n/a	n/a
n/a	0.000336	0.000052	n/a	n/a	n/a
n/a	0.000490	0.000002	n/a	n/a	n/a
n/a	0.000386	0.000001	n/a	n/a	n/a
n/a	0.000146	0.000001	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a
n/a	n/a	n/a	n/a	n/a	n/a

Table 7.19: Gem3, BWA, MiniMap2, XGBoost, NN1204 and NN1209 re-done on different seeds for consistency and average determination

Model	Total Accuracy	Precision	Recall	F1 Score	Hit	Miss	Accuracy	Hit	Miss	Accuracy
	Efficiency Measurement				Range 20 to 60			Best (Just 60)		
Gem3										
Gem3 NSeed	0.9546288	0.9982999	0.8832560	0.9372609	8754432	1925	0.9997802	8447912	56	0.9999934
Gem3 S100	0.9611856	0.9986222	0.9390572	0.9679242	9310776	1535	0.9998352	9024859	59	0.9999935
Gem3 S200	0.9610019	0.9986020	0.9388122	0.9677845	9308837	1527	0.9998360	9023668	56	0.9999938
Gem3 S300	0.9600776	0.9986255	0.9389267	0.9678564	9309754	1538	0.9998348	9024190	54	0.9999940
Gem3 S400	0.9602399	0.9985980	0.9389747	0.9678689	9310546	1578	0.9998305	9025328	41	0.9999955
Gem3 S500	0.9601425	0.9986305	0.9388432	0.9678143	9309345	1569	0.9998315	9024144	56	0.9999938
Average	0.9595461	0.9985630	0.9296450	0.9627515	9217281.67	1612.00	0.9998247	8928350.17	53.67	0.9999940
BWA										
BWA NSeed	0.9524695	0.9996530	0.8797585	0.9358815	8546261	1559	0.9998176	8191443	1439	0.9998244
BWA S100	0.9856497	0.9992076	0.9367041	0.9669468	9137683	4879	0.9994663	8785144	3786	0.9995692
BWA S200	0.9857095	0.9991904	0.9366033	0.9668851	9136189	4950	0.9994585	8784859	3805	0.9995671
BWA S300	0.9719936	0.9996716	0.9366651	0.9671433	9139915	1719	0.9998120	8787589	1585	0.9998197
BWA S400	0.9721263	0.9996798	0.9366515	0.9671399	9140142	1686	0.9998156	8787168	1562	0.9998223
BWA S500	0.9722210	0.9996720	0.9367944	0.9672123	9142190	1715	0.9998124	8789672	1598	0.9998182
Average	0.9733616	0.9995124	0.9271962	0.9618682	9040396.67	2751.33	0.9996971	8687645.83	2295.83	0.9997368
MiniMap2										
MiniMap2 NSeed	0.9613574	0.9986460	0.8668992	0.9281206	8217165	251	0.9999695	7551074	193	0.9999744
MiniMap2 S100	0.9792524	0.9987288	0.9234120	0.9595948	8812595	1472	0.9998330	8162725	1248	0.9998471
MiniMap2 S200	0.9792605	0.9987229	0.9232687	0.9595147	8812338	1548	0.9998244	8162428	1269	0.9998446
MiniMap2 S300	0.9813097	0.9988443	0.9233729	0.9596270	8814147	305	0.9999654	8163689	226	0.9999723
MiniMap2 S400	0.9813294	0.9988311	0.9234168	0.9596446	8815328	311	0.9999647	8164227	239	0.9999707
MiniMap2 S500	0.9813926	0.9988107	0.9234704	0.9596642	8815771	262	0.9999703	8165545	201	0.9999754
Average	0.9773170	0.9987640	0.9139734	0.9543610	8714557.33	691.50	0.9999212	8061614.67	562.67	0.9999308
XGBoost										
								Best (22:60)		
XGBoost NSeed	0.9398128	0.9392624	0.9843655	0.9612852	8713465	429	0.9999508	8384407	187	0.9999777
XGBoost S100	0.9844011	0.9844968	0.9903263	0.9874029	9277871	365	0.9999607	8965280	155	0.9999827
XGBoost S200	0.9842963	0.9843857	0.9903442	0.9873559	9277126	342	0.9999631	8965152	149	0.9999834
XGBoost S300	0.9855085	0.9763465	0.9907033	0.9834725	9313746	445	0.9999522	8965053	158	0.9999824
XGBoost S400	0.9722529	0.9639444	0.9914544	0.9775059	9311247	2124	0.9997719	8962038	1460	0.9998371
XGBoost S500	0.9723095	0.9639782	0.9914314	0.9775121	9313537	2148	0.9997694	9189755	1467	0.9998404
Average	0.9730969	0.9687357	0.9897708	0.9790891	9201165.33	975.50	0.9998947	8905280.83	596	0.9999339
NN1204										
								Best (21:60)		
NN1204 NSeed	0.9375290	0.9369234	0.9871623	0.9613870	8294853	68	0.9999918	7326821	5	0.9999993
NN1204 S100	0.9828106	0.9827727	0.9922321	0.9874797	8887534	68	0.9999923	7893944	3	0.9999996
NN1204 S200	0.9826911	0.9826503	0.9922586	0.9874311	8885238	70	0.9999921	7890981	2	0.9999997
NN1204 S300	0.9912016	0.9836839	0.9923123	0.9879793	8886594	65	0.9999927	7892784	4	0.9999995
NN1204 S400	0.9910994	0.9835625	0.9922888	0.9879064	8885919	71	0.9999920	7891173	1	0.9999999
NN1204 S500	0.9910719	0.9835358	0.9922832	0.9878902	8888091	48	0.9999946	7893112	2	0.9999997
Average	0.9794006	0.9755214	0.9914229	0.9833456	8788038.17	65.00	0.9999926	7798135.83	2.83	0.9999996
NN1209										
NN1209 NSeed	0.9372140	0.9365727	0.9876189	0.9614187	7872416	49	0.9999938	5919343	1	0.9999998
NN1209 S100	0.9826095	0.9825358	0.9925138	0.9874996	8475630	55	0.9999935	6430858	2	0.9999997
NN1209 S200	0.9824835	0.9824062	0.9925488	0.9874514	8473935	63	0.9999926	6429420	0	1.0000000
NN1209 S300	0.9907545	0.9834637	0.9925908	0.9880062	8474993	56	0.9999934	6432145	1	0.9999998
NN1209 S400	0.9906560	0.9833441	0.9925642	0.9879326	8473214	62	0.9999927	6429805	0	1.0000000
NN1209 S500	0.9906160	0.9833113	0.9925740	0.9879209	8475939	46	0.9999946	6431742	1	0.9999998
Average	0.9790556	0.9752723	0.9917351	0.9833716	8374354.50	55.17	0.9999934	6345552.17	0.83	0.9999999

