SACall: A Neural Network Basecaller for Oxford Nanopore Sequencing Data Based on Self-Attention Mechanism

Neng Huang, Fan Nie, Peng Ni, Feng Luo, and Jianxin Wang

Abstract—Highly portable Oxford Nanopore sequencer producing long reads in real-time at low cost has made many breakthroughs in genomics studies. However, a major limitation of nanopore sequencing is its high errors when deciphering DNA sequences from noisy and complex raw data. In this paper, we developed an end-to-end basecaller, SACall, based on convolution layers, transformer self-attention layers and a CTC decoder. In SACall, the convolution layers are used to downsample the signals and capture the local patterns. To achieve the contextual relevance of signals, self-attention layers are adopted to calculate the similarity of the signals at any two positions in the raw signal sequence. Finally, the CTC decoder generates the DNA sequence by a beam search algorithm. We use a benchmark consisting of nine isolated genomes to test the quality of different basecallers including SACall, Albacore, and Guppy. The performances of basecallers are evaluated from the perspective of read accuracy, assembly quality, and consensus accuracy. Among most of the genomes in the test benchmark, the reads basecalled by SACall have fewer errors than the reads basecalled by other basecallers. When assembling the basecalled reads of each genome, the assembly from SACall basecalled reads achieves a higher assembly identity. In addition, there are fewer errors in the polished assembly from reads basecalled by SACall compared to those basecalled by Albacore and Guppy. In general, SACall outperforms the Nanopore official basecallers Albacore and Guppy in the benchmark. Moreover, SACall is an open-source and freely available basecaller, which gives a chance for researchers to train their own basecalling models on specific data and basecall Nanopore reads.

Index Terms—ONT nanopore sequencing, artificial neural network, self-attention, higher accuracy

1 Introduction

Single molecule sequencing long reads technology (SMS) [1] such as Pacific Bioscience [2], [3] and Oxford Nanopore [4], [5] has made new impetus to genomics [6], [7], [8], transcriptomics [9] and epigenomics [10]. The MinION sequencer from Oxford Nanopore Technologies (ONT) is the only portable real-time DNA and RNA sequencing instrument with the size of a smartphone, and each flowcell can generate around 30 Gbp of DNA sequencing data. Low cost, high speed, versatility and portability make MinION a promising tool for rapid nucleic acid sequencing. Unlike fluorescent labeling sequencing technologies, Nanopore long-read sequencing is based on the following innovation: a protein nanopore is embedded in an electrically resistant polymer membrane. By setting a voltage across the membrane, an ionic current is passed through the nanopore. This event can create a characteristic disruption in the current as the analyte passes through or near the nanopore. By measuring that current, it is possible to identify the molecule. Based on this design, nanopore sequencing technology has many advantages such as long reads, real-time analysis, no PCR amplification, no GC bias and direct measurement of epigenetics [11].

High throughput sequencers from Oxford Nanopore Technologies are reported to generate sequencing reads longer than 800 kp [12] and even exceeding 2 MB [13], which overcomes the major limitation of short-read sequencing whose read length is limited to several hundred bases. These long reads provide great potentials for de novo genome assembly. When the long reads span repetitive elements, repeat copies can be anchored within an unique position of the genome. However, in contrast to next-generation sequencing short reads with accuracy greater than 99 percent, the main limitation of nanopore sequencing is its high error rate. Even with improvements in nanopore chemistry and basecalling tools, read error rates are still between 5 and 15 percent [9]. We can reasonably assume that the errors in nanopore sequencing are caused by two reasons. The first reason is that the inherent sequencing limitation of nanopore technology leads to low signal-to-noise ratio of the original raw data, which makes it hard to determine the true nucleotide sequence. There are some factors which may contribute to low signal-to-noise ratio: (i) structure similarity of the nucleotides; (ii) the signal remains unchanged within homopolymers [14]; (iii) five nucleotides simultaneously affect the signal (at each moment five nucleotides stay in the nanopore, which is $4^5 = 1024$ combinations); (iv) the nonuniform speed of DNA passing through the pore [15], [16]. Second, errors occur in the process of translating

- *Neng Huang, Fan Nie, Peng Ni, and Jianxin Wang are with the School of Computer Science and Engineering, Central South University, Changsha 410083, China and Hunan Provincial Key Lab on Bioinformatics, Central South University, Changsha 410083, China. E-mail: {huangneng, niefan, nipeng, jxwang}@mail.csu.edu.cn.*
- *Feng Luo is with the School of Computing, Clemson University, Clemson, SC 29634 USA. E-mail: luofeng@clemson.edu.*

Manuscript received 30 Mar. 2020; revised 17 Aug. 2020; accepted 6 Nov. 2020. Date of publication 19 Nov. 2020; date of current version 3 Feb. 2022. (Corresponding author: Jianxin Wang.)
Digital Object Identifier no. 10.1109/TCBB.2020.3039244

the electrical current signals into a nucleotide sequence. The information of DNA sequence is completely contained in the signal data, but the shortcomings of the basecalling tools prevent its correct interpretation. We can develop better basecallers for nanopore sequencing to reduce the second type of errors.

For the current MinION's chemistry, the frequency of sequencer measurement is 4 kHz while single-stranded DNA molecule moves forward at a speed of 450 bases/sec, which equals to nine measurements per base on average [17]. To obtain a nucleotide sequence from raw electrical current signals, a more sophisticated basecalling software is required. Early in Nanopore sequencing, the basecalling process was performed on cloud platform Metrichor and it has been discontinued since 2017 [18]. ONT now provides several basecalling tools such as MinKNOW, Albacore, Guppy and Scrappie. MinKNOW is the operating software that drives nanopore sequencing devices and can output the basecalled reads in FASTQ format. Albacore is the first local command-line general-purpose basecaller running on CPUs. Guppy is similar to Albacore, but it is designed to improve the basecalling speed with GPU devices. Scrappie is primarily a research basecaller for testing new approaches that will be integrated into new versions of Guppy and Albacore in the future. Albacore, Guppy and Scrappie all use an architecture which is called RGRGR in ONT because of its alternating reverse-GRU and GRU layers [11].

In addition to official basecalling tools, there are some independent basecallers developed by researchers in recent years, such as Nanocall [19], DeepNano [20] and Chiron [21]. Nanocall uses a hidden Markov model (HMM) to predict the nucleotide sequence. The hidden states of the HMM model represent all possible events. The emission probabilities are computed from pore models of nanopore sequencer. The transition probabilities of hidden states are trained on the training dataset. This method can model the movement and stay state of the molecule in the nanopore. DeepNano uses a bidirectional recurrent neural network to take longer range information in the event data. The input of each time step in RNN is an event which contains the mean and standard deviation value of a segment of raw signals. The output of each time step in the model is the nucleotide base. Both Nanocall and DeepNano are no longer under development. Chiron is an end-to-end, segmentation-free basecaller that directly translates the raw electrical sequence into DNA sequence. It couples a set of convolution layers with a set of recurrent layers. At the top of the network, a connectionist temporal classification decoder is used to provide the final nucleotide sequence based on the probabilities of output bases, which are calculated by convolution layers and recurrent layers. However, Chiron runs very slowly, which makes it impractical except for very small read sets [11].

