Clairvoyante: a multi-task convolutional deep neural network for variant calling in Single Molecule Sequencing

Author

Ruibang Luo[1,2,*], Fritz J. Sedlazeck[3], Tak-Wah Lam[1], Michael C. Schatz[2]

[1] Department of Computer Science, The University of Hong Kong, Hong Kong
[2] Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA
[3] Human Genome Sequencing Center, Baylor College of Medicine, Houston, TX, USA

[*] Correspondence should be addressed to rbluo@cs.hku.hk

Abstract

The accurate identification of DNA sequence variants is an important but challenging task in genomics. It is particularly difficult for single molecule sequencing, which has a per-nucleotide error rate of ~5%-15%. Meeting this demand, we developed Clairvoyante, a multi-task five-layer convolutional neural network model for predicting variant type (SNP or indel), zygosity, alternative allele and indel length from aligned reads. For the well-characterized NA12878 human sample, Clairvoyante achieved 99.73%, 97.68% and 95.36% precision on known variants, and 98.65%, 92.57%, 77.89% F1-score for whole-genome analysis, using Illumina, PacBio, and Oxford Nanopore data, respectively. Training on a second human sample shows Clairvoyante is sample agnostic and finds variants in less than two hours on a standard server. Furthermore, we identified 3,135 variants that are not yet indexed but are strongly supported by both PacBio and Oxford Nanopore data. Clairvoyante is available open-source (https://github.com/aquaskyline/Clairvoyante), with modules to train, utilize and visualize the model.

Introduction

A fundamental problem in genomics is to find the nucleotide differences in an individual genome relative to a reference sequence, i.e., variant calling. It is essential to accurately and efficiently call variants so that the genome sequence variants that underlie phenotypic differences and disease can be correctly detected[1]. Previous works have intensively studied the different data characteristics that might contribute to higher variant calling performance including the properties of the sequencing instrument[2], the quality of preceding sequence aligners[3] and the alignability of genome reference[4]. Today, these characteristics are carefully considered by state-of-the-art variant calling pipelines to optimize performance[5,6]. However,

most of these analyses were done for short read sequencing, especially the Illumina technology, and require further study for other sequencing platforms.

Single Molecule Sequencing (SMS) technologies are emerging in recent years for a variety of important applications. These technologies generate sequencing reads two orders of magnitude longer than standard short-read Illumina sequencing (10kbp to 100kbp instead of ~100bp), but they also contain 5%-15% sequencing errors rather than ~1% for Illumina. The two major SMS companies, Pacific Biosciences (PacBio) and Oxford Nanopore Technology (ONT) have greatly improved the performance of certain genomic applications, especially genome assembly and structural variant detection[7]. However, single nucleotide and small indel variant calling with SMS remain challenging because the traditional variant caller algorithms fail to handle such a high sequencing error rate, especially one enriched for indel errors.

Artificial Neural Networks (ANNs) are becoming increasingly prominent for a variety of classification and analysis tasks due to their advances in speed and applicability in many fields. One of the most important applications of ANNs has been to image classification, with many successes including MNIST[8] or GoogLeNet[9]. The recent DeepVariant package repurposed the Inception convolutional neural network[9] for DNA variant detection by applying it to analyzing images of aligned reads around candidate variants. At each candidate site, the network computes the probabilities of three possible zygosities (homozygous reference, heterozygous reference, and homozygous alternative), allowing accurate determination of the presence or absence of a candidate variant. However, the DeepVariant network is incomplete as a variant caller as it does not provide the full variant information including the exact alternative allele and variant type. As the authors pointed out in their manuscript, it is also sub-optimal to use an image classifier for variant calling, as valuable information that could contribute to higher accuracy are lost during the image transformation.

In this study, we present Clairvoyante, a multi-task convolutional deep neural network specifically designed for variant calling with SMS reads. We explored different ways to enhance Clairvoyante's power to extract valuable genomic features from the frequent background errors present in SMS. Experiments calling variants in multiple human genomes both at known variant sites and genome-wide show that Clairvoyante is on-par with GATK HaplotypeCaller on Illumina data, and greatly outperforms Nanopolish and DeepVariant on PacBio and ONT data.

Methods

In this section, we first introduce the DNA sequencing datasets of three different sequencing technologies: Illumina, PacBio, and ONT. We then formulate variant calling as a supervised machine learning problem. Finally, we present Clairvoyante for this problem and explain the key deep learning techniques used in Clairvoyante.

Datasets

While most of the variant calling in previous studies were done using a single computational algorithm on single sequencing technology, the Genome-in-a-Bottle (GIAB) dataset[10] first published in 2014 has been an enabler of our work. The dataset provides high-confidence SNPs and indels for a standard reference sample HG001 (also referred to as NA12878) by integrating and arbitrating between 14 datasets from five sequencing and genotyping

technologies, seven read mappers and three variant callers. For our study, we as our truth dataset the latest dataset version 3.3.2 for HG001 (**Supplementary Material, Data Source, Truth Variants**) that comprises 3,042,789 SNPs, 241,176 insertions and 247,178 deletions for the GRCh38 reference genome, along with 3,209,315 SNPs, 225,097 insertions and 245,552 deletions for GRCh37. The dataset also provides a list of regions that cover 83.8% and 90.8% of the GRCh38 and the GRCh37 reference genome, where variants were confidently genotyped. The GIAB extensive project[11] published in 2016 further introduced four standard samples, including the Ashkenazim Jewish sample HG002 we have used in this work, containing 3,077,510 SNPs, 249,492 insertions and 256,137 deletions for GRCh37, 3,098,941 SNPs, 239,707 insertions and 257,019 deletions for GRCh37. 83.2% of the whole genome was marked as confident for both the GRCh38 and GRCh37.

Illumina Data

The Illumina data was produced by the National Institute of Standards and Technology (NIST) and Illumina[11]. Both the HG001 and HG002 datasets were run on an Illumina HiSeq 2500 in Rapid Mode (v1) with 2x148bp paired-end reads. Both have an approximate 300x total coverage and were aligned to GRCh38 decoy version 1 using Novoalign version 3.02.07. In our study, we further down-sampled the two datasets to 50x to match the available data coverage of the other two SMS technologies (**Supplementary Material, Data Source, Illumina Data**).

Pacific Bioscience (PacBio) Data

The PacBio data was produced by NIST and Mt. Sinai School of Medicine[11]. The HG001 dataset has 44x coverage, and the HG002 has 69x. Both datasets comprise 90% P6-C4 and 10% P5-C3 sequencing chemistry and have a sequence N50 length between 10k-11kbp. Reads were extracted from the downloaded alignments and aligned again to GRCh37 decoy version 5 using NGMLR version 0.2.3 (**Supplementary Material, Data Source, PacBio Data**).

Oxford Nanopore (ONT) Data

The Oxford Nanopore data were generated by the Nanopore WGS consortium[12]. Only data for sample HG001 are available to date, thus limiting the "cross sample variant calling evaluation" and "combined sampled training" on ONT data in the Result section. In our study, we used the 'rel3' release sequenced on the Oxford Nanopore MinION using 1D ligation kits (450bp/s) and R9.4 chemistry. The release comprises 39 flowcells and 91.2G bases, about 30x coverage. The reads were downloaded in raw fastq formatted and aligned to GRCh37 decoy version 5 using NGMLR[13] version 0.2.3 (**Supplementary Material, Data Source, Oxford Nanopore Data**).

