TOWARDS A COMPREHENSIVE DEEP NEURAL NETWORK ARCHITECTURE FOR CAPTURING GENOMIC VARIANT FEATURES

2018-2019 Final Year Project
Interim Report

Wong Yat Sing (3035280790)
Supervisor: Dr. Ruibang Luo
1 Background

Human being has started to explore the mystery of heredity, the passing of traits to one’s offspring, since a long time ago. There is also a popular belief that a person’s trait is closely related to the parents. Nowadays, we know that heredity is highly related to our genome.

Since the 1860s when the DNA was first discovered, human had been making attempts to solve the mystery hiding inside. And finally in 2003, the Human Genome Project was completed and the whole sequence of the human genome was published. It is now an important problem in genetics, the study of heredity, to find out the difference of one person’s genome with the reference genome, as this is what causes the difference between different human beings, in appearance, traits, or even diseases. Hence decoding the information inside our genome is very important, not because we can have the ability to control the appearance of our offspring, but because we can avoid many genetic diseases.

This project will focus in doing variant calling, i.e. to find the difference between one’s genome with another. However, in order to do variant calling, it is necessary to “read”, or to sequence the DNA. Different technologies have been developed to tackle with this problem. One of which being the Illumina technology, is well-developed with reliable high performance. On the other hand, doing variant calling with Single Molecule Sequencing technology is still challenging because of its high error rate. Due to this situation, traditional algorithms cannot deal with the problem and this project hopes to rely on another emerging technology – Artificial Intelligence.

Artificial intelligence is a hot-topic in today’s world, from AlphaGo, which beats the top human players in the world in a specific problem, the chess game “Go”, to image recognition, which is revolutionizing the internet searching engine, to even autonomous cars. Many Artificial Intelligence are based on artificial neural networks, which are composed of artificial neurons or nodes. There are a various types of artificial neural networks, each having its own strength and weaknesses. With the great potential and wide variety of artificial networks, we hope that it can assist us in tackling this challenging problem.
2 Objective

The project aims to develop a solution for the variant calling problem with single molecule sequencing technology data. The target F-score is 95% for SNP (Single nucleotide polymorphisms), a change in the bases of the genome, and 50% for indel, an insertion or deletion of bases in the genome.

3 Methodology

This project will be based on the previous work by the project’s supervisor, Clairvoyante, a deep neural network for variant calling in single molecule sequencing (R. Luo, 2018). More specifically, this project will focus on improving the performance of the work stated above.

This section will first talk about the general design of the program, then the two modification that have been done to Clairvoyante, first, a change in the network architecture, and second, including additional information by modifying the input layer.

3.1 Detailed Design

Input Data

The program will take in raw DNA sequencing alignment data stored in BAM format, reference sequence data stored in FASTA format and a BED format file that contains valid regions. It will then process them into actual data to feed into the neural network.

Each position will be transformed into a 33 by 4 by 4 tensor, where the first dimension represents the position, encoding the information at that position and 16 flanking base-pairs each side. The second dimension represents the different bases (ACGT). The last dimension represents the different variations we would like to capture. While the first layer of the last dimension represent the reference, a high signal at the other layers may signal a high chance that a genetic variation existed at that position.

Fig 3.2 shows an example of an input tensor, where an insertion of “A” exists at the position 16 and an SNP of A to T exists at position 25 (with respect to the interested position (chr21:10344221) at position 16). In this case, the network should be able to tell that there is an insertion of length 1, but not the SNP as it is not in the middle.
Fig 3.1 A visualized input generated from the position 10344221 in chromosome 21.

Due to the limit of time, one type of technology is chosen as an indicator for performance. In this project, the Oxford Nanopore Technology (ONT) is chosen due to its worse performance comparing to other technology (Pacbio) and an ongoing project in HKU-BAL related to ONT. The data that will be used for training and evaluation comes from Nanopore WGS Consortium “rel5” release on sample na12878/HG001 aligned to reference GRCh37 using NGMLR version 0.2.3.

Neural Network

The neural network serves as the major functional component in the program. It will take in the information generated from data mentioned above and give an output determining the variation at a certain position.

Output

The output of the neural network is a 16 dimensional vector which is made up of 4 parts. The first 4 represents the base change, the next 2 represents the zygosity, with another 4 represents the type of variation and the last 6 represents the indel length.
Training

For the training of the neural network, the truth variants from the GIAB (Genome in a Bottle) dataset is used as the positive training data, while random positions are generated from the remaining positions for the negative training data. Positive and negative data are mixed up randomly in 1:9 ratio.

3.2 Modification One – Network Architecture

Clairvoyante suggested the use of a convolutional neural network (CNN) that comprise of 3 CNN layers followed by 2 fully connected layers. We have tried to replace the CNN layers with other types of layers such as the long short-term memory (LSTM) based RNN layer or the slice dense layer. This section will present the 2 best network as milestones in the search that based on the two types of layers respectively.