Table 7.20: Gem3 Multi-seed Results

	Gem3 Seed 2		Gem3 seed 100		Gem3 seed 200		Gem3 seed 300		Gem3 seed 400		Gem3 seed 500	
Map Score	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss
60	8447912	56	9024859	59	9023668	56	9024190	54	9025328	41	9024144	56
59	109290	119	108857	77	108837	68	109255	110	108866	111	108906	93
58	2558	6	2199	2	2192	3	2113	4	2157	3	2121	3
57	3551	11	2952	5	2905	8	2923	7	2919	7	2931	6
56	5822	18	4808	6	4849	10	4849	13	4784	9	4824	11
55	13278	26	10517	20	10243	22	10336	21	10504	28	10368	15
54	5597	22	4581	20	4596	21	4490	13	4630	18	4561	15
53	5931	19	5032	15	5014	10	4919	11	4944	15	4870	10
52	12687	32	10020	23	10150	36	10264	19	10053	21	10104	33
51	5307	26	4405	24	4500	14	4331	16	4472	13	4477	10
50	4824	11	4131	9	4194	18	4254	13	4157	8	4168	9
49	8042	31	6723	14	6549	14	6583	9	6583	24	6701	18
48	5792	18	4857	15	4842	22	4905	10	4738	13	4953	13
47	4468	18	3893	11	3889	7	3909	8	3789	13	3880	28
46	5206	15	4505	17	4370	9	4363	18	4470	18	4596	10
45	5453	31	4549	13	4600	26	4538	20	4395	18	4446	9
44	4234	19	3739	9	3702	20	3768	14	3766	21	3736	18
43	3949	20	3515	15	3686	16	3455	19	3645	23	3583	24
42	4641	25	3910	14	3901	17	3964	15	3963	13	4072	15
41	4677	27	4048	32	4041	16	3966	26	4163	22	3975	17
40	3876	26	3414	17	3419	19	3417	13	3485	19	3387	16
39	4004	31	3467	26	3603	21	3501	21	3621	21	3535	24
38	4468	32	3950	19	3870	33	3914	25	3856	25	3878	26
37	3663	23	3450	31	3448	27	3451	29	3419	27	3497	37
36	3717	37	3426	28	3407	29	3394	35	3482	30	3387	31
35	4126	37	3754	33	3761	35	3700	25	3837	32	3722	44
34	4199	35	3975	37	3810	36	3769	31	3783	36	3786	30
33	3713	38	3511	29	3486	39	3479	41	3518	36	3453	39
32	3905	49	3693	45	3612	40	3684	29	3706	32	3682	29
31	4137	57	3888	41	3923	37	3816	49	3828	39	3899	34
30	3782	52	3788	41	3770	36	3641	49	3682	38	3701	38
29	4066	63	3962	53	3766	49	3962	54	3869	47	3991	56
28	4288	73	4170	53	4183	58	4253	70	4243	63	4055	58
27	4295	79	4234	66	4199	62	4162	65	4150	56	4101	70
26	4331	74	4328	54	4283	74	4239	60	4217	72	4337	61
25	4588	81	4585	69	4703	56	4643	52	4687	56	4600	65
24	4825	92	4878	75	4836	64	4774	74	4808	66	4745	87
23	4829	99	5089	80	4968	65	5156	64	5016	73	5112	74
22	5269	104	5362	106	5492	76	5558	86	5434	94	5410	86
21	5520	131	5786	97	5758	110	5886	99	5554	113	5779	111
20	5612	162	5966	135	5812	148	5980	147	6025	164	5872	140
19	6096	200	6117	191	6246	184	6162	163	6080	176	6212	173
18	5983	265	6394	223	6223	195	6302	196	6185	224	6207	215
17	5923	254	6209	234	6044	229	6341	240	6117	264	6035	227
16	5592	305	5870	279	5810	257	5801	223	5842	261	5837	252
15	5128	316	5433	251	5532	299	5343	307	5362	309	5211	286
14	4676	352	4820	280	4891	279	4875	313	4842	313	4828	306
13	3977	403	4347	309	4125	330	4286	338	4187	315	4319	331
12	3679	407	3760	359	3639	382	3729	355	3741	339	3852	330
11	3293	457	3424	399	3318	407	3601	368	3425	411	3365	392
10	3163	558	3238	468	3182	459	3180	443	3087	456	3082	482
9	2788	556	2981	496	2852	512	2821	486	2947	524	2978	484
8	2700	635	2810	537	2796	539	2765	513	2738	524	2921	513
7	2529	664	2696	595	2642	610	2629	585	2682	605	2670	625
6	2364	763	2460	643	2486	703	2469	678	2389	673	2491	628
5	2085	857	2046	731	2178	733	2053	701	2135	772	2046	743
4	1623	932	1653	792	1591	826	1605	814	1658	858	1632	808
3	1230	1084	1309	942	1258	946	1283	984	1313	945	1210	959
2	1094	1441	1075	1359	1147	1399	1150	1389	1132	1380	1157	1351
1	939	2645	1004	2316	1002	2310	999	2273	976	2239	961	2185
0	438693	726994	375205	233434	376855	234220	386317	223628	384435	225015	385716	225066

Table 7.21: BWA Multi-seed Results

	BWA Seed 2		BWA Seed 100		BWA Seed 200		BWA Seed 300		BWA Seed 400		BWA Seed 500	
Map Score	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss
60	8191443	1439	8785144	3786	8784859	3805	8787589	1585	8787168	1562	8789672	1598
59	9878	3	9734	15	9863	10	10057	2	9917	2	9993	2
58	9805	3	9883	12	9743	14	9892	2	9785	3	9760	3
57	10004	2	9800	9	9792	18	9666	2	9711	0	9892	1
56	9668	1	9565	13	9420	14	9657	4	9568	0	9562	2
55	9542	4	9576	10	9542	19	9395	2	9441	2	9472	1
54	9422	4	9303	12	9137	11	9304	4	9457	2	9239	1
53	9439	2	9336	16	9191	10	9090	3	9272	0	9082	0
52	9148	1	9363	15	9133	15	9200	2	9205	4	9252	4
51	8809	1	8808	22	8709	23	8896	0	8793	0	8843	4
50	9031	2	9096	10	8993	21	9041	2	8881	2	8954	5
49	8650	3	8582	20	8691	19	8412	2	8681	2	8725	4
48	8956	2	8908	27	8748	14	8651	3	8880	2	8730	3
47	9020	2	9083	14	9055	21	9071	2	9072	3	9219	2
46	8119	2	8083	20	8087	36	8084	2	8110	3	8341	2
45	8612	3	8599	19	8718	28	8849	1	8369	3	8567	2
44	7539	2	7450	20	7364	18	7275	6	7425	1	7482	2
43	8103	2	8141	20	8092	25	8124	0	8222	2	8152	3
42	8413	1	8292	30	8247	22	8278	2	8359	2	8201	1
41	8577	3	8461	23	8723	26	8369	4	8546	2	8597	4
40	8597	1	8540	19	8745	30	8464	3	8518	4	8506	1
39	7750	1	7627	28	7582	31	7720	4	7612	2	7612	3
38	7698	0	7705	22	7793	37	7643	2	7922	1	7774	1
37	8416	8	8283	31	8225	32	8474	3	8452	2	8256	0
36	8036	1	8144	26	7924	18	7892	2	7974	2	7899	4
35	8900	3	9076	32	8874	30	8921	1	9000	4	8871	2
34	7134	5	7191	32	7061	26	7202	3	7060	3	6895	2
33	7652	2	7564	34	7614	32	7554	2	7623	7	7449	5
32	8793	3	8470	30	8420	28	8328	2	8428	2	8488	4
31	8335	2	8131	33	8286	40	8444	4	8183	2	8254	2
30	9433	6	9459	31	9405	26	9648	4	9648	5	9547	0
29	6701	2	6764	37	6819	42	6757	4	6815	4	6797	3
28	7095	2	7057	38	7154	31	7263	1	7215	6	7256	1
27	8298	5	7975	38	8061	42	8180	4	8173	3	8143	3
26	10352	5	10084	30	9860	39	10147	3	10004	1	9999	3
25	11079	7	11335	42	11030	32	11033	2	11227	6	11243	5
24	6639	4	6590	47	6720	43	6637	3	6577	8	6572	3
23	7218	2	7072	37	6910	55	7069	6	7141	7	7128	6
22	8707	2	8561	59	8517	55	8606	8	8508	2	8579	6
21	12885	8	12520	59	12609	65	12510	9	12655	6	12618	8
20	14365	8	14328	61	14473	47	14523	19	14545	12	14569	9
19	6635	6	6621	47	6527	45	6499	15	6521	6	6450	4
18	7202	11	7018	42	7128	58	7214	10	7213	9	7191	9
17	8486	10	8306	68	8337	80	8227	12	8330	14	8211	8
16	18344	19	17286	131	17165	120	17323	28	17342	24	17223	24
15	20757	28	20678	81	20726	92	20726	24	20703	20	20586	25
14	6830	22	6455	80	6679	89	6453	11	6484	12	6614	11
13	7753	28	7185	89	7267	80	7226	29	7322	23	7235	21
12	7814	35	7443	104	7581	120	7613	26	7477	28	7462	28
11	21931	78	19387	336	19724	343	20123	60	20114	54	20201	59
10	43035	152	40565	332	40205	336	40720	143	40835	115	40360	144
9	6362	65	5694	208	5871	182	5941	47	5866	58	6125	57
8	7867	106	6559	126	6600	131	6577	89	6565	85	6678	109
7	187	7	135	31	154	26	151	8	167	8	184	8
6	36	1	32	10	35	10	32	0	37	1	42	3
5	1002	20	778	26	875	27	845	16	822	25	813	18
4	5078	101	4301	72	4287	78	4229	79	4334	98	4257	83
3	8472	116	7034	151	6910	143	7003	122	6914	106	6998	81
2	66170	567	53043	445	52600	480	52958	497	52459	497	52303	520
1	4677	122	3885	165	4071	193	3995	141	4059	130	3942	146
0	472252	729796	136080	496409	135322	498164	276988	356166	275738	357557	274717	357145