Recently, Wick *et al.* [11] evaluated the performance of different basecalling tools including Albacore, Guppy, Scrappie and Chiron. They made a benchmark with several bacterial genomes. Although researchers are more interested in human DNA, Wick *et al.* still used bacterial genomes as the benchmark because bacterial genomes can provide a more confident reference sequence as ground truth to calculate accuracy than complex eukaryote genomes.

In this paper, we present SACall, an end-to-end basecalling tool, to generate the DNA sequence from raw electrical data directly without raw signal segmentation. SACall has a novel architecture combining convolution layers [22] and transformer self-attention layers. During the sequencing process in the Oxford Nanopore Sequencer, the electrical signals are continuous current values, which seem like a continuous curve. But the electrical signals retained in the off-machine data are sampled by the sequencer. The sequencer measures the electrical current at 4 kHz and the DNA moves forward at a rate of 450 bases/sec, equating to about nine measurements per base [17]. So the length of the raw signal is much longer than the length of the DNA sequence. Convolution layers are used to downsample the raw data and reduce the computation. In addition, convolution layers can capture the local current patterns in the signals. Behind the convolution layers, there are several self-attention layers, which mainly take the long-range dependencies of electrical currents. At the top of the model, SACall generates a DNA sequence from the probabilities of output bases by a connectionist temporal classification decoder. In addition, SACall is an open-source tool for users to basecall Nanopore sequencing data directly. At the same time, in order to allow users to train the model on their specific dataset, we give a training interface in SACall. We compare SACall with ONT basecallers including Albacore and Guppy from the perspective of read accuracy, assembly quality and consensus accuracy. SACall outperforms the ONT official basecalling tools on most of the test datasets.

2 Materials and Methods

2.1 Experiment Data

The benchmark dataset, released by wick *et al.* [11], is used to evaluate the performance of different basecalling tools. This dataset contains two parts: training dataset and test dataset. The training dataset consists of fifty individual species genomes, which are thirty *Klebsiella pneumoniae* genomes, ten genomes of other species of *Enterobacteriaceae* and ten genomes from other families of *Proteobacteria*. The test dataset is composed of nine species including three *Klebsiella pneumoniae*, *Shigella sonnei*, *Serratia marcescens*, *Haemophilus haemolyticus*, *Acinetobacter pittii*, *Stenotrophomonas maltophilia*, and *Staphylococcus aureus*. In addition to the raw signal data, the high-quality contigs for each species are also provided in the benchmark as the ground truth data. These contig sequences were assembled from Illumina reads by wick *et al.* using SKESA assembler [23] (v2.3.0).

2.2 Labeling DNA Sequence on the Raw Signal

To train a SACall basecalling model, we need to offer the data consists of raw signals and the corresponding DNA sequences. The raw signal should be labeled by the true nucleotide base. Since the basecalled DNA sequences have too many errors, we need to correct the basecalled reads first by mapping the reads to the reference. After correction, the raw signals are aligned to the corrected basecalled reads. Tombo (https://github.com/nanoporetech/tombo) is a tool for analyzing and visualizing raw nanopore signal. It corrects the errors in basecalled reads by mapping each read to the reference using minimap2 and annotates the corrected

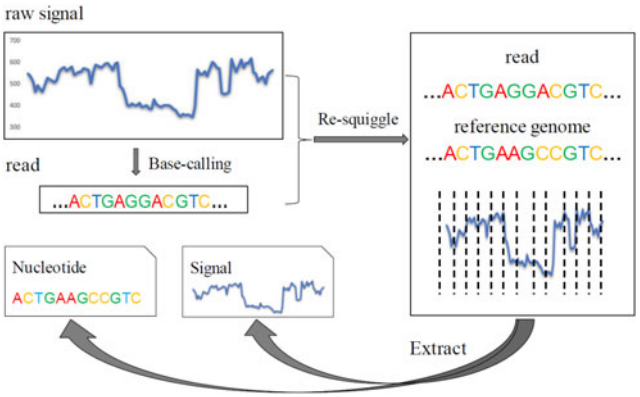


Fig. 1. The procedure for labeling raw signals. First, raw signals are basecalled by existing basecaller Guppy. Then Tombo is used to correct the basecalled reads and re-squiggle the raw signal with the corrected DNA sequence. Finally, we extract the input signal and the corresponding label sequence.

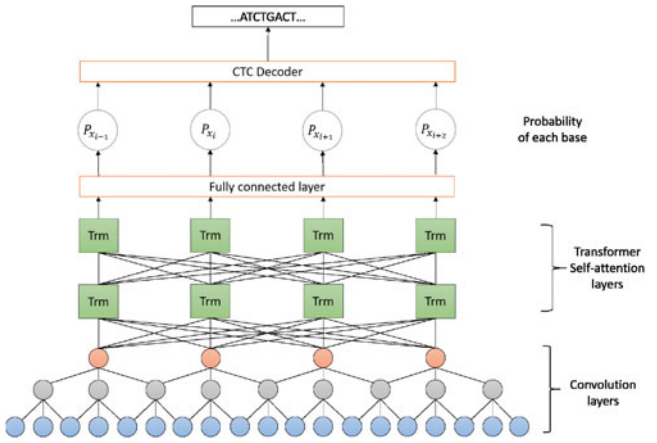


Fig. 2. An unrolled sketch of the SACall architecture. At the bottom of the model, there are two convolution layers to downsample the raw signals and extract the local pattern of the signals. The middle part of SACall is multiple self-attention layers for computing the correlation of any two positions in the signal. At the top of the network, a CTC decoder is used to generate the final DNA sequence with the correlation information in the signals from previous layers.

read to the raw signals by the dynamic time warping algorithm. We use Tombo to correct reads and label the raw signal. Then we can train the model with the signal and label sequence extracted from the re-squiggled files. The workflow of labeling raw electrical sequence can be seen in Fig. 1.

2.3 Deep Neural Network Architecture

Because the value of each signal is determined by the five nucleotides that stay in the nanopore [24], it is not easy to recognize the signal pattern of a single nucleotide. Besides, the 5-mer measured at the previous moment has a 4-base overlap with the 5-mer measured at the current moment. The signal at each time step is related to the surrounding signals. As we known, deep neural network is good at extracting high-level features in signal and image data. So, we have developed a novel deep learning architecture that combines the convolution layers and the transformer self-attention layer to extract the patterns and correlations from the raw electrical current of ONT sequencer. The whole architecture is shown in Fig. 2. Since each base of DNA has multiple measurements, the convolution layer acts as a downsampling of the raw data to reduce computational cost and calculate the local signal features. After that, the transformer self-attention layer [25] is used to learn the correlation of signals at different time steps. Because of the multiple measurements of each base, the target DNA sequence is shorter than the input signal sequence. Excessive measurements of each base will cause insertion errors for this base. So SACall uses a connectionist temporal classification (CTC) [26] layer at the top of the neural network to predict the label of each time step. Then we achieve the final nucleotide sequence by removing the repeated characters and blank characters in the prediction. Next, we will explain different modules in the neural network architecture.