Variant Calling as Multi-Task Regression and Classification

Assume there are *N* truth variants *V* and their corresponding read alignments *B* as training samples; the training dataset is expressed as $\{x_i, a_i, z_i, t_i, l_i\}_{i=1}^{N}$, where:

- $x_i \in \mathbb{R}_{o\times p\times q}^{B}$ denotes a three-dimensional tensor of real numbers that represents a piled-up form of read alignments in the *i*th sample
- $a_i \in \mathbb{R}_{w}^{V}$ denotes the probability of four possible reference and alternative alleles - A, C, G and T of truth variants (w=4)
- $z_i \in \mathbb{Z}_{x}^{V}$ denotes the zygosity (homozygous or heterozygous) of truth variants (x=2)
- $t_i \in \mathbb{Z}_{y}^{V}$ denotes the variant type (reference, SNP, insertion or deletion) of truth variants (y=4)

- $l_i \in \mathbb{Z}_z^V$ denotes the length of Indel, or 0 otherwise, of truth variants (z is set to 6 in our study to support six different Indel lengths including 0, 1, 2, 3, 4 and ≥4bp).

Our target is to infer the functional mapping $\mathrm{F} : \mathbb{R}_{o\times p\times q}^B \rightarrow (\mathbb{R}_w^V, \mathbb{Z}_x^V, \mathbb{Z}_y^V, \mathbb{Z}_z^V)$ that fits $\{x_i, a_i, z_i, t_i, l_i\}_{i=1}^N$, which can be viewed as a multi-task regression and classification problem.

We use mean squared error on a_i and cross-entropy loss on z_i, t_i and l_i to evaluate the predictive performance of each truth variant v,

$$cost = \frac{1}{N'}\sum_{i=1}^{N'}\left(\sum_w (a_{iw} - \hat{a}_{iw})^2 - \sum_x \hat{z}_{ix}\log(z_{ix}) - \sum_y \hat{t}_{iy}\log(t_{iy}) - \sum_z \hat{l}_{iz}\log(l_{iz})\right)$$

, where N' is the number of testing samples and the variables with a caret are predicted values of the ith truth variant. We define cost as the average of all sample losses and use it as the cost function for model training.

In our study, good performance implies correct predictions could be made even when the evidence is marginal to distinguish a genuine variant from non-variant (reference). To achieve the goal, we paired each truth variant with two non-variants randomly sampled from the genome at all possible non-variant and non-ambiguous sites for model training. With about 3.5M truth variants from the GIAB dataset, about 7M non-variants are added as samples for model training.

We randomly partitioned all samples into 90% for training and 10% for validation. We intentionally did not hold out any sample of the data for testing as other projects commonly do because, in our study, we can use an entirely different dataset for testing samples. For example, we can use the samples of HG002 to test against a model trained on HG001, and vice versa.

Clairvoyante

Clairvoyante is a multi-task five-layer convolution neural network with the last two layers as feedforward layers (**Figure 1**). The multi-task neural network makes four groups of predictions on each input: 1) alternative alleles, 2) zygosity, 3) variant type, and 4) indel length. The predictions in 2, 3 and 4 are mutually exclusive while the predictions in 1 are not. The alternative allele predictions are computed directly from the first fully connected layer (FC4), while the other three group of predictions are computed from the second fully connected layer (FC5). The indel length prediction group has six possible outputs indicating an indel with a length between 0-3bp or ≥4bp of any unbounded length. The prediction limit on indel length is configurable in Clairvoyante and can be raised when more training data on longer indels could be provided. The Clairvoyante network is succinct and fine-tuned for the variant calling purpose. It contains only 1,631,496 parameters, about 13-times fewer than DeepVariant [14] using the Inception-v3 network architecture, which was originally designed for general purpose image recognition. Additional details of Clairvoyante are introduced in the different subsections below.

For each input sample (truth or candidate variants), the overlapping sequencing read alignments are transformed into a multi-dimensional tensor of order 33 by 4 by 4. We added 16 flanking base-pairs on both sides of a candidate (yielding 33 bp of total context), which we have measured to be sufficient to manifest background noise while providing a good computational efficiency. For each base-pair, we use one-hot encoding and its depth as the

value that assigns the depth to the active base and other inactive bases as zero. The first tensor in the third dimension encodes the reference sequence and the number of reads supporting the reference alleles. The second, third and fourth tensors use the relative count against the first tensor: the second tensor encodes the inserted sequences, the third tensor encodes the deleted base-pairs, and the fourth tensor encodes alternative alleles. **Figure 2** illustrates how the tensors can represent a SNP, an insertion, a deletion, and a non-variant (reference), respectively. The non-variant in figure 2 also depicts how the matrix will show background noise. A similar but simpler read alignment representation was proposed by Jason Chin[15] in mid-2017, the same time as we started developing Clairvoyante. Different from Chin's representation, ours decouples the substitution and insertion signal into separate arrays and allows us to precisely record the allele of inserted sequence.

Our study used the widely adopted Tensorflow[16] as its primary programming framework. Using the 44x coverage HG001 PacBio dataset as an example, a near optimal model can be trained in three hours using the latest desktop GPU model nVidia GTX 1080 Ti. Using a trained model, about two hours is needed to call variants genome-wide using a 2 x 14-core CPU-only server, and it takes only a few minutes to call variants at known variant sites or in an exome (>5,000 candidate sites per second). Several techniques have been applied to minimize computational and memory consumption (see the Computational Performance subsection).

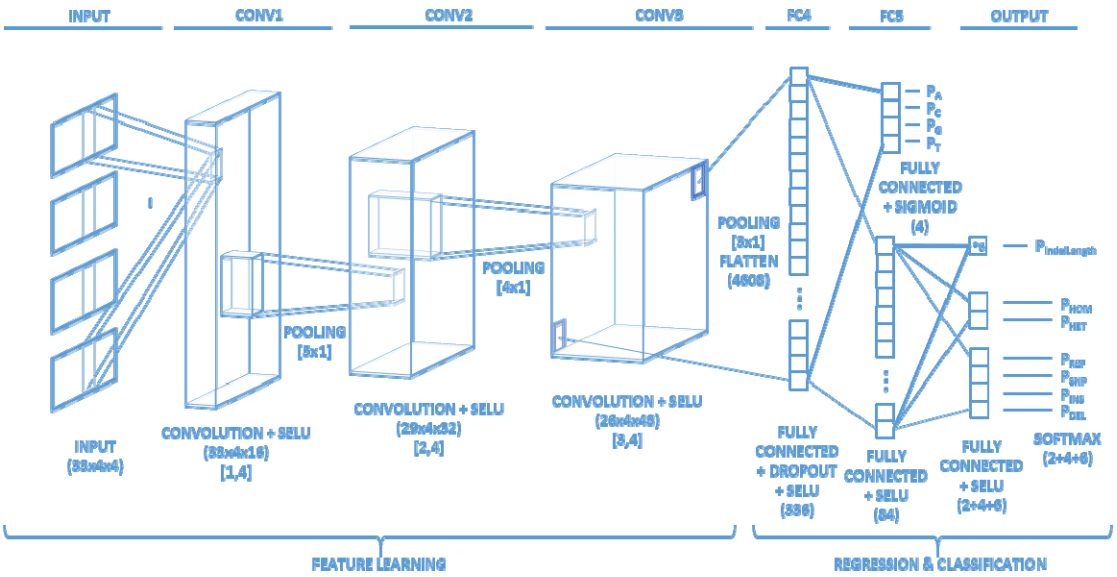

Figure 1. Clairvoyante network architecture and layer details. The descriptions under each layer includes 1) the layer's function; 2) the activation function used; 3) the dimension of the layer in parenthesis (Input layer: Height x Width x Arrays, Convolution layer: Height x Width x Filters, Fully connected layer: Nodes), and; 4) kernel size in brackets (Height x Width).