Table 7.22: MiniMap2 Multi-seed Results

	MiniMap2 Seed 2		MiniMap2 Seed 100		MiniMap2 Seed 200		MiniMap2 Seed 300		MiniMap2 Seed 400		MiniMap2 Seed 500	
Map Score	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss
60	7551074	193	8162725	1248	8162428	1269	8163689	226	8164227	239	8165545	201
59	17869	1	17777	2	17705	2	17402	0	17347	0	17525	1
58	16908	0	16428	7	16603	6	16662	2	16369	0	16451	0
57	16148	1	16647	2	16009	6	16358	0	16323	1	16164	0
56	16191	0	16312	3	16244	1	16242	0	16138	0	16601	0
55	15996	0	15754	3	15746	5	15754	1	15694	0	15651	1
54	23839	0	22246	5	22703	6	22734	1	22632	1	22219	2
53	24833	1	23525	7	23490	5	23728	2	23833	1	23545	1
52	17215	0	16845	1	17024	7	16892	0	17131	0	17176	0
51	17362	2	16925	3	16851	4	16690	1	16856	0	16640	2
50	16323	0	16082	3	16044	4	16156	0	15858	1	16018	0
49	16165	0	15907	1	15953	7	16091	0	16076	1	16069	1
48	17208	0	16850	3	16806	5	16798	0	16941	1	16738	1
47	16189	1	15935	2	15851	3	15978	0	15861	0	15972	0
46	16226	0	15964	2	16086	6	16033	3	16118	0	15891	0
45	15770	1	15397	4	15812	7	15707	1	15791	1	15598	0
44	15368	1	15262	4	15016	6	15189	0	15357	2	15044	2
43	15034	0	14767	4	14806	2	15026	1	14858	2	14789	0
42	17143	1	16326	1	16411	3	16507	1	16523	0	16641	0
41	14809	0	14900	6	14793	1	14754	0	14657	2	14651	1
40	14707	0	14591	4	14683	5	14570	0	14544	0	14814	0
39	14377	1	14366	3	14484	5	14352	2	14247	1	14238	0
38	14504	1	14192	6	14350	5	14475	0	14463	0	14332	0
37	14303	3	14392	2	14324	5	14001	2	14429	2	14245	1
36	41404	3	36893	13	37233	8	37052	4	37469	3	37509	3
35	21977	1	20574	9	20665	7	20924	0	20743	2	20619	1
34	16032	0	15575	11	15645	8	15695	2	15660	2	15830	1
33	15493	0	15452	4	15132	6	15290	2	15358	1	15223	0
32	14600	0	14235	11	14065	3	14286	2	14436	3	14199	0
31	14700	1	14496	2	14448	10	14455	0	14567	1	14527	1
30	15418	2	15347	5	15026	7	15084	1	15360	3	15329	0
29	14780	1	14472	5	14698	11	14518	4	14578	1	14423	0
28	14589	4	14330	10	14296	11	14246	3	14057	3	14318	3
27	13958	1	13919	6	13792	9	13858	1	13972	3	13933	4
26	13831	3	13825	5	13757	5	13776	3	13907	5	13837	6
25	13903	2	14018	8	13948	12	13746	6	13755	2	13747	8
24	13986	2	13705	9	13549	17	13637	10	13471	2	13627	4
23	13531	5	13393	6	13358	15	13378	5	13304	5	13457	1
22	13880	3	13749	10	13722	12	13502	3	13603	4	13598	5
21	15035	8	14479	16	14701	12	14753	9	14758	7	14793	7
20	14487	8	14018	16	14081	20	14159	7	14057	9	14245	4
19	14183	12	13905	7	13964	23	14090	9	13986	9	13870	9
18	14362	15	14060	18	13945	18	14319	12	14215	14	14281	14
17	13958	18	13762	20	13798	26	13770	18	13728	17	13825	19
16	14393	17	14086	26	14046	26	14124	15	13985	24	13908	17
15	57122	87	46151	41	45905	34	46411	74	45856	70	45868	91
14	19592	23	18250	34	18064	37	18245	31	17889	40	18085	37
13	26840	101	24775	44	24415	62	24765	69	24527	71	24389	80
12	22019	111	21158	63	20836	53	20754	89	21272	100	20687	126
11	20217	146	19118	85	19259	61	18821	140	18912	137	18985	142
10	22205	191	19610	88	19711	76	19341	170	19557	171	19580	179
9	17105	107	16469	80	16532	98	16282	93	16560	78	16521	95
8	17536	117	16762	103	17035	121	16673	122	16753	130	16788	128
7	17801	167	16992	124	16950	136	17162	138	16820	123	16952	118
6	17510	164	16953	159	16984	166	16966	171	16774	168	16603	141
5	18937	230	17952	202	17844	201	17925	198	17903	190	18061	255
4	18232	300	17348	274	17274	264	17346	282	17261	287	17341	279
3	20916	441	19883	410	19854	461	19795	424	19684	418	19799	408
2	22707	621	21812	648	21984	623	21837	632	21869	661	21782	629
1	66015	8621	61638	7842	61062	7758	61102	7680	61321	7776	61465	7955
0	374686	954759	195736	569245	195603	570805	176231	589222	175911	589094	175090	589365

Table 7.23: XGBoost Multi-seed Results

	XGBoost Seed 2		XGBoost Seed 100		XGBoost Seed 200		XGBoost Seed 300		XGBoost Seed 400		XGBoost Seed 500	
Map Score	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss
60	0	0	0	0	0	0	0	0	0	0	0	0
59	0	0	0	0	0	0	0	0	0	0	0	0
58	0	0	0	0	0	0	0	0	0	0	0	0
57	0	0	0	0	0	0	0	0	0	0	0	0
56	0	0	0	0	0	0	0	0	0	0	0	0
55	0	0	0	0	0	0	0	0	0	0	0	0
54	0	0	0	0	0	0	0	0	0	0	0	0
53	0	0	0	0	0	0	0	0	0	0	0	0
52	0	0	0	0	0	0	1	0	0	0	0	0
51	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0
49	0	0	0	0	1	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0	0
47	0	0	0	0	0	0	1	0	1	0	0	0
46	0	0	2	0	1	0	0	0	1	0	4	0
45	0	0	0	0	2	0	0	0	1	0	0	0
44	1	0	4	0	1	0	2	0	2	0	1	0
43	1	0	1	0	2	0	5	0	1	0	1	0
42	1	0	3	0	3	0	4	0	2	0	2	0
41	3	0	8	0	8	0	7	0	7	0	8	0
40	11	0	15	0	11	0	14	0	18	0	16	0
39	20	0	25	0	27	0	31	0	27	0	33	0
38	32	0	40	0	38	0	46	0	32	0	37	0
37	56	0	61	0	63	0	67	0	55	0	45	0
36	96	0	106	0	119	0	100	0	102	0	88	0
35	127	0	126	1	134	0	154	0	153	1	133	0
34	189	0	191	0	185	1	205	1	207	1	166	2
33	269	0	266	1	253	0	271	0	289	0	304	0
32	422	0	481	1	457	0	436	1	448	1	404	0
31	665	1	687	0	669	0	641	0	671	1	730	1
30	1391	1	1285	0	1325	0	1366	1	1410	6	1327	3
29	2268	0	2408	0	2421	0	2354	0	2468	4	2399	2
28	4473	0	4343	1	4505	1	4455	2	4439	7	4454	8
27	10040	3	10077	1	10071	1	10035	0	10019	4	9958	6
26	22035	0	21784	2	22008	2	21979	4	21649	14	21693	15
25	45245	3	44419	8	44078	4	44315	9	44185	37	44294	32
24	119165	7	116394	15	117392	15	116393	10	116979	79	117190	63
23	413454	59	412385	40	413319	32	412085	35	412689	245	411988	207
22	7764443	113	8350169	85	8348059	93	8350086	95	8346183	1060	8350118	1128
21	235701	118	223673	117	223859	102	223576	101	224832	304	224362	337
20	93357	124	88918	93	88115	91	89303	97	88907	210	88314	211
19	37588	109	36179	79	35789	106	35814	89	35470	150	35468	133
18	22637	108	21414	110	21474	115	21448	93	21765	144	21163	134
17	15754	146	15217	120	15042	134	15123	132	14945	140	15067	141
16	11626	162	11330	168	11349	160	11418	155	11150	187	11130	166
15	8583	199	8447	175	8491	170	8251	132	8429	163	8188	210
14	6366	212	6555	171	6368	204	6388	192	6398	203	6537	231
13	7400	347	7583	296	7515	305	7648	306	7626	360	7433	374
12	6832	420	7241	390	6961	358	7033	363	7203	365	6911	393
11	8762	759	9093	650	9107	615	9012	596	8849	680	8995	650
10	6843	799	7111	660	7027	711	7048	676	7144	706	7108	731
9	3588	498	3658	471	3702	456	3547	457	3607	459	3589	462
8	3206	646	3254	558	3235	586	3297	538	3265	529	3300	564
7	2878	841	2961	709	2884	683	2913	678	2953	632	2861	703
6	2008	753	2119	603	2002	637	2140	690	2152	613	2139	642
5	2519	1254	2494	987	2580	1025	2508	965	2561	995	2541	1114
4	5707	4633	4657	1418	4682	1390	4639	1452	3525	2532	3529	2575
3	244769	238314	217821	25285	217438	25502	218789	25331	120886	122926	120090	121846
2	106021	177995	82305	63383	83058	63767	83660	62952	58797	87933	58487	87929
1	38011	169824	23883	56957	24446	57393	32057	48762	25425	55780	25625	55892
0	3424	143565	2434	92818	2378	92687	2800	91620	11517	83085	11552	83313