Due to the differences in mean and variance of raw signals for a different batch of Nanopore sequencing data [24], raw signals in each read are normalized using median shift and median absolute deviation (MAD) scale parameters as follows [27]:

$$NormSignal = \frac{RawSignal - Shift}{Scale}. \tag{1}$$

Considering computing speed and memory, the normalized raw signal sequence is divided into pieces, each of which is 2,048 in length. The input of the model is one piece with T time steps $x = [x_1, x_2, \ldots, x_T]$ and the corresponding DNA sequence with m bases $y = [y_1, y_2, \ldots, y_m]$ where $y_i \in \{A, T, C, G\}$.

Convolution Module. At the bottom of the architecture, there is a convolution module. In this module, two convolution layers are used to downsample signals and extract local patterns. The first convolution layer has a $(1 * k)$ convolution filter, $(1 * p)$ stride and $d_{model}/2$ output channel. k is the kernel size of the convolution, p is the step size that the kernel moves in the convolution operation and d_{model} is the number of output channels in the convolution. The second convolution layer also has a $(1 * k)$ convolution filter, $(1 * p)$ stride and d_{model} output channel. The signal vector x in the convolution layers is calculated as follows [28]:

$$x_c = Conv(x) = \sum_{i=0}^{T} \sum_{j=0}^{k-1} w_j * x_{i+j}. \tag{2}$$

Here w is the weight of convolution kernel. In the model, we set parameters $k = 3$, $p = 2$ and $d_{model} = 256$.

In the convolution module, there is a batch normalization layer behind each convolution layer to prevent the mean and variance from saturation. The batch normalization [29] over a batch data is

$$\begin{aligned} BatchNorm_{\gamma,\beta,\epsilon}(x_{bn}) = & \frac{\gamma}{\sqrt{Var[x_{bn}] + \epsilon}} \cdot x_{bn} \\ & + (\beta - \frac{\gamma E[x_{bn}]}{\sqrt{Var[x_{bn}] + \epsilon}}). \end{aligned} \tag{3}$$

Here γ, β and ϵ are parameters to be learned. Batch normalization is helpful to improve the stability of the network [29]. Behind each batch normalization, there is a rectified linear unit (ReLU) [30] activation operation,

which sets a negative value to zero

$$Relu(x_{ac}) = \begin{cases} 0, if\, x_{ac} < 0; \\ x_{ac}, otherwise. \end{cases} \tag{4}$$

Transformer Self-Attention Module. Following the convolution module, there are multiple transformer self-attention layers, which are widely used in natural language processing (NLP) models like BERT [31] and GPT-2 [32]. They perform strongly on various tasks such as machine translation, document generation, and syntactic parsing. In our model, we use six transformer self-attention layers as a feature extractor. In each self-attention layer, the similarity is calculated for any two signals in the raw signal sequence, which represents the contextual relevance of signals. The core modules of the transformer are Positional Encoding, Scaled Dot-Product Attention, Multi-Head Attention and Position-wise Feed-Forward Network.

In recurrent neural networks, current hidden state calculation depends on the previous hidden state, which limits the ability of model parallelization. To overcome this limitation, the transformer model contains no recurrence. To utilize the sequential information of the raw signal data, positional information of each signal in the signal sequence is encoded into the input vector by positional coding [25]

$$PositionEncode(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \tag{5}$$

$$PositionEncode(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right). \tag{6}$$

Here pos is the position in the input sequence, and i is the dimension of the channel.

In the transformer, scale dot production is used as the attention strategy. The input of scaled dot-product attention consists of queries Q (with channel dimension d_k), keys K (with channel dimension d_k) and values V (with channel dimension d_v). The attention matrix is calculated as follows [25]:

$$Q = linear_Q(x_{attn}) \tag{7}$$

$$K = linear_K(x_{attn}) \tag{8}$$

$$V = linear_V(x_{attn}) \tag{9}$$

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{10}$$

Multi-head attention allows the transformer model to combine the information from different representation subspaces when calculating the scaled dot-product attention [25].

$$MultiHead(Q, K, V) = Concat(head_1, head_2, \ldots, head_h)W^O \tag{11}$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V). \tag{12}$$

Here $W_i^Q \in R^{d_{model} \times d_k}$, $W_i^K \in R^{d_{model} \times d_k}$ and $W_i^V \in R^{d_{model} \times d_v}$ are linear projection parameters to be learned and h is the hyperparameter. In our model, we set $h = 8$.

Following each attention layer, there is a fully connected feed-forward network which consists of two linear layers and a ReLU activation:

$$FFN(x_f) = ReLU(x_f W_{f_1} + b_{f_1}) \times W_{f_2} + b_{f_2}. \tag{13}$$

Decoder Module. At the top of the architecture, there is a decoder module that generates the final nucleotide sequence from high-dimension representation of correlation information in electrical signals. The correlation information is obtained in the previous convolution module and transformer self-attention module. In the decoder module, we use a fully connected layer followed by a softmax operation to calculate the probability of the characters at each position. The softmax is calculated as

$$P(o_i = c) = \frac{e^{W_c h_i}}{\sum_c e^{W_c h_i}}, \tag{14}$$

where o_i is the predicted symbol at position i, and h_i is the hidden state after transformer self-attention layer at position i. The output symbol is $c \in (A, T, C, G, -)$ and the character '-' represents a blank symbol. During training, the CTC decoder behind softmax operation uses a CTC loss function [26] to compute the distance of the distribution of high-level features of raw signal and nucleotide sequence label y. During testing, the CTC decoder uses a beam search decoding algorithm to iteratively generate candidate characters (beams) and rate these candidate characters. At each position, only the beams with top $beam\_width$ scores from the previous time-step are retained and form a new beams together with the characters at current time-step. In our model, we set $beam\_width = 3$. At the end of this procedure, the sequence with the highest score will be chosen as the final nucleotide sequence after removing the blank symbol in this sequence.

2.4 Training and Basecalling

In the training stage, the Adam optimizer [33] with $lr = 1e^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e^{(} - 8)$ and $weight\_decay = 0.01$ is used to update the parameters. Scheduling the learning rate with warmup strategy is used to prevent excessive initial learning rates, which is calculated as

$$lr = d_{model}^{-0.5} \cdot min(n^{-0.5}, n \cdot warmup^{-1.5}). \tag{15}$$

Here n is the number of training step, and the parameter $warmup$ is set to 10,000. All dropout ratio [34] in SACall are set to 0.1.