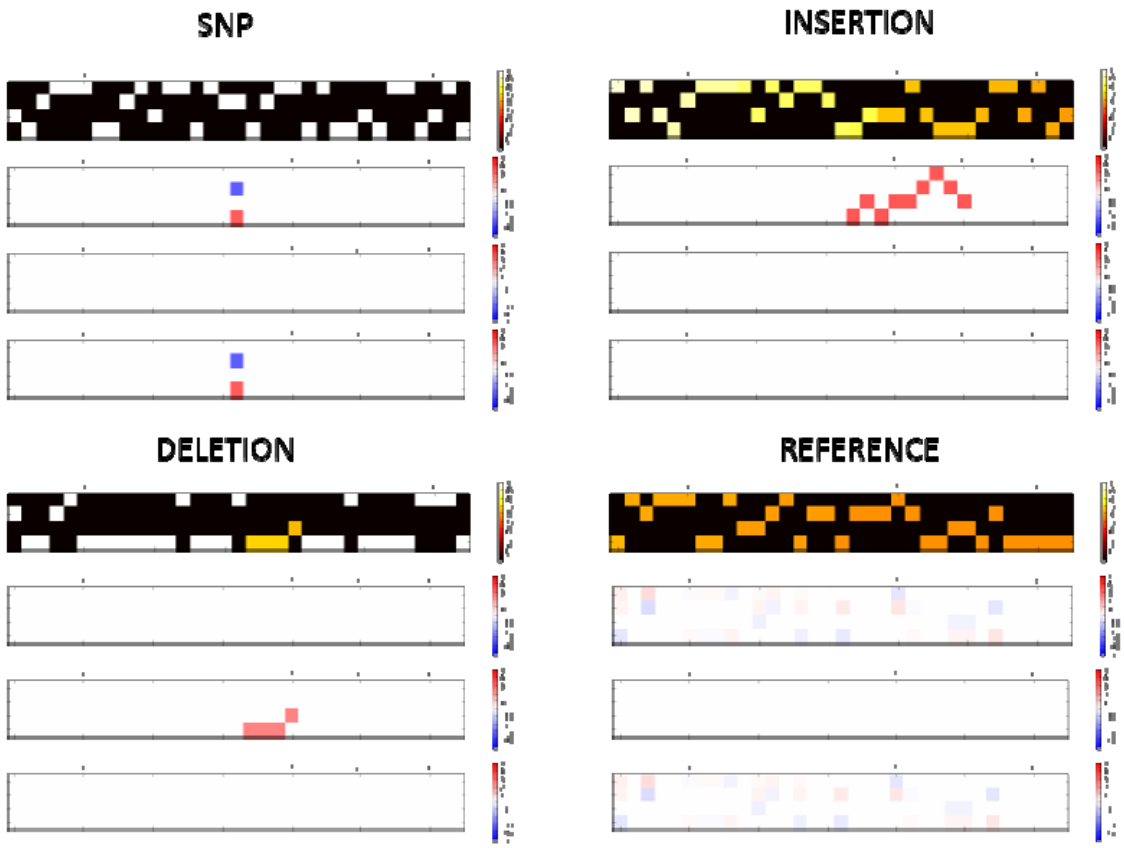

Figure 2. Illustrations of how Clairvoyante representations four variant types including: (**top left**) a C>G SNP, (**top right**) a 9bp insertion, (**bottom left**) a 4bp deletion, and (**bottom right**) a non-variant with reference allele.

Model Initialization

Weight initialization is important to stabilize the variances of activation and back-propagated gradients at the beginning of model training. We used a He initializer[17] to initialize the weights of hidden layers in Clairvoyante, as the He initializer is optimized for training extremely deep models using rectified activation function directly from scratch. For each layer, the weight of each node is sampled from a univariate normal distribution with , where denote the number of in-degree of the node.

Activation Function

Batch normalization is a technique to ensure zero mean and unit variance in each hidden layer to avoid exploding or diminishing gradients during training. However, batch normalization has often been identified as a computational bottleneck in neural network training because computing the mean and the standard deviation of a layer is not only a dependent step but also a reduction step that cannot be efficiently parallelized. To tackle this problem, we will use the new activation function called "Scaled Exponential Linear Units" (SELUs)[18], a variant of the rectified activation function. Different from a standard batch normalization approach that adds an implicit layer for the named purpose after each hidden layer, SELUs utilizes the Banach fixed-point theorem to ensure convergence to zero mean and unit variance in each hidden layer without batch normalization.

Optimizer and Learning rate

We used an Adam optimizer with default settings [19] to update the weights by adaptive node-specific learning rates, whereas setting a global learning rate only functions to set an upper limit to the learning rates. This behavior allows Clairvoyante to use a higher learning rate for longer to speed up the training process.

Although the Adam optimizer performs learning rate decay intrinsically, we found decreasing the global learning rate when the cost of the model in training plateaued can lead to better model performance in our study. In Clairvoyante, we implemented two types of training modes. The fast training mode is an adaptive decay method that uses an initial learning rate at $1e^{-3}$, decreases the learning rate by a factor of 0.1 when the validation rate goes up and down for five rounds and stops after two times of decay. The nonstop training mode allows users to decide when to stop and continue using a lower learning rate.

Dropout and L2 Regularization

Although more than three million labeled truth variants are available for training, the scarcity of some labels, especially variants with a long indel length, could fail the model training by overfitting to abundantly labeled data. To alleviate the class imbalance, we apply both dropout[20] and L2 regularization[21] techniques in our study. Dropout is a powerful regularization technique. During training, dropout randomly ignoring nodes in a layer with probability p, then sums up the activations of remaining nodes and finally magnify the sum by $1/p$. Then during testing, sums up the activations of all nodes with no dropout. With probability p, the dropout technique is creating up to $1 \div (1-p)^n$ possible subnetworks during the training. Therefore, dropout can be seen as dividing a network into subnetworks with reused nodes during training. However, for a layer with just enough nodes, applying dropout will require more nodes to be added, thus potentially increasing the time needed to train a model. In balance, we applied dropout only to the first fully connected layer (FC4) with p=0.5, and L2 regularization to all the hidden layers in Clairvoyante. In practice, we set the lambda of L2 regularization the same as the learning rate.

Visualization

We created an interactive python notebook accessible within a web browser or a command line script for visualizing inputs and their corresponding node activations in hidden layers and output layers. **Supplementary Figure 1** shows the input and node activations in all hidden layers and output layers of an A>G SNP variant in sample HG002 test against a model trained with samples from HG001 for a thousand epochs at $1e^{-3}$ learning rate. Each of the nodes can be considered as a feature deduced through a chain of nonlinear transformations of the read alignments input.

Computational Performance

Making Clairvoyante a computationally efficient tool that can run on modern desktop and server computers with commodity configurations is one of our primary targets. Here we introduce the two critical methods used for decreasing computational time and memory consumption.

Clairvoyante can be roughly divided into two group of code, one is sample preparation (preprocessing and model training), and the second is sample evaluation (model evaluation and visualization). Model training runs efficiently because it invokes Tensorflow, which is maintained by a large developer community and has been intensively optimized with most of

its performance critical code written in C, C++ or CUDA. Using the native python interpreter, sample preprocessing became the bottleneck, and the performance did not improve by using multi-threading due to the existence of Global Interpreter Lock (GIL). We solved the problem by using Pypy[22], a Just-In-Time (JIT) compiler that performs as an alternative to the native python interpreter and requires no change to our code. In our study, Pypy sped up the sample preparation code by 5 to 10 times.

The memory consumption in model training was also a concern. For example, with a naïve encoding HG001 requires 40GB memory to store the variant and non-variant samples, which could prevent effective GPU utilization. We observed that these samples are immutable and follow the "write once, read many" access pattern. Thus, we applied in-memory compression using the blosc[23] library with the lz4hc compression algorithm, which provides high compression ratio, 100MB/s compression rate, and an ultra-fast decompression rate at 7GB/s. Our benchmarks show that applying in-memory compression makes no difference to the speed but decreased the memory consumption by five times.

Results

In this section, we first benchmarked Clairvoyante on Illumina, PacBio, and ONT data at known variant sites. Based on the benchmarking results, we have addressed several important questions regarding the results, the model training, and the input data. Last, we evaluated Clairvoyante's performance to call variants genome-wide.