Table 7.24: NN1204 Multi-seed Results

	NN1204 Seed 2		NN1204 Seed 100		NN1204 Seed 200		NN1204 Seed 300		NN1204 Seed 400		NN1204 Seed 500	
Map Score	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss
29	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0
27	1	0	2	0	0	0	0	0	2	0	0	0
26	7	0	1	0	5	0	1	0	3	0	5	0
25	36	0	33	0	37	0	44	0	40	0	35	0
24	452	0	436	0	467	0	435	0	448	0	403	0
23	3482	0	3490	1	3576	0	3573	0	3668	0	3573	0
22	1615782	1	1795003	0	1792267	0	1790875	0	1794453	0	1794802	0
21	5707061	4	6094979	2	6094629	2	6097856	4	6092559	1	6094294	2
20	968032	63	993590	65	994257	68	993810	61	994746	70	994979	46
19	269695	101	261829	87	263145	64	262540	98	263213	89	262690	73
18	82226	198	78268	174	78430	192	78314	163	78487	162	78434	169
17	90895	428	76934	331	76485	357	76692	335	76407	335	76568	381
16	33849	401	29535	351	29657	381	29972	364	29808	374	29591	392
15	27419	536	26042	467	25693	506	25826	437	25523	445	25610	475
14	13333	528	13537	456	13541	482	13507	487	13615	450	13327	446
13	7973	577	8294	526	8053	524	7984	499	8009	511	8149	458
12	7218	742	7216	607	7223	640	7132	630	6818	628	6903	665
11	5524	646	5676	600	5742	650	5698	581	5700	602	5746	603
10	6912	958	7185	771	7141	809	7048	847	7122	799	6921	820
9	4196	896	4510	812	4449	772	4378	793	4491	805	4483	840
8	4107	1324	4219	1076	4067	977	4037	1064	4086	1043	4032	1063
7	4064	1480	4149	1082	4171	1082	4059	1129	4091	1023	4131	1114
6	7249	2242	7372	1729	7382	1695	7441	1557	7421	1739	7502	1710
5	2881	3721	3022	2547	3044	2650	3216	2503	2959	2525	3071	2530
4	11144	14121	9822	7145	9949	7282	10106	7228	10138	7146	9970	7313
3	321780	381723	273354	71735	274208	71786	275621	71136	274814	71908	272808	71815
2	57050	189172	40417	64835	40362	65105	48320	56714	48525	56970	48599	56871
1	4082	23311	3776	15559	3762	16153	3956	15297	3872	15503	4075	15606
0	1537	118840	936	75415	912	75169	1024	74608	1057	74797	1118	74789

Table 7.25: NN1209 Multi-seed Results

	NN1209 Seed 2		NN1209 Seed 100		NN1209 Seed 200		NN1209 Seed 300		NN1209 Seed 400		NN1209 Seed 500	
Map Score	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss
29	0	0	0	0	0	0	0	0	0	0	0	0
28	1	0	2	0	0	0	0	0	2	0	0	0
27	5	0	1	0	4	0	0	0	3	0	5	0
26	15	0	11	0	20	0	23	0	17	0	19	0
25	159	0	156	0	155	0	154	0	166	0	131	0
24	910	0	907	0	955	0	945	0	961	0	894	0
23	3214	0	3160	1	3116	0	3258	0	3256	0	3129	0
22	149545	0	147713	0	147108	0	147006	0	146971	0	147363	0
21	5765494	1	6278908	1	6278062	0	6280759	1	6278429	0	6280201	1
20	1953073	48	2044772	53	2044515	63	2042848	55	2043409	62	2044197	45
19	446178	74	439999	55	440349	50	441278	51	441859	52	441267	44
18	236629	78	230075	77	230352	62	229498	88	230254	86	229848	81
17	79902	182	74369	172	74361	186	74256	178	74611	173	74616	171
16	35147	244	34770	246	34762	233	34787	245	34548	225	34771	249
15	81141	436	66189	348	65844	366	66150	322	65991	353	65566	383
14	30747	487	27163	389	27069	413	27072	402	26802	408	26794	373
13	22422	518	20243	478	20174	484	20036	434	20181	440	20086	455
12	15193	560	14154	478	14184	490	14005	500	14118	450	14071	478
11	8980	557	8485	464	8547	481	8613	468	8455	478	8473	455
10	5688	512	5717	497	5744	515	5574	470	5674	478	5602	511
9	4405	671	4649	571	4453	595	4469	602	4352	524	4444	570
8	3822	893	3792	765	3777	737	3682	765	3835	758	3734	762
7	7788	1126	7793	985	7710	968	7756	925	7843	978	7798	977
6	3354	877	3493	735	3358	720	3430	725	3403	716	3432	795
5	2086	1294	2071	1043	2150	1110	2068	1030	2073	1034	2145	1061
4	2525	1601	2536	1460	2447	1401	2626	1395	2550	1413	2456	1471
3	301742	344910	253686	53775	254310	53947	255392	53511	254681	54286	252967	54099
2	89834	227816	73329	84844	73745	85346	80874	77659	80697	78006	80796	78117
1	7040	44027	4936	25920	4868	26483	6407	24332	6389	24421	6497	24570
0	948	115101	548	73016	515	72696	499	72377	545	72584	517	72513

Table 7.26: The Highest Percent Accuracies possible for each respective tool. Gem3, BWA, MiniMap2, XGBoost and Neural Network Models have been averaged from seeds 2, 100, 200, 300, 400 and 500. TP and FP values have been rounded to the nearest integer. Same as the table 4.1, except containing the specific ranges.