3.2.1 QLY v1 – Slice dense layer based network

![Diagram of QLY v1 network structure]

The network of QLY v1 comprise of 4 layers with 2 slice dense layer followed by 2 dense layers. In the first layer, the input is “cut” along the position dimension (marked as red in Fig 3.2) then densely connected to the next layer with dimension 30 by 33. In the second layer, it is cut along the other dimension (marked as green in Fig 3.2) and connected to the next layer. The output of layer 2 is then flattened...
and connected to the next two dense layers and the output. A dropout of 0.5 is applied to the third layer.

3.2.2 QLY v2 – RNN(LSTM) based network

![Diagram of QLY v2 network structure]

Fig 3.3 The visualized network structure of QLY v2

The network of QLY v2 comprises of 5 layers with 2 bidirectional LSTM layer followed by a slice dense layer then 2 dense layer. On the other hand, it can be obtained by replacing the first layer in QLY v1 with 2 bidirectional LSTM layer with 128 cells in each direction.
3.2.3 – Comparison between QLY v1, v2 and Clairvoyante v3

In this section, the result is generated by the 3 different networks trained with variants from whole genome, then evaluated using the high confidence region in chromosome 19.

<table>
<thead>
<tr>
<th>Network Type</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Precision</td>
<td>Sensitivity</td>
<td>F-1 score</td>
</tr>
<tr>
<td>SNP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QLY v1</td>
<td>0.8928</td>
<td>0.8001</td>
<td>0.8439</td>
</tr>
<tr>
<td>QLY v2</td>
<td>0.9125</td>
<td>0.8239</td>
<td>0.8658</td>
</tr>
<tr>
<td>Clairvoyante v3</td>
<td>0.8504</td>
<td>0.7341</td>
<td>0.7880</td>
</tr>
<tr>
<td>Indel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QLY v1</td>
<td>0.0447</td>
<td>0.0468</td>
<td>0.0457</td>
</tr>
<tr>
<td>QLY v2</td>
<td>0.2765</td>
<td>0.1328</td>
<td>0.2443</td>
</tr>
<tr>
<td>Clairvoyante v3</td>
<td>0.2390</td>
<td>0.0678</td>
<td>0.1057</td>
</tr>
</tbody>
</table>

Table 3.1 Comparison between QLY v1, v2 and Clairvoyante v3

From the result above, we can see an improvement in both SNP and indel on QLY v2. For the poor performance on indel for QLY v1, we believe that it is caused by right-aligned indels. It is a problem caused by the sequencer and cannot be solved with trivial algorithms. The effect of it is that the location of indel inside the input tensor may not always be in the middle, which the dense layer based QLY v1 should not be able to learn it effectively. In the remaining of the project, this problem is unlikely to be tackled but the effect of it should be aware of.

3.3 Modification Two – Additional Strand Information

The DNA is in a double-stranded structure. The sequence of one direction of strand is related but not the same as the other direction. In usual practice, the sequence of one direction of strand is flipped to match with the sequence of the other strand. This may causes systematic error called “strand bias”.

Clairvoyante did not have the strand information for a single read, i.e. the reads from different direction of strand will be treated as they are the same. Here we try expanded the input size to include the strand information. In the modified version, the 33 by 4 by 4 tensor is replaced with a 33 by 8 by 4 tensor. Where the new second “8” dimension is used to represent the expanded base “A+,C+,G+,T+,A-,C-,G-,T-“. The previously best QLY v2 network is used for the modification.

3.3.1 – Comparison between QLY v2 (strand ver.) and QLY v2

In this section, the result is generated by the QLY v2 with and without strand information as input, trained with variants from whole genome, then evaluated using the high confidence region in the whole genome.

<table>
<thead>
<tr>
<th>Network Type</th>
<th>Precision</th>
<th>Sensitivity</th>
<th>F-1 score</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QLY v2 (strand ver.)</td>
<td>0.9724</td>
<td>0.9216</td>
<td>0.9463</td>
</tr>
<tr>
<td>QLY v2</td>
<td>0.9329</td>
<td>0.8600</td>
<td>0.8950</td>
</tr>
<tr>
<td>Indel</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>QLY v2 (strand ver.)</td>
<td>0.4557</td>
<td>0.1903</td>
<td>0.2684</td>
</tr>
<tr>
<td>QLY v2</td>
<td>0.3527</td>
<td>0.1772</td>
<td>0.2358</td>
</tr>
</tbody>
</table>

Table 3.2 Comparison between QLY v2 (strand ver.) and QLY v2

From the result above, we can see a further improvement in the performance in both SNP and indel. A significant drop in number of false positives contributes to the large improvement in precision. Further investigation observes that a major of the reduced false positives are strand bias, showing that the improvement is indeed come from the expanded input.
4 Future Work

In the following days, we will focus on tackling the problem using a generative network approach. A generative network is usually comprise of a generative network and a discriminative network. We hope that the both network can benefit us as the generative network can possibly help us to solve the problem with very limited training data, while the discriminative network could help us to output a good quality score for the network output.

We will also try to tackle the SNP and indel with two different network as the nature of them are quite different, further observation and network design is required.

5 References