Training Runtime Performance

We recommend using GPU acceleration for model training and CPU-only for variant calling. **Table 1** shows the performance of different GPU and CPU models in training. Using a high-performance desktop GPU model GTX 1080 Ti, 170 seconds are needed per epoch, which leads to about 5 hours to finish training a model with the fast training mode. However, for variant calling the speed up by GPU is insignificant because CPU workloads such as VCF file formatting and I/O operations dominate. Variant calling at 3.5M known variant sites takes about 20 minutes using 28 CPU cores. Variant calling genome-wide varies between 30 minutes to a few hours subject to which sequencing technology and alternative allele frequency cutoff were used (see **Results**).

Table 1. Time per epoch of different models of GPU and CPU in model training.

Equipment	Seconds per Epoch per 11M samples
GTX 1080 Ti	170
GTX 980	250
GTX Titan	520
Tesla K40 w/ top power setting	580
Tesla K40	620
Tesla K80 (one socket)	700
GTX 680	780
Intel Xeon E5-2680 v4 28-core	2900

Call Variants at Known Sites

Although Clairvoyante was designed targeting SMS, the method is generally applicable for short read data as well. We tested Clairvoyante on three sequencing technologies: Illumina, PacBio, and ONT. We tested both the fast and the nonstop training mode. In nonstop training mode, we firstly trained a model from 0 to 999-epoch at learning rate $1e^{-3}$, then to 1499-

epoch at $1e^{-4}$, and finally to 1999-epoch at $1e^{-5}$. We then benchmarked the model generated by the fast mode, and all three models stopped at different learning rates in the nonstop mode. We also benchmarked variant calling on one sample (e.g. HG001) using a model trained on another sample (e.g. HG002). Further, we ran GATK UnifiedGenotyper[6] and GATK HaplotypeCaller[6] for comparison. We also tried running three other tools including PacBio GenomicConsensus v5.1[24], the paftools in minimap2 r763[25] and Nanopolish v0.9.0[26], but we only succeeded in Nanopolish. The reason why the other tools failed, and the commands used for generating the results in this section are presented in **Supplementary Material, Call Variants at Known Sites, Commands**.

We used the submodule *vcfeval* in RTG Tools[27] version 3.7 to benchmark our results and generate three metrics including Precision, Recall, and F1-score. From the number of true positives (*TP*), false positives (*FP*), and false negatives (*FN*), we compute the three metrics as Precision = $TP \div (TP + FP)$, Recall = $TP \div (TP + FN)$, and F1-score = $2TP/(2TP + FN + FP)$. *FP* are defined as variants existing in the GIAB dataset that also identified as a variant by Clairvoyante, but with discrepant variant type, alternative allele or zygosity. *FN* are defined as the variants existing in the GIAB dataset but identified as a non-variant by Clairvoyante. F1-score is the harmonic mean of the precision and recall. RTG *vcfeval* also provides the best variant quality cutoff for each dataset, filtering the variants under which can maximize the F1-score. To the best of our knowledge, RTG *vcfeval* was also used by the GIAB project itself. *Vcfeval* cannot deal with Indel variant calls without an exact allele. However, in our study, Clairvoyante was set to provide the exact allele only for Indels ≤4bp. Thus, all Indels >4bp were removed from both the baseline and the variant calls before benchmarking. The commands used for benchmarking are presented in **Supplementary Material, Benchmarking, Commands**.

Table 2 shows the performance of Clairvoyante on Illumina data. The best accuracy is achieved by calling variants in HG001 using the model trained on HG001 at 1499-epoch, with 99.73% precision, 99.62% recall and 99.68% F1-score. A major concern of using deep learning or any statistical learning technique for variant calling is the potential for overfitting to the training samples. Our results show Clairvoyante is not affected by overfitting, and we validated the versatility of the trained models by calling variants in a genome using a model trained on a second, independent sample. Interestingly, the performance of calling variants in HG002 using a model trained on HG001 (for convenience, hereafter denoted as HG002>HG001) is 0.25% higher (99.52% against 99.27%) than HG002>HG002 and similar to HG001>HG001. As we know the truth variants in HG001 were verified and rectified by more orthogonal genotyping methods than HG002[11], we believe it is the higher quality of truth variants in HG001 than HG002 that gave the model trained on HG001 a higher performance. Clairvoyante achieved 0.14% higher (99.68% against 99.54%) F1-score than GATK HaplotypeCaller on HG001 but almost the same (99.52% against 99.53%) on HG002. This again corroborated the importance of high-quality truth variants for Clairvoyante to achieve superior performance.

Table 3 shows the performance of Clairvoyante on PacBio data. The best performance is achieved by calling variants in HG001 using the model trained on HG001 at 1999-epoch, with 97.65% precision, 96.53% recall and 97.09% F1-score. DeepVariant[14] was benchmarked the same dataset in their studied and reported 97.25% precision, 88.51% recall and 92.67% F1-score. We noticed our benchmark differs from DeepVariant because we have removed Indels >4bp from both the baseline and variant calls. If we simply assume DeepVariant can

identify all the 91k Indels >4bp correctly, the recall will increase to 90.73%, which is still 5.8% lower than Clairvoyante.

Table 4 shows the performance of Clairvoyante on ONT data. As there are no available deep coverage ONT data sets for HG002, we only provided the benchmarks of variant calls in HG001 using models also trained on HG001. The best precision is 95.36%, achieved at 1499-epoch. The best recall is 88.70%, and the best F1-score is 91.83%, both achieved at 1999-epoch. We also benchmarked Nanopolish[26] using the same dataset, but due to its high computational resource requirement, we only called variants in chr19, which finished in about eleven hours with the peak memory at 125GB. Nanopolish achieved 97.09%, 80.56% and 88.06% on precision, recall, and F1-score, respectively.

Table 2. Performance of Clairvoyante on Illumina data at known variant sites. *: fast training mode.

Sequencing Technology	Train using Variant in	Trained Epochs	Ending Learning Rate and Lambda	Call Variants in	Best Variant Quality Cutoff	Precision	Recall	F1 Score
Illumina	HG001	67	1.E-05	HG001	54	99.68%	99.50%	99.59%
		999	1.E-03		72	99.71%	99.58%	99.65%
		1499	1.E-04		93	99.73%	99.62%	99.68%
		1999	1.E-05		91	99.73%	99.62%	99.68%
	HG001	67	1.E-05	HG002	54	99.63%	99.38%	99.50%
		999	1.E-03		82	99.64%	99.41%	99.52%
		1499	1.E-04		118	99.60%	99.38%	99.49%
		1999	1.E-05		129	99.58%	99.37%	99.47%
	HG002	66	1.E-05	HG001	60	99.26%	98.98%	99.12%
		999	1.E-03		83	99.26%	99.04%	99.15%
		1499	1.E-04		121	99.21%	99.00%	99.11%
		1999	1.E-05		141	99.20%	98.98%	99.09%
	HG002	66	1.E-05	HG002	51	99.29%	99.07%	99.18%
		999	1.E-03		76	99.32%	99.15%	99.24%
		1499	1.E-04		75	99.33%	99.21%	99.27%
		1999	1.E-05		85	99.33%	99.21%	99.27%
GATK UnifiedGenotyper, HG001						99.84%	87.36%	93.18%
GATK HaplotypeCaller, HG001						99.88%	99.19%	99.54%
GATK UnifiedGenotyper, HG002						99.82%	87.80%	93.42%
GATK HaplotypeCaller, HG002						99.88%	99.19%	99.53%

Table 3. Performance of Clairvoyante on PacBio data at known variant sites. *: fast training mode.