SACall is an open-source tool for basecalling Oxford Nanopore reads, which contains two main modules, training and basecalling. Users can train new models on specific datasets with the SACall-training module and then load the models into the SACall-calling module to basecall nanopore reads. Besides, users also can directly use the SACall-calling module and the provided trained model for basecalling without training new models. The default model of SACall is mainly trained from the genome of Klebsiella pneumoniae. Main steps of the SACall-training module include:

TABLE 1
Read Level Error Rate of Four Basecalling Models on the Test Dataset

Dataset	Basecaller	Deletion rate(%)	Insertion rate(%)	Mismatch rate(%)	Error rate(%)
Klebsiella Pneumoniae NUH29	SACall	4.93	2.63	3.50	**11.06**
	Guppy-KP	5.68	3.33	4.09	13.10
	Guppy	4.85	5.51	5.34	15.69
	Albacore	4.85	5.76	5.08	15.69
Klebsiella Pneumoniae KSB2	SACall	5.26	2.31	3.69	**11.26**
	Guppy-KP	5.90	3.38	4.56	13.84
	Guppy	5.42	4.65	5.66	15.73
	Albacore	5.37	4.99	5.45	15.80
Klebsiella Pneumoniae INF042	SACall	5.13	2.27	3.56	**10.95**
	Guppy-KP	5.67	3.34	4.42	13.44
	Guppy	5.27	4.60	5.48	15.35
	Albacore	5.18	5.00	5.30	15.48
Serratia Marcescens	SACall	4.37	2.42	3.72	**10.51**
	Guppy-KP	5.21	3.28	4.40	12.90
	Guppy	4.47	3.56	4.56	12.59
	Albacore	4.50	4.87	5.18	14.55
Haemophilus Haemolyticus	SACall	3.87	2.59	3.25	**9.71**
	Guppy-KP	4.56	3.79	4.21	12.56
	Guppy	3.57	4.52	4.05	12.14
	Albacore	3.23	6.40	4.30	13.93
Stenotrophomonas Maltophilia	SACall	5.20	2.60	3.62	**11.42**
	Guppy-KP	5.90	3.28	4.12	13.30
	Guppy	5.08	5.10	5.18	15.36
	Albacore	4.98	5.40	4.85	15.24
Shigella Sonnei	SACall	5.27	2.20	3.99	**11.47**
	Guppy-KP	5.76	3.21	5.05	14.03
	Guppy	5.80	3.12	5.31	14.23
	Albacore	5.98	3.46	5.39	14.84
Acinetobacter Pittii	SACall	4.87	2.43	3.94	**11.23**
	Guppy-KP	5.59	3.38	4.61	13.58
	Guppy	4.45	3.38	4.35	12.18
	Albacore	4.52	4.91	5.01	14.44
Staphylococcus Aureus	SACall	4.17	2.03	3.02	9.23
	Guppy-KP	4.62	2.76	3.47	10.85
	Guppy	3.73	2.27	2.88	**8.87**
	Albacore	4.12	3.11	3.40	10.64

(1) Translating Nanopore raw signal data to noisy reads using existing basecalling tool Guppy.
(2) Correcting the basecalled reads and annotating raw signals using the re-squiggle module of Tombo.
(3) Extracting signals and label sequences from resquiggled fast5 files as the inputs of the neural network.
(4) Training the neural network and exporting the basecalling model.

After training, we can use SACall to basecall Nanopore raw signals. Loading the model parameter file and original sequencing fast5 files, the SACall-calling module will output the basecalled fasta file.

3 Experiments and Results

Currently, the most widely used basecalling tools are Albacore and Guppy launched by Oxford Nanopore Technologies. However, Oxford Nanopore Technologies has not released the training data of these two basecallers. So we can only train our model with the different training datasets. Wick *et al.* released a Guppy model (v2.2.3) [11] that was trained on the approximate $1/10$ of reads in the training benchmark. Here we named it Guppy-KP to distinguish it from the general Guppy model released by ONT. To train SACall, we also randomly picked $1/10$ of reads in the training benchmark. Next, we evaluate the performance of these four tools, SACall, Guppy-KP, Guppy (v2.3.8) and Albacore (v2.3.4), on the test benchmark.

3.1 Error Rate of Basecalled Reads

Table 1 presents the error rate of four basecaller methods including SACall, Guppy-KP, Guppy and Albacore on the test dataset. Deletion, insertion, and mismatch rates are defined as the number of deleted, inserted, and mismatched bases divided by the alignment length. Error rate is defined as the sum of deletion, insertion, and mismatch rates. On the data of Klebsiella Pneumoniae NUH29, SACall achieves the error rate of 11.06 percent, which is smaller than that of other basecallers. On the data of Klebsiella Pneumoniae KSB2, the error rate of SACall basecalled reads is 11.26 percent, the error rate of Albacore basecalled reads is 15.80 percent, and the error rate of Guppy basecalled reads is 15.73 percent. On the data of Klebsiella Pneumoniae genomes INF042, SACall achieves the smallest error rate among four basecallers, which is 10.95 percent. On the data of Serratia Marcescens genome, SACall has the error rate of 10.51 percent, the error rate of Guppy-KP is 12.90 percent, and the

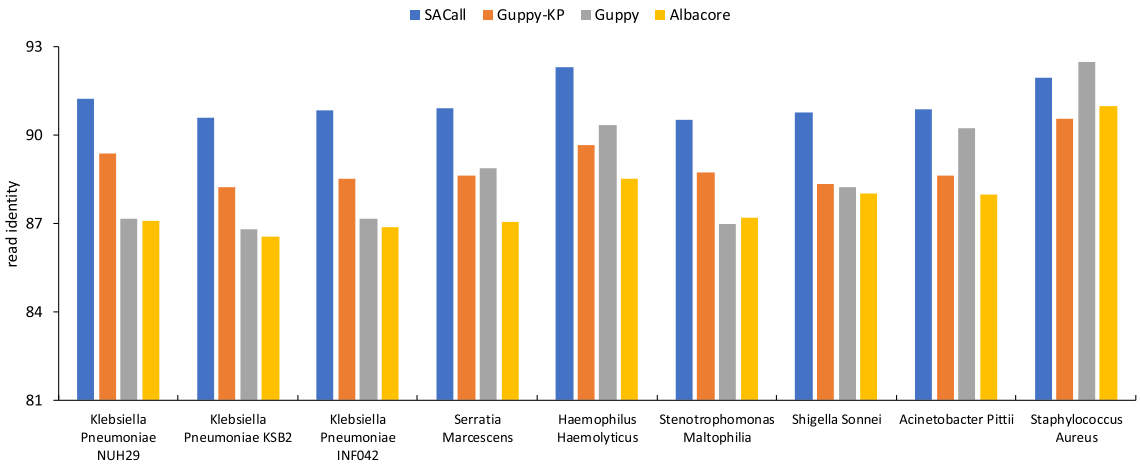


Fig. 3. The read level identity of four basecalling models on the test dataset.