Method	Map Score Range	TP	FP	Percent Accuracy
Support Vector Machine	Just a score of 60	8942055	105192	0.9883730
BWA	Just a score of 60	8687646	2296	0.9997368
Random Forest	Just a score of 60	8055968	749	0.9999070
SNAP	Just a score of 70	8267170	676	0.9999182
MiniMap2	Just a score of 60	8061615	563	0.9999308
Gem3	Just a score of 60	8928359	54	0.9999940
LR with Regularization	Just a score of 60	8015468	46	0.9999943
XGBoost + GEM3 + LR	Just a score of 60	8001311	30	0.9999963
NN 1204	21 & up.	7798136	3	0.9999996
NN 1209	21 & up.	6345552	1	0.9999999

Table 7.27: SVM Results with varying sample sizes to learn from, performed on on the 10MR sample

Number of Samples	Hit or Miss Accuracy
10000	97
30000	98.7
50000	98.8

Table 7.28: Possible predictor values per column for a sample Chromosome 1 test generated by Gem3

Seemingly Useful Columns	Range From	Range To	Number of unique items
2	-1.000	51.000	40
3	-101.000	12.000	117
4	0.267	1.000	118
5	0.840	1.000	34
6	0.133	1.000	157
11	0, 1.000	1.000	124
12	0, 1.000	1.000	46
13	0, 1.000	1.000	219
15	0, 1.000	22.000	22
16	0, 1.000	81.000	76
17	0, 1.000	11.000	12
20 & 21	0.857	9.929	139
Seemingly Useless Columns			
7	93.000	n/a	1
8	1.000	n/a	1
9	0.000	n/a	1
10	0.000	n/a	1
14	0.000	1.000	2
18	0.000	n/a	1
Uncertain Columns			
19	10.571	10.714	3
22	-8.837	7.876	2463246

Table 7.29: Sample times taken to run tools on sample hardware all on CPU mode except for the Neural Network ran on the 1080Ti.

	Relevant Hardware	
	Intel Xeon E312xx	Mobile i7-3610QM + SSD
Tools	Time(Seconds)	
Gem3 w/o Predictors	177	n/a
MiniMapper2	1312	n/a
BWA	5946	n/a
SNAP	>6000	n/a
BowTie2	>6000	n/a
	Theoretical time (s) it takes/would take after Running Gem3	
Logistic Regression	0	
XGBoost	351	216
Neural Network	n/a	1418

Table 7.30: Number of samples which were possible to train on 64gb of ram using various Methods

Model	Number of training samples with 64gb of ram	Number of Parameters
Neural network	>100 Million	13
LR	100 Million	15
LrwR	75 Million	15
XGBoost	20 Million	15
Random Forest	1 million	13
SVM	100000	13

7.2 25bp, 50bp and 100bp tests

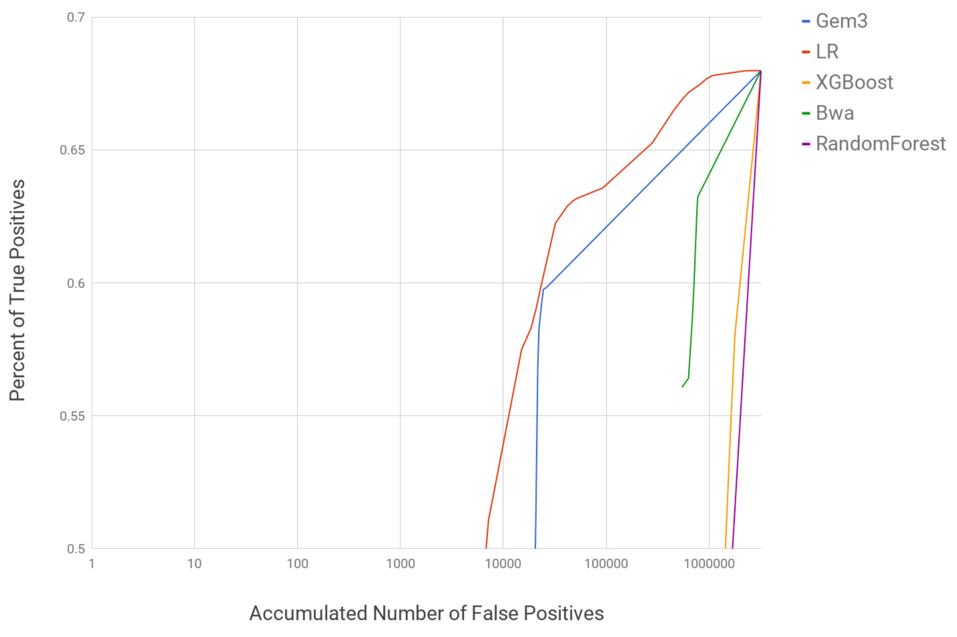

Figure 7.21: Total percent of True positives versus the total number of False positives for multiple tested tools and machine learning methods using 25bp length reads. Data available Appendix 2.

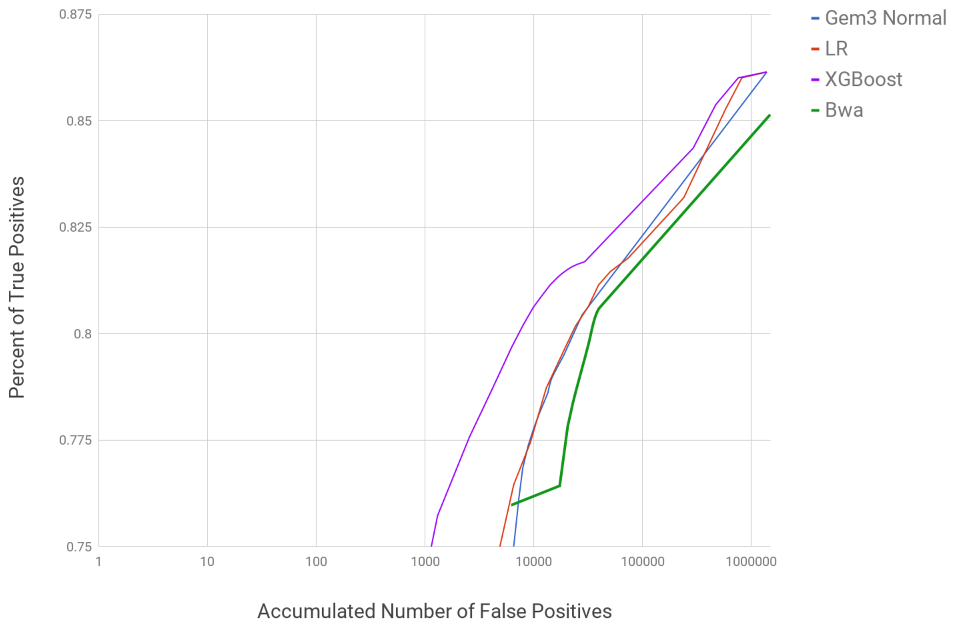

Figure 7.22: Total percent of True positives versus the total number of False positives for multiple tested tools and machine learning methods using 50bp length reads. Data available Appendix 2.

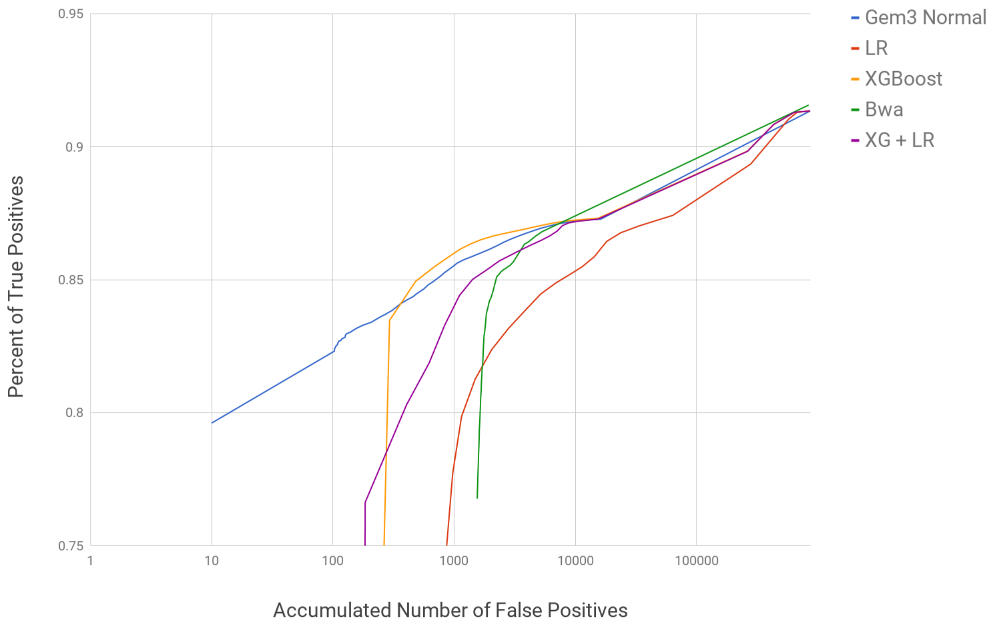

Figure 7.23: Total percent of True positives versus the total number of False positives for multiple tested tools and machine learning methods using 100 bp length reads. Data available Appendix 2.