Sequencing Technology	Train using Variant in	Trained Epochs	Ending Learning Rate and Lambda	Call Variants in	Best Variant Quality Cutoff	Precision	Recall	F1 Score
PacBio	HG001	50	1.E-05	HG001	45	96.91%	94.35%	95.62%
		999	1.E-03		52	97.46%	95.42%	96.43%
		1499	1.E-04		55	97.68%	96.38%	97.03%
		1999	1.E-05		52	97.65%	96.53%	97.09%
	HG001	50	1.E-05	HG002	48	96.65%	94.16%	95.39%
		999	1.E-03		58	96.94%	94.43%	95.67%
		1499	1.E-04		63	96.66%	94.35%	95.49%
		1999	1.E-05		60	96.54%	94.37%	95.44%

	HG002	72	1.E-05	HG001	38	96.97%	93.11%	95.00%
		999	1.E-03		68	97.50%	92.72%	95.05%
		1499	1.E-04		75	96.94%	92.98%	94.92%
		1999	1.E-05		75	96.68%	92.85%	94.73%
	HG002	72	1.E-05	HG002	34	96.83%	94.46%	95.63%
		999	1.E-03		36	96.83%	95.51%	96.17%
		1499	1.E-04		68	98.13%	95.73%	96.91%
		1999	1.E-05		51	97.65%	96.33%	96.99%
GATK UnifiedGenotyper, HG001						68.74%	24.23%	35.83%
GATK HaplotypeCaller, HG001						64.73%	1.98%	3.85%
GATK UnifiedGenotyper, HG002						69.28%	24.19%	35.86%
GATK HaplotypeCaller, HG002						66.84%	1.21%	2.37%

Table 4. Performance of Clairvoyante on ONT data at known variant sites. *: fast training mode.

Sequencing Technology	Train using Variant in	Trained Epochs	Ending Learning Rate and Lambda	Call Variants in	Best Variant Quality Cutoff	Precision	Recall	F1 Score
Oxford Nanopore	HG001	110*	1.E-05	HG001	33	94.90%	84.35%	89.34%
		999	1.E-03		33	94.07%	85.87%	89.79%
		1499	1.E-04		37	95.36%	88.12%	91.59%
		1999	1.E-05		37	95.20%	88.70%	91.83%
GATK UnifiedGenotyper, HG001						82.65%	15.70%	26.38%
GATK HaplotypeCaller, HG001						75.23%	1.28%	2.52%

What are those false positives and false negatives?

While we have settled on a highly optimized version of Clairvoyante for the experiments in this paper, it is still interesting to study the remaining FP and FN variant calls and how they are distributed. To achieve this, we have randomly picked 100 FP and 100 FN from the variants called in HG002 using the model trained on HG001 using the fast training mode (stopped at 67-epoch), generated plots on their input and output and manually inspected each one. A summary of the results are shown in **Figure 3**. The most problematic category of FP and FN variants, accounting for 71FP and 42 FN, are variants with two or more alternative alleles at the same position. This type of variant is not currently supported by Clairvoyante and instead only one allele will be reported (this limitation is further discussed in the Discussion section). Another 17 FP and 47 FN failed either because of “difficult reference” (low complexity sequence, tandem repeat or homopolymer run) or “lack of evidence” (low depth or even zero coverage). Among the 100 FP and 100 FN variants, we successfully determined 12.06% (12 FP and 11 FN) are truly wrongly classified by Clairvoyante, 20.50% (17 FP and 24 FN) suggest either an alignment artifact or were wrongly missed by GIAB dataset, and the remainders were undecidable even manually or with multiple alternative alleles. More details for each FP and FN are shown in **Supplementary Tables 1 and 2** and the plots are available online (**Supplementary Material, Call Variants at Known Sites, Resources, FP/FN plots**).

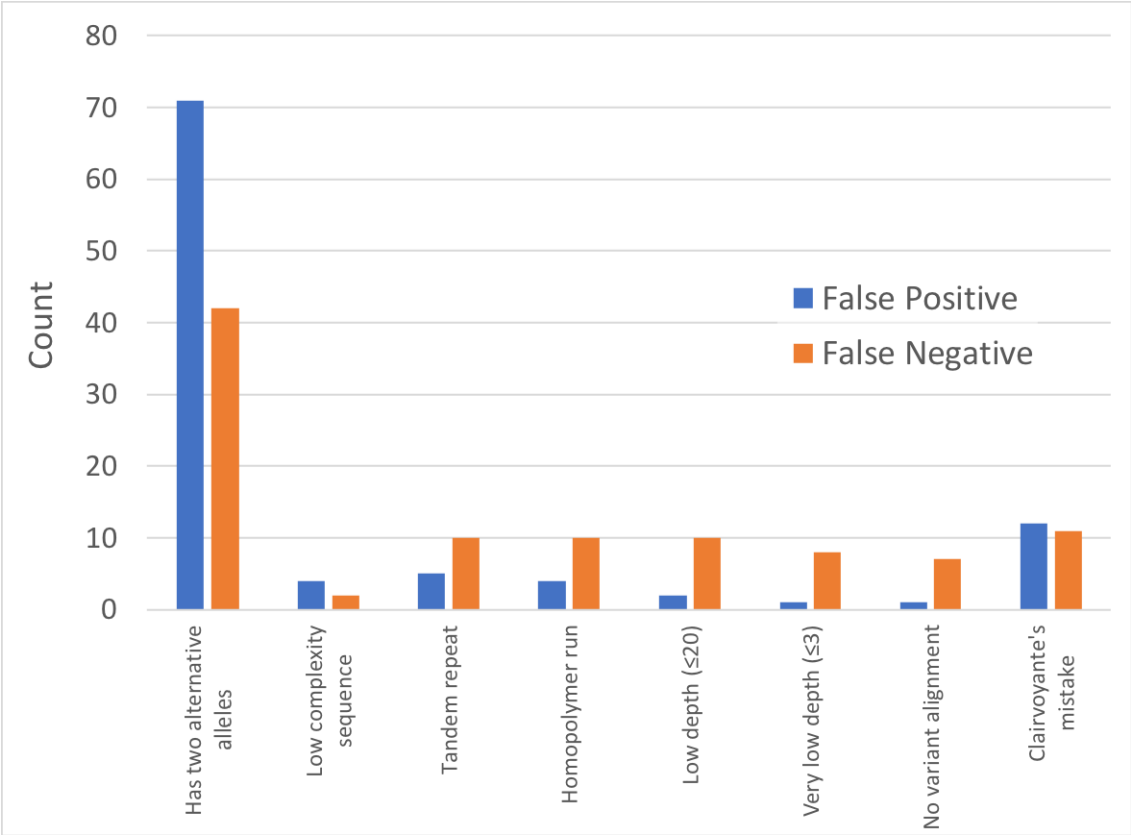

Figure 3. Summary of the reason for failure on 100 randomly picked false positive variants, and 100 randomly picked false negative variants.

Can lower learning rate and longer training provide better performance?

The benchmarking results on the three models stopping at different learning rates allow us to study whether lower learning rate can provide better results and derive how much training is enough. For ONT, both from 999-epoch to 1499-epoch and from 1499-epoch to 1999-epoch, significant improvements were observed. However, in PacBio (**Table 3**), from 1499-epoch to 1999-epoch, the F1-Score increased (97.03% to 97.09%, 96.91% to 96.99%) when both variant calling and model training are using the same sample, but decreased (95.49% to 95.44%, 94.92% to 94.73%) when using different samples. The results suggest that Clairvoyante was overfitting the training data with a too low learning rate. The same behavior is also observed with Illumina data (**Table 2**). Thus, we suggest the Clairvoyante users to 1) stop at a higher learning rate for less noisy data; 2) train multiple samples stopping at different learning rates and select the best through performance evaluation; or 3) use a model trained on truth variants from multiple samples.

Can a model train on truth variants from multiple samples provide better performance?