error rate of Guppy is 12.59 percent. On the data of Haemophilus haemolyticus genome, SACall achieves the error rate of 9.71 percent, which is more accurate than Guppy-KP, Guppy, and Albacore. On the data of Stenotrophomonas Maltophilia genome, the error rate of SACall is 11.42 percent which is the smallest among four basecallers. On the data of Shigella Sonnei genome, SACall has an error rate of 11.47 percent, while the error rates of the other three basecallers are bigger than the error rate of SACall. On the data of Acinetobacter Pittii genome, the error rate of SACall is 11.23 percent, which is the smallest among four basecallers. On the data of Staphylococcus Aureus genome, SACall obtains the error rate of 9.23 percent, and Guppy achieves the smallest error rate of 8.87 percent. As a whole, SACall achieves the smallest error rate on 8 of 9 genomes in the test dataset. The error rate of SACall is a little bigger than Guppy only on the data of Staphylococcus Aureus genome.

3.2 Read Identity of Basecalled Reads

To assess the read identity of different basecallers, we align the basecalled reads to the high-quality contigs by minimap2 [35]. The read identity is calculated as follows:

$$identity = \frac{M}{M + S + I + D} * 100\%. \tag{16}$$

Here, M is the number of match bases and S is the number of mismatch bases. I represents the number of insertion bases in the alignment, and D represents the number of deletion bases. The overall identity is the median of all reads' identities. The read identity results in Fig. 3 show that the basecalled reads generated by SACall have the highest identity in all genomes except Staphylococcus Aureus. It also exceeds the Guppy-KP model, which trained on the same training data.

3.3 Quality of Assemblies From Reads Basecalled by Different Basecaller

We use Flye(v2.4.2) to generate assembly from the reads basecalled by different basecaller and evaluate the quality of the genome assembly. Flye is a very fast and accurate de novo assembler designed for single-molecule sequencing reads. Then a quality assessment tool, Quast, is used to evaluate the contigs of the assembly. The Quast report of assemblies is presented in Table 2. From the table, we can see that assemblies from SACall basecalled reads have fewer mismatch errors per 100k basepairs and indel errors per 100 k basepairs. To calculate the identity of each assembly, we divided the assembly of each reads set into the size of 100 kbp. Each divided piece is aligned to the high-quality contigs by minimap2. The identity is defined as the number of matching bases divided by the alignment length. The final assembly identity is the median of the identities of these 10 kbp assembly segments. The results show that reads basecalled by SACall have a higher assembly identity than reads basecalled by other basecalling tools. In most of the datasets, assembly from reads generated by SACall has the highest assembly identity.

3.4 Consensus Accuracy and Error Summary of Polished Assemblies

To access the consensus accuracy, we polish the previous assemblies with four rounds of Racon and one round of Medaka to generate the consensus sequences as ONT officials suggested. Racon [33] is an ultrafast consensus module for raw de novo genome assembly of long uncorrected reads. And Medaka (https://github.com/nanoporetech/medaka) is a tool to create a consensus sequence from nanopore sequencing data developed by Oxford Nanopore Technologies. The assembly identity of each round polishing iteration is shown in Fig. 4. For understanding the effect of four basecallers on different kinds of consensus errors, we count the number of errors for six different types respectively in the polished assembly, which include homopolymer insertion errors, other insertion errors, homopolymer deletion errors, other deletion errors, substitution errors and Dcm errors. Dcm errors are the errors occurring in the CCAGG/CCTGG Dcm motif. Homopolymer errors are changes in the length of a homopolymer (three or more bases in length) compared to the reference sequence. Other insertion and deletion errors refer to the insertion or deletion errors that occur at positions other than homopolymers. Substitution errors refer to the mismatch of bases in contigs and references. The consensus error summaries of different basecallers on nine bacterial genomes are presented in Fig. 5. ONT official basecaller Guppy and Albacore perform poorly on Dcm sites in genomes including K.pneumoniae

TABLE 2
The Quality of Assemblies From Reads Sets Generated by Four Basecalling Models

Genome	Basecaller	Misassemblies	Mismatchs (bp/100k)	Indels (bp/100k)	N50 (Mbp)	Genome fraction	Relative length	Identity (%)
Klebsiella Pneumoniae NUH29	SACall	0	8.02	179.41	5.1	99.99%	100.10%	**99.81**
	Guppy-KP	0	18.25	319.17	5.1	99.99%	100.26%	99.67
	Guppy	0	416.42	503.09	5.2	100.00%	100.41%	99.07
	Albacore	0	406.69	583.24	5.2	99.98%	100.45%	98.99
Klebsiella Pneumoniae KSB2	SACall	0	8.87	146.22	5.2	99.99%	100.05%	**99.84**
	Guppy-KP	0	23.48	293.52	5.2	100.00%	100.23%	99.68
	Guppy	0	428.95	322.17	5.2	99.99%	100.18%	99.24
	Albacore	1	421.19	410.68	5.2	99.95%	100.23%	99.16
Klebsiella Pneumoniae INF042	SACall	0	5.68	146.46	5.3	100.00%	100.06%	**99.85**
	Guppy-KP	0	18.06	292.23	5.4	100.00%	100.24%	99.69
	Guppy	0	255.27	327.19	5.3	100.00%	100.23%	99.41
	Albacore	0	255.79	410.21	5.4	100.00%	100.31%	99.33
Serratia Marcescens	SACall	1	43.71	337.58	5.5	99.92%	103.90%	**99.64**
	Guppy-KP	4	87.86	502.04	5.5	99.24%	103.90%	99.41
	Guppy	1	120.74	444.11	5.5	99.41%	104.55%	99.43
	Albacore	1	116.76	712.57	5.5	99.25%	104.05%	99.16
Haemophilus Haemolyticus	SACall	3	14.88	428.25	2.1	100.00%	100.80%	**99.55**
	Guppy-KP	3	24.48	917.01	2.1	100.00%	101.33%	99.04
	Guppy	3	16.40	556.48	2.1	100.00%	100.98%	99.41
	Albacore	3	22.91	956.52	2.1	100.00%	101.39%	98.96
Stenotrophomonas Maltophilia	SACall	0	14.68	141.11	4.8	100.00%	100.09%	**99.84**
	Guppy-KP	0	26.57	199.18	4.8	100.00%	100.13%	99.78
	Guppy	0	41.31	366.69	4.8	100.00%	100.35%	99.58
	Albacore	0	23.99	376.22	4.8	100.00%	100.36%	99.59
Shigella Sonnei	SACall	0	9.21	171.07	4.8	100.00%	100.08%	**99.82**
	Guppy-KP	0	21.66	351.57	4.8	100.00%	100.31%	99.63
	Guppy	0	290.03	227.08	4.8	100.00%	100.55%	99.48
	Albacore	0	287.69	274.00	4.8	99.99%	100.12%	99.43
Acinetobacter Pittii	SACall	0	13.45	410.04	3.8	99.97%	100.31%	99.58
	Guppy-KP	0	28.00	742.05	3.8	99.98%	100.69%	99.22
	Guppy	0	14.58	364.71	3.8	99.97%	100.32%	**99.62**
	Albacore	0	26.64	707.10	3.8	99.97%	100.64%	99.25
Staphylococcus Aureus	SACall	0	9.72	449.61	2.9	100.00%	100.40%	99.54
	Guppy-KP	0	12.09	774.65	2.9	100.00%	100.78%	99.21
	Guppy	0	9.13	309.36	2.9	100.00%	100.30%	**99.68**
	Albacore	0	12.20	506.36	2.9	100.00%	100.48%	99.50