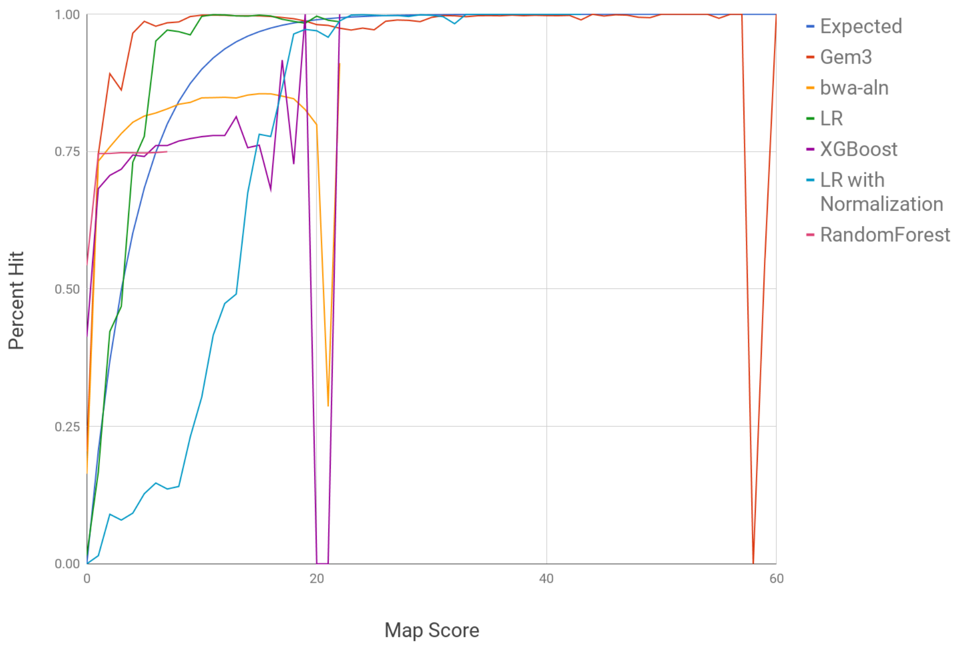

Figure 7.24: Percent hit vs miss for the 25bp reads using various methods.

Table 7.31: 25bp results with varying methods

	Desired 20 to 60 Range				
Mapper	TP from 20 to 60	FP from 20 to 60	% Accuracy for 20:60	Total Hit in 20:60	%Hit All
Random Forest	0	0	0	0	
BWA	5957875	708069	89.40%	6665944	
XGBoost 30M	14	1	93.33%	15	
Gem3	336634	340856	98.70%	677490	
Gem3 LR	5903985	20895	99.65%	5924880	
Specific Cases where improvement is seen Outside of 20 to 60					68%
	Total 10 to 60	% hit from 10	Total Hit from 10	% Hit from a Score of 5	
Random Forest	n/a	n/a	5927517	75%	
BWA	n/a	n/a	n/a	n/a	
XGBoost 30M	5874774	78%	n/a	n/a	
Gem3	5708192	99.64%	n/a	n/a	
Gem3 LR	n/a	n/a	n/a	n/a	

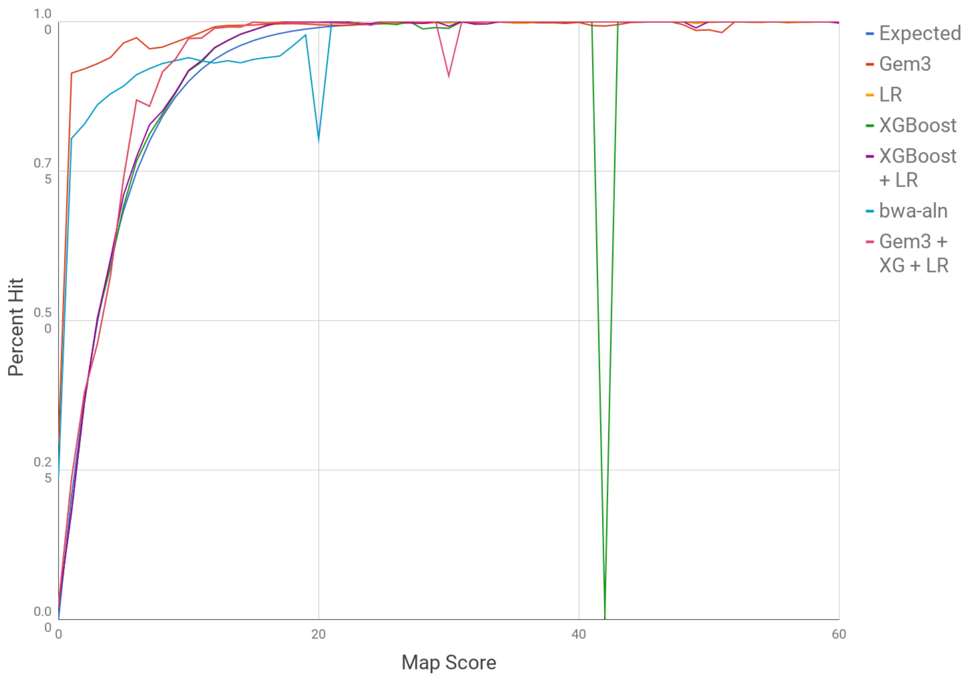

Figure 7.25: Percent hit vs miss for the 50bp reads using various methods.

Table 7.32: 50bp results with varying methods

	Range of 20:60			18:60		Only 60		
Mapper	TP 20:60	FP 20:60	% Accuracy	Total 18:60	% Hit18:60	TP	FP	Only 60 %
BWA	7834726	22724	99.71	7891524	99.69	n/a	n/a	n/a
XGBoost	46181	79	99.83	7168557	99.991	n/a	n/a	n/a
XGB + LR	35678	40	99.88	7158735	99.992	n/a	n/a	n/a
Gem3	6534252	6536393	99.96	6835994	99.95	6054908	2	99.99997
LR	2603778	805	99.97	5634040	99.98	n/a	n/a	n/a
Gem3 + XG + LR	6065202	21	99.9997	6112100	99.998	6054908	2	99.99997

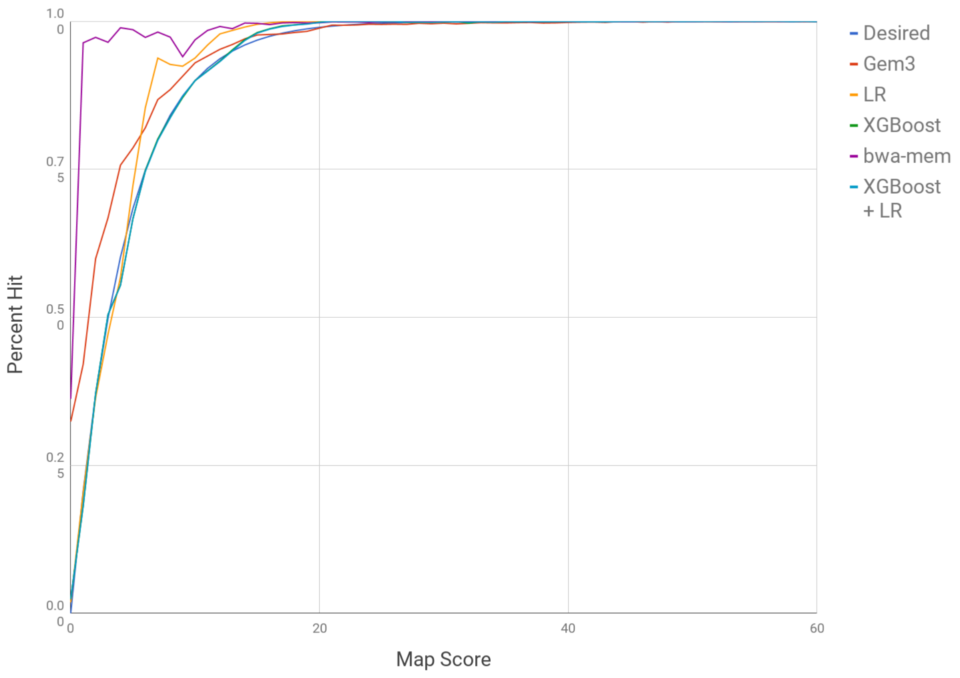

Figure 7.26: Percent hit vs miss for the 100bp reads using various methods.