Intuitively, a model trained on truth variants from two or more samples should perform better than those trained on just a single sample, provide that the truth variants from different samples have similiarly high quality. The model might even be more versatile if the characteristics of input, such as average depth, differ between samples. To verify our hypothesis, we benchmarked the variants called in HG003 (**Supplementary Material, Data Source, PacBio Data**) on three different models trained on 1) HG001; 2) HG002, and; 3)

HG001+HG002. All three models were trained for 1000 epochs at learning rate $1e^{-3}$, then another 500 epochs at learning rate $1e^{-4}$. Noteworthy, the time used for training the HG001+HG002 model doubled, with doubling the number of true variants and paired non-variants. If our hypothesis is correct, the variant calling performance should increase for HG003 when using the HG001+HG002 model than the HG001 model or the HG002 model. The results are shown in **Table 5**. Using the HG001+HG002 model, the F1-score is 0.55% higher than using HG001 only and 2.88% higher than using the HG002 only. We conclude that using multiple samples for model training can increase the performance of Clairvoyante, although we expect marginal improvement gains when using more than a few samples.

Table 5. Performance of variant calls in HG003 on three different models including HG001 only, HG002 only and HG001+HG002.

Train using Variant in	Best Variant Quality Cutoff	Precision	Recall	F1 Score
HG001	72	95.61%	90.51%	92.99%
HG002	71	93.38%	88.09%	90.66%
HG001 + HG002	56	95.91%	91.29%	93.54%

Can a higher input data quality improve the variant calling performance?

In Table 4, we used the 'rel3' ONT dataset generated by the Nanopore WGS consortium. Very recently, the consortium released an augmented dataset labeled 'rel5' (see Supplementary Material, Data Source, Oxford Nanopore Data). The 'rel5' data are a merger of NA12878 DNA sequencing data from 'rel3' (regular sequencing protocols, about 30x) and 'rel4' (ultra-read set, 7.7x extra), recalled with the latest base-caller. Thus, we expect to see improved performance, given that the performance of Clairvoyante on ONT was limited by the input data quality. We trained a model on 'rel5' for 999 epochs at learning rate $1e^{-3}$. Compare to 'rel3', the precision improved from 94.07% to 97.21%, the recall improved from 85.87% to 88.80%, and the F1-Score improved 89.79% to 92.81%. Thus, the result reflects our intuition that Clairvoyante's performance on ONT data is limited by the input data quality and thus will improve over time as the technology and basecalling mature.

Network topology and capacity evaluation

In the previous subsection, we have shown Clairvoyante's capacity to perform better on noisy PacBio and ONT data when trained with more data of higher quality. We next evaluated the performance by considering a "slim version" of Clairvoyante with smaller capacity that could potentially improve computational requirements. With the slim version, we expect to see greater performance in higher quality Illumina data than noisy data like ONT and PacBio data as the classification problem is easier with less noisy data. The slim version includes 165k parameters, which is about ten times lower than the original version. Instead of isometrically scaling down the original network, we evaluated several different designs resulting in some network components with significantly reduced runtime than others or even reducing the parameters by 10 times while still achieving the best runtime and F1-Score possible.

Our final slim network design removes the pooling between convolutional layers, slightly enlarged the kernel size in convolution and reduced the number of nodes in the two fully-connected layers by ten times. We trained models using the fast training mode on HG001 and benchmarked the Illumina, PacBio and ONT data on both HG001 and HG002. The results are

shown in **Table 6**. As expected, the F1-scores degraded least in the Illumina datasets (0.82% and 0.73%), while degraded most in the ONT dataset (2.23%), with PacBio in the middle (1.68% and 1.90%). The slim version is available as a part of the Clairvoyante toolset and can be enabled with option '--slim.'

Table 6. F1-scores of different datasets on different network designs. Both the original models and slim models were trained on HG001 using the fast training mode.

	Illumina		PacBio		ONT
	HG001	HG002	HG001	HG002	HG001
Original	99.59%	99.50%	95.62%	95.39%	89.34%
Slim	98.77%	98.77%	93.94%	93.49%	87.11%
Degraded	0.82%	0.73%	1.68%	1.90%	2.23%

Genome-wide Variant Identification

Beyond benchmarking variants at known sites, in this section we benchmarked Clairvoyante's performance on calling variants genome-wide. Calling variants genome-wide is challenging because it tests not only how good Clairvoyante can derive the correct variant type, zygosity and alternative allele of a variant when evidence is marginal, but also in reverse, how good Clairvoyante can deny a non-variant when evidence is also marginal. Instead of naively evaluating all three billion sites of the whole genome with Clairvoyante, we tested the performance at different alternative allele cutoffs for all three sequencing technologies. As expected, a higher allele cutoff speeds up variant calling by producing fewer candidates to be tested by Clairvoyante, but worsens recall especially for noisy data like PacBio and ONT. Our experiments provide a reference point on how to choose a cutoff for each sequencing technology to achieve a good balance between recall and running speed. All models were trained for 1000 epochs with learning rate at $1e^{-3}$. All the experiments were performed on two Intel Xeon E5-2680 v4 using all the 28 cores. The commands used for generating the results in this section are presented in **Supplementary Material, Call Variants Genome-wide, Commands**.

The results are shown in **Table 7**. As expected, with higher alternative allele frequency cutout (20%), in all experiments the precision was higher while the recall and time consumption were lower. For Illumina data, the best F1-score (with 0.2 allele frequency) for Clairvoyante was 98.65% for HG001 and 98.61% for HG002. The run time varied between half and an hour, being 40 minutes for the best F1-score. Using the same configuration, GATK HaplotypeCaller achieved F1-score 98.82% for HG001 and 98.86% for HG002; both ran for about 8 hours. Clairvoyante achieved a higher recall but a lower precision than GATK HaplotypeCaller. Inspecting the false positive and false negative variant calls for Clairvoyante, we found about 0.19% in FP, and 0.15% in FN were because of scenarios of two alternative alleles. Future works include extending Clairvoyante to support variants with multiple alternative alleles could increase the F1-score to about 98.82% if all variants with two alternative alleles can be correctly identified. As reported, DeepVariant did the same experiment on a 60x HG001 dataset (10x deeper than our experiment) and achieved 95.21% F1-score[14], which is 3.44% lower than Clairvoyante.

For the PacBio data, the best F1-scores were also achieved at 0.2 allele frequency cutoff. The best F1-score is 92.57% for HG001 and 93.05% for HG002. In contrast, DeepVariant has achieved 35.79% F1-score (22.14% precision, 93.36% recall) on HG001 as reported in their paper[14]. The run time for Clairvoyante at 0.25 frequency cutoff is about 2 hours, which is about half the time consumption at 0.2 frequency cutoff, and about 1/5 the time consumption

at 0.1 frequency cutoff. For ONT data, the best F1-score 77.89% was achieved at 0.1 frequency cutoff. However, the F1-score at 0.25 frequency cutoff is just slightly lower (76.95%), but ran about 5 times faster, thus we suggest using 0.25 as the frequency cutoff. The run time is on average about 1.5 times longer than PacBio, suggesting a higher level of noise in data. As discussed in the previous section, we expect the performance over Oxford Nanopore data will further improve in the near future with improved base calling.

Table 7. Performance of using Clairvoyante for variant calling genome-wide on Illumina, PacBio and ONT datasets. All models were trained for 1000 epochs with learning rate at $1e^{-3}$.