NUH29, K.pneumoniae KSB2, K.pneumoniae INF042 and Shigella Sonnei, but Guppy-KP and SACall have less Dcm errors on these genomes. This is because the genomes in the dataset used to train Guppy-KP and SACall are close to the genomes of K.pneumoniae NUH29, K.pneumoniae KSB2, K.pneumoniae INF042 and Shigella Sonnei. Guppy-KP and SACall have learned the information of Dcm methylation on these genomes. However, on these genomes, the Dcm error rate of SACall basecalled reads is still smaller than that of Guppy-KP basecalled reads. Besides Dcm errors, incorrect lengths of homopolymers make up the majority of consensus errors. In terms of homopolymer insertion errors, SACall significantly reduces the homopolymer insertion errors in basecalled reads compared to the other three tools on most of the species including K.pneumoniae NUH29, K. pneumoniae KSB2, K.pneumoniae INF042, Serratia Marcescens, Haemophilus Haemolyticus and Stenotrophomonas Maltophilia. In terms of homopolymer deletion errors, the reads basecalled by Guppy and Guppy-KP have the smallest error rate. The amount of homopolymer deletion errors in SACall basecalled reads is similar to that in Albacore basecalled reads. For other insertion errors, SACall reduces such errors in basecalled reads compared to other basecallers. In genomes of K.pneumoniae NUH29, K.pneumoniae KSB2, K.pneumoniae INF042, Serratia Marcescens, Haemophilus Haemolyticus, Stenotrophomonas Maltophilia, Shigella Sonnei and Staphylococcus Aureus, SACall basecalled reads have the smallest insertion error rate. For other deletion errors, on most of the genomes, SACall obtains similar performance with the other three basecallers. In terms of substitution errors, SACall reduces this type of errors in basecalled reads than other basecaller. In genomes including K.pneumoniae NUH29, K.pneumoniae KSB2, K. pneumoniae INF042, Serratia Marcescens, Haemophilus Haemolyticus, Stenotrophomonas Maltophilia and Shigella Sonnei, SACall obtains the smallest substitution error rate. In general, SACall performs better than the ONT official basecallers on consensus accuracy, mainly due to the reduction of Dcm errors, homopolymer insertion errors, other insertion errors and substitution errors.

3.5 Comparing the Speed of Different Basecallers

In terms of basecalling speed, we perform SACall and Guppy on Nvidia TITAN V GPU. Since Albacore only

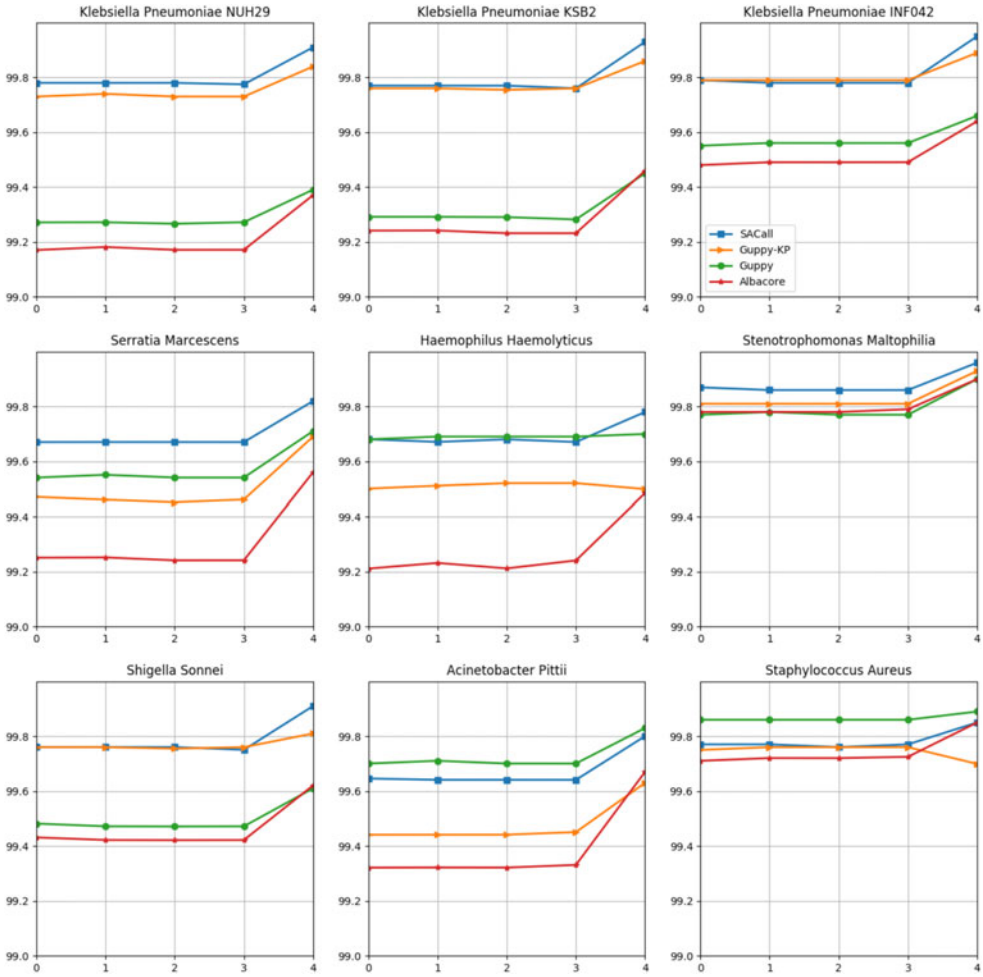


Fig. 4. The consensus identity of four basecallers on the test dataset. Each assembly is polished by four rounds Racon and one round Medaka. 0 to 3 on the horizontal axis means Racon polishing, 4 means Medaka polishing.

runs on central processing unit processor (CPU), we test Albacore on Intel(R) Xeon(R) E5-2630 v4 @ 2.20 GHz CPU. The basecall rate of SACall is 80,000 bases per second on average, and Albacore has a speed of 120,000 bp/sec when using 40 threads. Of all the tools, the speed of Guppy basecalling is the fastest which is 800,000 bp/sec.