Table 7.33: 100bp results with varying methods. *XGB + LR + Gem3 results are in the range of 22:60

	Desired 20 to 60			Just a Score of 60		
Map Tool	TP	FP	Percent Accuracy	TP	FP	Percent Accuracy
BWA	8381859	1870	99.978	7676462	1555	99.97974
GEM3	8575783	1215	99.986	7960504	10	99.99987
GEM3 LR	7045652	767	99.989	n/a	n/a	n/a
XG Boost	8494559	484	99.994	n/a	n/a	n/a
XGB+LR	7662743	185	99.998	n/a	n/a	n/a
XGB + LR + Gem3*	8063938	100	99.999	7960504	10	99.99987

Table 7.34: 25bp data. True Positive and False Positive for each Map Score of varying methods.

	Gem3		LR w/ Normalisation		BWA		Random Forest	
Map Score	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss
60	161	0	0	0	0	0	0	0
59	5	4	0	0	0	0	0	0
58	0	1	0	0	0	0	0	0
57	8	0	0	0	0	0	0	0
56	26	0	0	0	0	0	0	0
55	271	2	0	0	0	0	0	0
54	63	0	0	0	0	0	0	0
53	7	0	0	0	0	0	0	0
52	2	0	0	0	0	0	0	0
51	2	0	0	0	0	0	0	0
50	24	0	0	0	0	0	0	0
49	320	2	0	0	0	0	0	0
48	725	4	0	0	0	0	0	0
47	13519	22	0	0	0	0	0	0
46	3064	3	0	0	0	0	0	0
45	927	3	1	0	0	0	0	0
44	97	0	16	0	0	0	0	0
43	674	7	136	0	0	0	0	0
42	20821	44	8	0	0	0	0	0
41	7386	20	0	0	0	0	0	0
40	4017	10	0	0	0	0	0	0
39	10055	19	2	0	0	0	0	0
38	6607	18	127	0	0	0	0	0
37	13452	22	83	0	5607384	547130	0	0
36	8092	23	3	0	0	0	0	0
35	6690	17	13	0	0	0	0	0
34	725	2	5	0	0	0	0	0
33	4051	18	26	0	0	0	0	0
32	14315	41	57	1	0	0	0	0
31	12631	32	229	1	0	0	0	0
30	3312	19	3699	7	0	0	0	0
29	4251	58	6982	7	0	0	0	0
28	25950	290	2690	11	0	0	0	0
27	25705	271	19363	45	0	0	0	0
26	8122	105	16043	40	0	0	0	0
25	6042	176	58657	101	34126	85019	0	0
24	9252	239	889770	713	0	0	0	0
23	13229	391	4108743	6258	230430	57882	0	0
22	22204	578	642375	7888	0	0	0	0
21	38404	795	81239	3545	0	0	0	0
20	51426	986	73718	2278	85935	18038	0	0
19	92406	1146	78808	2233	0	0	0	0
18	171858	1409	242266	9020	52184	9475	0	0
17	305310	1811	64625	9958	35971	6295	0	0
16	533217	2144	11274	3224	27645	4680	0	0
15	846508	2503	8038	2246	39978	6757	0	0
14	1129684	3222	9168	4414	15405	2656	0	0
13	1080698	3134	39351	40812	34792	6251	0	0
12	649650	1199	169467	188289	24420	4347	0	0
11	326451	519	120500	168927	18115	3239	0	0
10	214107	360	46393	106484	22498	4036	0	0
9	140570	581	24788	82523	18736	3578	0	0
8	83762	1187	20173	123117	15406	3017	0	0
7	39292	620	12626	80117	13716	2852	120	40
6	16867	375	8797	51028	12392	2714	41278	13882
5	5973	78	7539	51546	9482	2155	4389722	1482475
4	2191	78	4942	48620	8698	2127	101767	34308
3	1582	253	3377	39053	6779	1879	140370	47258
2	2013	244	4600	46398	5511	1754	192409	65273
1	3168	1081	18165	1209909	4246	1549	130913	44465
0	817014	3174879	73	912232	475106	2423615	1801130	1514590

Table 7.35: 50bp data. True Positive and False Positive for each Map Score of varying methods.

	Gem3		LR		BWA		XGBoost		XGB + LR		XGB + LR + Gem3	
Map Score	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss
60	6054908	2	2974	2	0	0	0	0	777	2	6054908	2
59	47	0	500	0	0	0	0	0	47	0	0	0
58	21	0	946	1	0	0	0	0	69	0	0	0
57	26	0	754	1	0	0	0	0	58	0	0	0
56	38	0	426	1	0	0	0	0	85	0	0	0
55	86	0	550	0	0	0	0	0	66	0	0	0
54	58	0	622	1	0	0	0	0	107	0	0	0
53	86	0	779	1	0	0	0	0	103	0	0	0
52	58	0	1300	0	0	0	0	0	64	0	0	0
51	107	2	1138	1	0	0	0	0	78	0	0	0
50	143	2	675	1	0	0	0	0	107	0	0	0
49	133	2	924	3	0	0	0	0	94	1	0	0
48	198	1	1027	0	0	0	0	0	99	0	0	0
47	344	0	1403	0	0	0	0	0	83	0	0	0
46	562	0	2197	1	0	0	0	0	82	0	0	0
45	961	1	3238	2	0	0	0	0	78	0	0	0
44	1350	2	3000	0	0	0	0	0	105	0	0	0
43	1774	9	5558	3	0	0	0	0	154	0	0	0
42	1468	11	2755	1	0	0	0	1	190	0	0	0
41	1210	8	4648	2	0	0	1	0	212	0	0	0
40	1416	2	9599	3	0	0	0	0	187	0	0	0
39	2006	6	18890	6	0	0	3	0	173	0	1	0
38	3376	7	18632	6	0	0	1	0	159	0	0	0
37	5869	13	7110	4	7597199	6207	3	0	183	0	1	0
36	7180	13	5517	16	0	0	3	0	169	0	1	0
35	8374	20	9069	21	0	0	4	0	194	0	2	0
34	11412	10	17070	9	0	0	11	0	215	0	2	0
33	9631	6	28362	11	0	0	10	0	261	1	2	0
32	9417	19	23521	9	0	0	39	0	227	1	4	0
31	9735	13	10396	17	0	0	41	0	223	0	4	0
30	11267	23	13582	27	0	0	87	1	251	2	10	2001
29	13572	27	17658	34	0	0	99	1	244	0	20	0
28	14744	43	24799	28	0	0	161	2	341	1	41	0
27	16758	37	32412	27	0	0	263	0	515	1	65	0
26	17952	43	21774	52	0	0	800	4	486	0	87	0
25	23214	66	15245	63	45323	11074	685	2	441	0	161	0
24	32389	161	20278	84	0	0	1601	8	293	1	320	2
23	43656	234	25318	51	139080	3181	2034	6	376	1	609	1
22	53540	354	2075068	71	0	0	4150	10	446	0	934	3
21	79761	486	67841	108	0	0	9720	15	644	0	2267	5
20	95405	518	106223	137	53124	2262	26465	29	26992	29	5742	7
19	128098	512	853364	153	0	0	154627	88	154980	88	15729	24
18	170404	587	2175755	185	32332	1742	6967127	455	6967493	456	31103	42
17	198060	749	618191	216	22370	1371	404863	683	405344	683	148000	306
16	186249	854	154129	292	16683	1069	184642	1257	185165	1257	149578	411
15	153222	887	786691	564	24705	1665	116148	1647	116715	1647	786591	699
14	140527	872	148429	1269	9081	672	94950	2056	96133	2058	146940	1385
13	101879	679	126455	1076	20979	1468	51737	1713	52658	1718	124142	1162
12	71194	622	178077	1966	14654	1091	41716	1923	42324	1924	176457	2020
11	51660	935	99114	2761	11072	779	28176	2038	29277	2043	98608	2810
10	41455	1120	128135	3711	13587	872	24637	2224	25031	2226	128406	3765
9	34034	1210	86629	5746	11385	796	17386	2360	17675	2363	87090	5818
8	27867	1240	59009	5393	9585	722	8702	1586	9035	1586	59399	5459
7	21974	1050	45086	7426	8930	762	8866	2053	9862	2055	45291	7475
6	18156	505	51447	7755	8329	812	6793	2070	7076	2071	51562	7783
5	13673	507	32076	11408	6348	768	6059	2709	6634	2712	32149	11422
4	13860	885	30835	22781	5619	772	6368	4427	6700	4427	30888	22795
3	17025	1288	142340	165619	4229	683	267015	262518	267561	262518	142373	165637
2	26648	2289	207980	339293	3116	644	102210	178804	102680	178810	208009	339329
1	93947	8864	73460	238549	2259	549	62573	285019	62732	285024	73481	238589
0	570497	1357523	13701	568351	454447	1445603	13905	629610	13928	629613	13704	568367