Sequencing Technology	Train using Variants in	Call Variants in	Alt. Allele Freq. Cutoff	Best Variant Quality Cutoff	Precision	Recall	F1 Score	Time Consumption
Illumina	HG001	HG001	0.1	189	98.16%	98.93%	98.55%	1:08
			0.2	182	98.41%	98.88%	98.65%	0:43
			0.25	180	98.71%	97.95%	98.33%	0:26
		HG002	0.1	192	98.13%	98.77%	98.45%	1:11
			0.2	183	98.35%	98.77%	98.56%	0:41
			0.25	182	98.67%	97.88%	98.27%	0:30
	HG002	HG001	0.1	198	98.59%	98.50%	98.54%	1:16
			0.2	192	98.75%	98.39%	98.57%	0:47
			0.25	184	98.94%	97.60%	98.27%	0:25
		HG002	0.1	195	98.53%	98.59%	98.56%	1:07
			0.2	188	98.71%	98.50%	98.61%	0:44
			0.25	182	98.95%	97.73%	98.33%	0:25
	GATK UnifiedGenotyper, HG001			57	99.23%	84.82%	91.46%	0:46
	GATK HaplotypeCaller, HG001			2	99.10%	98.54%	98.82%	8:45
	GATK UnifiedGenotyper, HG002			54	99.07%	85.35%	91.70%	0:46
	GATK HaplotypeCaller, HG002			2	99.05%	98.68%	98.86%	8:23
PacBio	HG001	HG001	0.1	157	96.31%	88.63%	92.31%	9:46
			0.2	130	98.12%	87.62%	92.57%	3:53
			0.25	125	98.62%	83.11%	90.20%	2:01
		HG002	0.1	153	97.00%	89.08%	92.87%	9:24
			0.2	132	97.93%	88.30%	92.86%	3:34
			0.25	116	98.06%	84.69%	90.89%	1:46
	HG002	HG001	0.1	163	95.58%	86.69%	90.92%	14:55
			0.2	147	97.49%	85.64%	91.18%	3:29
			0.25	139	98.16%	81.47%	89.04%	1:39
		HG002	0.1	150	97.10%	89.31%	93.04%	15:31
			0.2	134	98.09%	88.51%	93.05%	3:34
			0.25	118	98.20%	84.76%	90.98%	1:46
Oxford Nanopore	HG001	HG001	0.1	140	86.24%	71.01%	77.89%	13:01
			0.2	139	87.24%	70.21%	77.80%	4:47
			0.25	136	87.76%	68.51%	76.95%	2:40
			0.35	130	90.96%	57.43%	70.41%	1:30

Novel variants unraveled by PacBio and ONT

Although the truth SNPs and Indels provided by GIAB were intensively called and meticulously curated, they are not yet complete, and they were generated without any SMS technology[11]. Bearing in mind that some false positive variant calls in the previous experiments might actually be real, we tried to identify a set of variants that are only approachable with the PacBio and ONT data, but not yet indexed in GIAB. For the HG001 sample (variants called in HG001 using a model trained on HG001), we extracted the "false positive" variants (identified genome-wide with a 0.2 alternative allele frequency cutoff)

called in both the PacBio and ONT dataset. Then we calculated the geometric mean of the variant qualities of the two datasets, and we filtered the variants with a mean quality lower than 135 (calculated as the geometric mean of the two best variant quality cutoffs, 130 and 139). Finally, we removed those "false positive" variants within the flanking one base of the known variants in GIAB. The resulting catalog of 3,135 variants retained are listed in **Supplementary Table 3**. 2,732 are SNPs, 298 are deletions, and 105 are insertions. Among the SNPs, 1,602 are transitions, and 1,130 are transversions. The Ti/Tv ratio is ~1.42, which is significantly higher than random (0.5). We manually inspected the top ten variants in quality using IGV[28] to verify their authenticity (**Figure 4a** and **Supplementary Figure 2a-2i**). All ten variants were visually reviewed, and are strongly supported by the data.

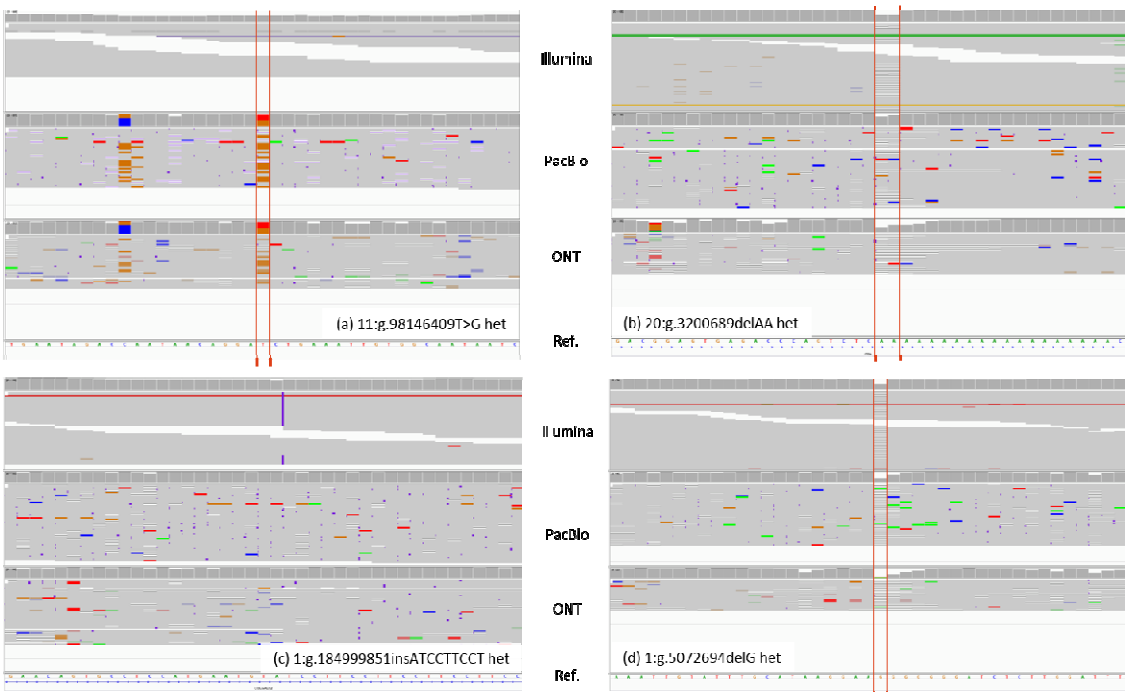

Figure 4. The IGV screen capture of (**a**) a heterozygote SNP from T to G at chromosome 11, position 98,146,409 only detectable by PacBio and ONT, (**b**) a heterozygote deletion AA at chromosome 20, position 3,200,689 undetectable by all three technologies, (**c**) a heterozygote insertion ATCCTTCCT at chromosome 1, position 184,999,851 only detectable by Illumina, and; (**d**) a heterozygote deletion G at chromosome 1, position 5,072,694 detectable by all three technologies. The tracks from top to down show the alignments of the Illumina, PacBio and ONT reads from HG001 aligned to the human reference GRCh37.

We also analyzed why some variants cannot be detected by the PacBio and ONT technologies. **Figure 5** shows the number of known variants undetected by different combinations of sequencing technologies. We inspected the genome sequence immediately after the variants and found among the 12,331 variants undetected by all three sequencing technologies, 3,289 (26.67%) are located in homopolymer runs, and 3,632 (29.45%) are located in short tandem repeats. Among the 178,331 variants that cannot be detected by PacBio and ONT, 102,840 (57.67%) are located in homopolymer runs, and 33,058 (18.54%) are located in short tandem repeats. For Illustration, **Figure 4b** to **d** depicted b) a known variant in homopolymer runs undetected by all three sequencing technologies, c) a known variant in short tandem repeats that cannot be detected PacBio and ONT, and d) a known variant flanked by random sequenced detected by all three sequencing technologies. It is a

known problem that single molecule sequencing technologies including PacBio and ONT has significantly increase error rate at homopolymer runs and short tandem repeats[29]. Future improvements to the base-calling algorithm or sequencing chemistry will lead to raw reads with higher accuracy at these troublesome genome regions and hence, further decrease the number of known variants undetected by Clairvoyante.

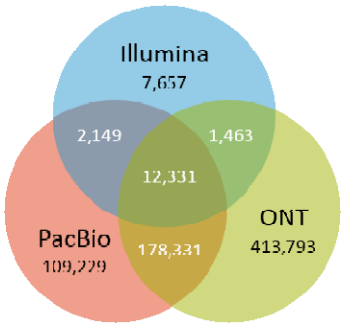

Figure 5. A Venn diagram that shows the number of undetected known variants by different sequencing technologies or combinations.