4 Conclusion

Currently, Nanopore basecalling tools under development are all based on recurrent neural networks and its variants LSTM and GRU. ONT official basecaller Albacore and Guppy use the alternative reverse-GRU layers and GRU layers architecture. Chiron uses the architecture

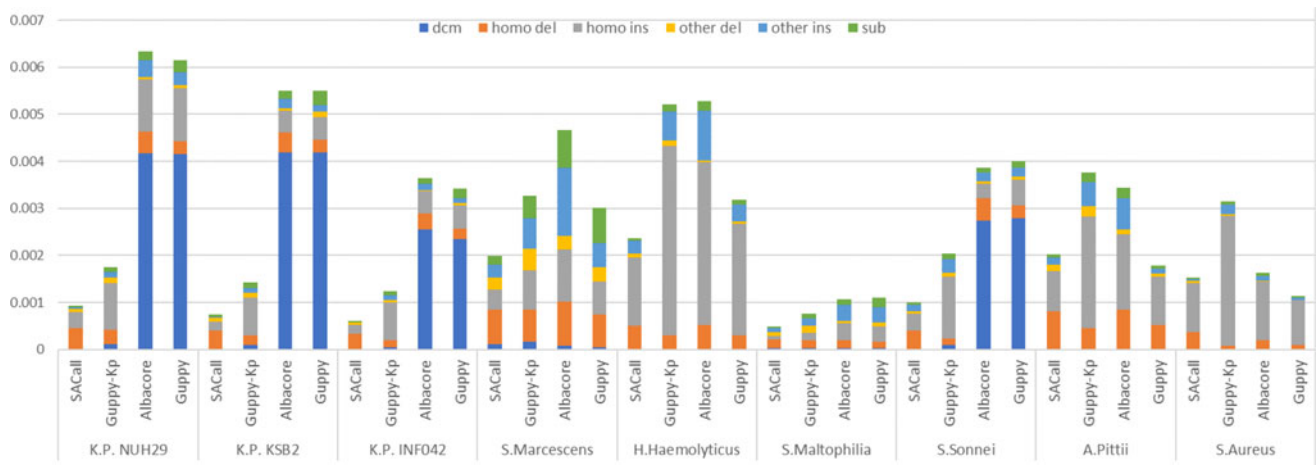


Fig. 5. The polish error summary of four basecallers. The error types include Dcm methylation errors, homopolymer deletion errors, homopolymer insertion errors, other deletion errors, other insertion errors, and substitution errors.

combining the convolution layers and recurrent neural networks. In this study, we present a novel architecture that combines the convolution layers and self-attention layers to learn the correlation in nanopore sequencing signals. In this model, the local pattern of sequencing raw signal is captured by multiple convolution layers. The self-attention layers are used to calculate the similarity at any two positions in the sequence, which provides the contextual relevance of signals. Finally, the basecalled DNA sequence is generated with the connectionist temporal classification decoder by a beam search algorithm. For speed and memory considerations, the raw signals are divided into pieces with the size of 2,048. Each signal piece is fed into the neural network to generate the nucleotide sequence fragments. Then SACall concatenates these DNA fragments to form the final basecalled DNA sequence.

Compared with Albacore, Guppy and Guppy-KP trained on the same dataset as SACall, our proposed basecaller gets the highest accuracy on 8 of 9 datasets in the benchmark in terms of read level identity. From the perspective of the quality of genome assembly, the assembly from SACall basecalled reads achieves the highest identity in most of the test datasets. The misassemblies, N50 and genome fraction of assembly from reads set basecalled by SACall are close to the assembly from reads set generated by ONT basecallers. To access the consensus accuracy of SACall basecalled reads, we polish the assemblies with four rounds of Racon and one round of Medaka. The polished assembly from SACall basecalled reads achieves the highest consensus accuracy on 7 of 9 datasets. In the analysis of consensus errors, we found that SACall outperforms other tools, mainly due to the reduction of Dcm-methylation errors, homopolymer insertion errors, other insertion errors and substitution errors.

In short, we develop a new basecalling tool for nanopore sequencing, which can generate high accuracy basecalled reads. With SACall, users can train their models on datasets for some specific genomic applications such as detecting base modification [36], [37]. At present, we do not consider DNA base modification when training the model. In future work, we will directly predict base modification during the basecalling process. This requires us to annotate the base modification on the raw electrical current signal when preparing the training data.

Acknowledgments

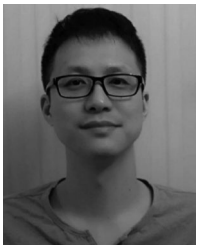
This work was supported in part by the National Natural Science Foundation of China under Grants (Nos. U1909208 and 61772557), 111 Project (No. B18059), Hunan Provincial Science and Technology Program (No. 2018wk4001) to J.W., the U. S. National Institute of Food and Agriculture (NIFA) under grant 2017-70016-26051 and the U.S. National Science Foundation (NSF) under grants ABI-1759856 to F.L. We thank Ryan Wick *et al.* for releasing their bacterial sample dataset. We thank the Oxford Nanopore technique support specialist Chen Du for answering questions about the software and resource requirements of Guppy and Albacore.