Table 7.36: 100bp data. True Positive and False Positive for each Map Score of varying methods.

	Gem3		LR		BWA		XGBoost		XGBoost + LR		XGBoost + LR + Gem3	
Map Score	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss	Hit	Miss
60	7960504	10	12035	6	7676462	1555	0	0	1344	1	7960504	10
59	270573	93	1885	0	21081	2	0	0	210	0	19	0
58	5349	0	2357	1	22473	8	0	0	607	0	32	0
57	4903	1	3933	0	20874	4	0	0	1669	0	21	0
56	4253	1	6182	0	10457	1	0	0	3389	0	43	0
55	3639	1	5178	1	21606	7	0	0	3149	0	73	0
54	3192	1	8103	1	20317	5	0	0	4619	0	57	0
53	2894	2	10204	2	10162	5	0	0	3854	0	64	0
52	2898	1	10196	1	20979	4	0	0	4314	0	81	0
51	2881	0	9897	1	21129	3	0	0	4564	0	80	0
50	3222	1	13683	4	17941	7	0	0	3892	0	88	0
49	3800	1	14883	5	11693	4	0	0	4157	0	103	0
48	4447	5	14943	0	21957	5	0	0	4889	0	142	0
47	5161	2	14864	5	17942	4	0	0	5164	0	177	0
46	5576	7	16332	2	20749	2	0	0	5450	0	193	0
45	6215	1	19272	4	11670	7	0	0	6169	0	337	0
44	6618	2	18810	6	19816	8	2	0	6323	0	333	0
43	6671	11	18140	12	18060	7	0	0	7067	0	349	0
42	6769	7	18651	11	11539	3	1	0	6913	0	433	0
41	7693	10	20940	13	22360	9	4	0	6806	0	661	0
40	9355	16	20919	6	17545	5	3	0	7024	0	589	0
39	6940	17	21141	10	20021	4	7	0	7529	0	445	0
38	6894	19	20030	15	9846	5	5	0	7906	1	593	1
37	8292	15	21681	11	18794	13	18	0	8768	0	1004	0
36	11300	23	21732	14	19834	2	44	0	8647	0	803	0
35	11035	28	22921	12	25183	12	54	0	9075	0	768	0
34	14023	35	21236	28	8849	6	89	0	9098	1	1106	1
33	24769	53	22009	20	17710	5	182	0	10271	0	1741	0
32	13857	45	21453	29	17281	11	319	1	9410	1	1469	1
31	11670	49	23066	21	9161	9	619	0	10613	1	1283	1
30	12043	36	22859	26	19680	11	1026	1	10893	2	1669	2
29	18106	72	23364	39	18691	6	1857	1	12732	2	2441	2
28	15050	48	21157	28	29636	10	2585	3	11339	3	2551	3
27	11657	58	32257	48	8277	7	4696	5	17153	6	3192	6
26	11255	52	319287	32	15049	6	8683	4	313182	6	4948	6
25	11136	56	366714	57	16296	8	17734	12	359942	14	8816	14
24	12311	58	1847831	46	8429	16	48612	16	1374281	19	16766	19
23	11026	67	1226266	56	17346	16	146367	33	567050	35	49964	34
22	12058	79	1230469	64	20531	19	3509484	90	4185694	91	185709	90
21	12595	80	935678	64	36677	31	4604673	128	260495	1	1534	1
20	13153	152	563094	66	7756	18	147495	190	377092	1	1276	1
19	12695	217	453671	103	13289	35	56954	219	366861	221	41368	219
18	12820	245	273329	106	13329	27	36503	215	156784	217	34402	217
17	13033	283	212886	178	14274	37	26505	203	141017	212	57927	212
16	13049	294	140126	340	8423	46	21223	267	113190	274	95902	274
15	12683	298	110723	547	25244	82	15222	292	60568	314	54653	313
14	12139	369	79863	768	55229	155	12957	413	35716	438	31728	438
13	11652	473	67890	1050	7240	90	9542	482	33513	500	27553	499
12	10871	538	62384	1345	10276	89	12868	923	33806	930	24909	929
11	10102	629	41418	1727	10777	167	10371	947	25016	964	19694	963
10	8755	659	31111	2052	374	12	8519	946	18409	962	14880	961
9	6916	706	29695	2431	189	12	6164	911	16863	938	13383	938
8	5280	688	37341	2929	1099	30	4519	874	17892	919	15536	919
7	4320	658	57904	3826	8219	151	2865	711	20793	756	19398	756
6	3482	763	32379	5524	2951	82	2840	957	10271	980	9777	980
5	3362	914	26092	10023	14006	199	3116	1559	6102	1569	6006	1569
4	3055	980	39493	30083	67197	730	6282	5050	8334	5058	8307	5058
3	2970	1475	192733	215901	7851	287	252585	247844	254811	247859	254809	247859
2	2705	1811	169136	295718	23108	640	100413	170519	101269	170520	101269	170520
1	2230	3074	27371	106847	16024	602	45015	203706	45064	203706	45064	203706
0	406734	849075	3439	183099	475954	837745	5614	227842	5614	227842	5614	227842

Table 7.37: Neural Network model generated with 100bp reads using the base settings of NN 1209. *No optimisation. Model generated in under 10 minutes.

				20 to 60			Best		
Model	Precision	Recall	F1 Score	Hit	Miss	Accuracy	Hit	Miss	Accuracy
Gem3	0.9981372	0.8742142	0.9320747	8575783	1215	0.9998583	7960504	10	0.9999987
Quick* Neural Network	0.9338268	0.9759738	0.9544352	8324848	378	0.9999546	68868	0	1.0000000

Table 7.38: Data from an *unoptimized Neural Network generated using 1.5 million 100bp reads, using the base settings of Neural Network 1209.

	Neural Network with 100 bp reads*	
Map Score	Hit	Miss
60	0	0
59	0	0
58	0	0
57	0	0
56	0	0
55	0	0
54	0	0
53	0	0
52	0	0
51	0	0
50	0	0
49	0	0
48	0	0
47	1	0
46	4	0
45	9	0
44	24	0
43	36	0
42	65	0
41	109	0
40	181	0
39	281	0
38	461	0
37	756	0
36	1122	0
35	1667	0
34	2272	0
33	3163	0
32	4267	0
31	5825	0
30	7500	0
29	10254	0
28	13688	0
27	17183	0
26	22754	1
25	35492	3
24	72416	4
23	124911	5
22	687479	12
21	5480246	45
20	1832682	308
19	90956	341
18	82524	371
17	58627	361
16	26220	378
15	19599	323
14	15837	364
13	13681	321
12	18715	588
11	21987	1861
10	9306	722
9	6215	420
8	4856	451
7	4022	408
6	6075	1215
5	4534	1617
4	5086	1102
3	45945	39812
2	283515	342674
1	85870	253155
0	6218	218502