Discussion

In this paper, we presented Clairvoyante, a multi-task convolutional deep neural network for variant calling using single molecule sequencing. By first benchmarking on the Illumina data, we show Clairvoyante performance was on-par to the gold-standard GATK HaplotypeCaller. We analyzed the false positive and false negative variant calls in depth and found complex variants with multiple alternative alleles to be the dominate source of error. We further evaluated several different aspects of Clairvoyante to assess the quality of the design and how can we further improve its performance by training longer with lower learning rate, combining multiple samples for training, or improving the input data quality. Our experiments on using Clairvoyante to call variants genome-wide suggested a range to search for the best alternative allele cutoff to balance the run time and recall for each sequencing technology. To the best of our knowledge, Clairvoyante is the first method for SMS to finish a whole genome variant calling within two hours on a single CPU-only server, while providing better precision and recall than other state-of-the-art variant callers such as Nanopolish. A deeper look into the “false positive” variant calls has identified 3,135 variants in HG001 that are not yet indexed in GIAB and are only approachable by Single Molecule Sequencing technologies like PacBio and ONT.

Clairvoyante relies on high-quality training samples to provide accurate and unbiased variant calling. This hinders Clairvoyante from being applied to completely novel sequencing technologies and chemistries, for which high-quality sequencing dataset on standard GIAB samples has yet been produced. Nevertheless, with the increasing agreement for NA12878 as a gold-standard reference this requirement seems to be quite manageable. The current design of Clairvoyante ignore variants with two or more alternative alleles. Although the number of variants with two or more alternative alleles is small, a few thousands of the 3.5M total sites, the design will be improved in the future to tackle this small but important group of variants. Due to the rareness of long indel variants for model training, Clairvoyante was set to provide the exact alternative allele only for indel variants ≤4bp. The limitation can be lifted with more and more high-quality training samples available. The current Clairvoyante implementation also does not consider the base quality of the sequencing reads as Clairvoyante was targeting SMS, which do not have meaningful base quality values to

improve the quality of variant calling. Nevertheless, Clairvoyante can be extended to consider base quality by imposing it as a weight on depth or add it as an additional tensor to the input. We do not suggest removing any alignment by their mapping quality because low quality mappings will be learned by the Clairvoyante model to be unreliable. This provides valuable information about the trustworthiness of certain genomic regions. In future work, we plan to extend Clairvoyante to support somatic variant calling and trio-sample based variant calling. Based on GIAB's high confidence region lists for variant calling, we also plan on making PacBio-specific, and ONT-specific high confidence region lists by further investigating the false positive and false negative variant calls made by Clairvoyante on the two technologies.

Acknowledgments

We thank Guangyu Yang for adding code to Clairvoyante to enable visualization using TensorBoard. We thank Chi-Man Liu and Yifan Zhang for benchmarking Nanopolish. R. L. and T. L. were partially supported by Innovative and Technology Fund ITS/331/17FP from the Innovation and Technology Commission, HKSAR. This work was also supported, in part, by awards from the National Science Foundation (DBI-1350041) and the National Institutes of Health (R01-HG006677 and UM1-HG008898).

Author Contributions

R.L. and M.S. conceived the study. All authors analyzed the data and wrote the manuscript.

References

1 Goodwin, S., McPherson, J. D. & McCombie, W. R. Coming of age: ten years of next-generation sequencing technologies. *Nat Rev Genet* **17**, 333-351, doi:10.1038/nrg.2016.49 (2016).

2 Nakamura, K. *et al.* Sequence-specific error profile of Illumina sequencers. *Nucleic Acids Res* **39**, e90, doi:10.1093/nar/gkr344 (2011).

3 Hatem, A., Bozdag, D., Toland, A. E. & Catalyurek, U. V. Benchmarking short sequence mapping tools. *BMC Bioinformatics* **14**, 184, doi:10.1186/1471-2105-14-184 (2013).

4 Li, H. Toward better understanding of artifacts in variant calling from high-coverage samples. *Bioinformatics* **30**, 2843-2851, doi:10.1093/bioinformatics/btu356 (2014).

5 Luo, R., Schatz, M. C. & Salzberg, S. L. 16GT: a fast and sensitive variant caller using a 16-genotype probabilistic model. *GigaScience* (2017).

6 Van der Auwera, G. A. *et al.* From FastQ data to high confidence variant calls: the Genome Analysis Toolkit best practices pipeline. *Curr Protoc Bioinformatics* **43**, 11 10 11-33, doi:10.1002/0471250953.bi1110s43 (2013).

7 Sedlazeck, F. J., Lee, H., Darby, C. A. & Schatz, M. C. Piercing the dark matter: bioinformatics of long-range sequencing and mapping. *Nat Rev Genet*, doi:10.1038/s41576-018-0003-4 (2018).

8 LeCun, Y. The MNIST database of handwritten digits. *http://yann.lecun.com/exdb/mnist/* (1999).

9 Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. & Wojna, Z. in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2818-2826.

10 Zook, J. M. *et al.* Integrating human sequence data sets provides a resource of benchmark SNP and indel genotype calls. *Nat Biotechnol* **32**, 246-251, doi:10.1038/nbt.2835 (2014).
11 Zook, J. M. *et al.* Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Sci Data* **3**, 160025, doi:10.1038/sdata.2016.25 (2016).
12 Jain, M. *et al.* Nanopore sequencing and assembly of a human genome with ultra-long reads. *Nat Biotechnol* **36**, 338-345, doi:10.1038/nbt.4060 (2018).
13 Sedlazeck, F. J. *et al.* Accurate detection of complex structural variations using single molecule sequencing. *bioRxiv*, doi:10.1101/169557 (2017).
14 Poplin, R. *et al.* Creating a universal SNP and small indel variant caller with deep neural networks. *bioRxiv*, 092890 (2016).
15 Chin, J. *Simple Convolutional Neural Network for Genomic Variant Calling with TensorFlow*, <https://towardsdatascience.com/simple-convolution-neural-network-for-genomic-variant-calling-with-tensorflow-c085dbc2026f> (2017).
16 Abadi, M. *et al.* Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* (2016).
17 He, K., Zhang, X., Ren, S. & Sun, J. in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)* 1026-1034 (IEEE Computer Society, 2015).
18 Klambauer, G., Unterthiner, T., Mayr, A. & Hochreiter, S. Self-Normalizing Neural Networks. *arXiv preprint arXiv:1706.02515* (2017).
19 Kingma, D. & Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
20 Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580* (2012).
21 Cortes, C., Mohri, M. & Rostamizadeh, A. in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence.* 109-116 (AUAI Press).
22 Rigo, A. *et al. Pypy*, <https://pypy.org/> (2018).
23 Alted, F. *Blosc: A blocking, shuffling and lossless compression library*, <http://blosc.org/> (2018).
24 Biosciences, P. *Genomic Consensus*, <https://github.com/PacificBiosciences/GenomicConsensus> (2018).
25 Li, H. Minimap2: versatile pairwise alignment for nucleotide sequences. *arXiv* **1708** (2017).
26 Loman, N. J., Quick, J. & Simpson, J. T. A complete bacterial genome assembled de novo using only nanopore sequencing data. *Nature methods* **12**, 733 (2015).
27 Cleary, J. G. *et al.* Joint variant and de novo mutation identification on pedigrees from high-throughput sequencing data. *J Comput Biol* **21**, 405-419, doi:10.1089/cmb.2014.0029 (2014).
28 Robinson, J. T., Thorvaldsdottir, H., Wenger, A. M., Zehir, A. & Mesirov, J. P. Variant Review with the Integrative Genomics Viewer. *Cancer Res* **77**, e31-e34, doi:10.1158/0008-5472.CAN-17-0337 (2017).
29 Lu, H., Giordano, F. & Ning, Z. Oxford Nanopore MinION sequencing and genome assembly. *Genomics, proteomics & bioinformatics* **14**, 265-279 (2016).