References

[1] E. E. Schadt, S. Turner, and A. Kasarskis, "A window into third-generation sequencing," *Hum. Mol. Genetics*, vol. 19, no. R2, pp. R227–R240, 2010.
[2] C.-S. Chin *et al.*, "Nonhybrid, finished microbial genome assemblies from long-read SMRT sequencing data," *Nat. Methods*, vol. 10, no. 6, 2013, Art. no. 563.
[3] J. Eid *et al.*, "Real-time DNA sequencing from single polymerase molecules," *Science*, vol. 323, no. 5910, pp. 133–138, 2009.
[4] M. Jain, I. T. Fiddes, K. H. Miga, H. E. Olsen, B. Paten, and M. Akeson, "Improved data analysis for the minion nanopore sequencer," *Nat. Methods*, vol. 12, no. 4, 2015, Art. no. 351.
[5] I. Sović, M. Šikić, A. Wilm, S. N. Fenlon, S. Chen, and N. Nagarajan, "Fast and sensitive mapping of nanopore sequencing reads with graphmap," *Nat. Commun.*, vol. 7, 2016, Art. no. 11307.
[6] S. Koren *et al.*, "Hybrid error correction and de novo assembly of single-molecule sequencing reads," *Nat. Biotechnol.*, vol. 30, no. 7, 2012, Art. no. 693.
[7] J.-S. Seo *et al.*, "De novo assembly and phasing of a korean human genome," *Nature*, vol. 538, no. 7624, 2016, Art. no. 243.
[8] L. Shi *et al.*, "Long-read sequencing and de novo assembly of a chinese genome," *Nat. Commun.*, vol. 7, 2016, Art. no. 12065.
[9] A. R. Wu, J. Wang, A. M. Streets, and Y. Huang, "Single-cell transcriptional analysis," *Annu. Rev. Anal. Chem.*, vol. 10, pp. 439–462, 2017.
[10] J. T. Simpson, R. E. Workman, P. Zuzarte, M. David, L. Dursi, and W. Timp, "Detecting DNA cytosine methylation using nanopore sequencing," *Nat. Methods*, vol. 14, no. 4, 2017, Art. no. 407.
[11] R. R. Wick, L. M. Judd, and K. E. Holt, "Performance of neural network basecalling tools for Oxford nanopore sequencing," *Genome Biol.*, vol. 20, no. 1, 2019, Art. no. 129.
[12] M. Jain *et al.*, "Nanopore sequencing and assembly of a human genome with ultra-long reads," *Nat. Biotechnol.*, vol. 36, no. 4, 2018, Art. no. 338.
[13] A. Payne, N. Holmes, V. Rakyan, and M. Loose, "BulkVis: A graphical viewer for Oxford nanopore bulk FAST5 files," *Bioinformatics*, vol. 35, no. 13, pp. 2193–2198, 2019.
[14] P. Sarkozy, Á. Jobbágy, and P. Antal, "Calling homopolymer stretches from raw nanopore reads by analyzing k-mer dwell times," in *Proc. Eur. Med. Biol. Eng. Conf.*, 2017, pp. 241–244.
[15] E. A. Manrao *et al.*, "Reading DNA at single-nucleotide resolution with a mutant MspA nanopore and phi29 DNA polymerase," *Nat. Biotechnol.*, vol. 30, no. 4, 2012, Art. no. 349.
[16] G. M. Cherf, K. R. Lieberman, H. Rashid, C. E. Lam, K. Karplus, and M. Akeson, "Automated forward and reverse ratcheting of DNA in a nanopore at 5-å precision," *Nat. Biotechnol.*, vol. 30, no. 4, 2012, Art. no. 344.
[17] R. R. Wick, L. M. Judd, and K. E. Holt, "Deepbinner: Demultiplexing barcoded oxford nanopore reads with deep convolutional neural networks," *PLoS Comput. Biol.*, vol. 14, no. 11, 2018, Art. no. e1006583.
[18] F. J. Rang, W. P. Kloosterman, and J. de Ridder, "From squiggle to basepair: Computational approaches for improving nanopore sequencing read accuracy," *Genome Biol.*, vol. 19, no. 1, 2018, Art. no. 90.
[19] M. David, L. J. Dursi, D. Yao, P. C. Boutros, and J. T. Simpson, "Nanocall: An open source basecaller for Oxford nanopore sequencing data," *Bioinformatics*, vol. 33, no. 1, pp. 49–55, 2016.
[20] V. Boža, B. Brejová, and T. Vinař, "DeepNano: Deep recurrent neural networks for base calling in minion nanopore reads," *PLoS One*, vol. 12, no. 6, 2017, Art. no. e0178751.
[21] H. Teng, M. D. Cao, M. B. Hall, T. Duarte, S. Wang, and L. J. Coin, "Chiron: Translating nanopore raw signal directly into nucleotide sequence using deep learning," *GigaScience*, vol. 7, no. 5, 2018, Art. no. giy037.
[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[23] A. Souvorov, R. Agarwala, and D. J. Lipman, "SKESA: Strategic k-mer extension for scrupulous assemblies," *Genome Biol.*, vol. 19, no. 1, 2018, Art. no. 153.
[24] N. J. Loman, J. Quick, and J. T. Simpson, "A complete bacterial genome assembled de novo using only nanopore sequencing data," *Nat. Methods*, vol. 12, no. 8, 2015, Art. no. 733.
[25] A. Vaswani *et al.*, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

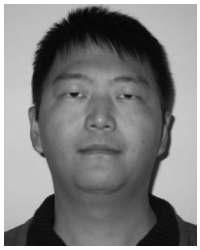
[26] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. 23rd Int. Conf. Mach. Learn.*, 2006, pp. 369–376.
[27] M. H. Stoiber *et al.*, "De novo identification of DNA modifications enabled by genome-guided nanopore signal processing," 2016, *bioRxiv*, Art. no. 094672.
[28] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.
[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.
[30] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn.*, 2010, pp. 807–814.
[31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.
[32] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.
[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[34] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 56, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html
[35] H. Li, "Minimap2: Pairwise alignment for nucleotide sequences," *Bioinformatics*, vol. 34, no. 18, pp. 3094–3100, 2018.
[36] Q. Liu, L. Fang, G. Yu, D. Wang, C.-L. Xiao, and K. Wang, "Detection of DNA base modifications by deep recurrent neural network on Oxford nanopore sequencing data," *Nat. Commun.*, vol. 10, no. 1, 2019, Art. no. 2449. [Online]. Available: https://doi.org/10.1038/s41467–019-10168-2
[37] P. Ni *et al.*, "DeepSignal: Detecting DNA methylation state from Nanopore sequencing reads using deep-learning," *Bioinformatics*, vol. 35, no. 22, pp. 4586–4595, Apr. 2019. [Online]. Available: https://doi.org/10.1093/bioinformatics/btz276

Neng Huang received the BS degree in mathematics from the University of South China, China. Currently he is working toward the PhD degree in computer science, Central South University, Changsha, Hunan, P.R. China. He His main research interests include deep learning, bioinformatics, and algorithm analysis.



Fan Nie received the bachelor's degree in applied mathematics and master's degree in computational mathematics from the Huazhong University of Science and Technology, China. Now he is working toward the PhD degree in computer science at Central South University, China. His main research interests include sequencing error correction and genome assembly.

Peng Ni received the BS degree in computer science from Shandong University, China, and the MS degree in computer science from Central South University, P.R. China. Now he is working toward the PhD degree in computer science at Central South University, China. His main research interests include DNA and RNA methylation detection.



Feng Luo (Member, IEEE) received the PhD degree in computer science from the University of Texas at Dallas, Richardson, Texas, August 2004. He is an associate professor of School of Computing, Clemson University, Clemson, South Carolina. He was a post-doctorial senior research associate at the Department of Pathology, University of Texas Southwestern Medical Center at Dallas, Dallas, Texas and joined Clemson, in 2006. He is also a member of ACM. His current research focusing areas include computational genomics and genetics, high throughput biological data analysis, data-intensive bioinformatics, network biology, personalized medicine, disease diagnostics, big data analytics, deep learning and application.

Jianxin Wang (Senior Member, IEEE) received the BS and MS degrees in computer science and application from the Central South University of Technology, P. R. China, and the PhD degree in computer science and technology from Central South University, China. Currently, he is the dean and a professor with the School of Computer Science and Engineering, Central South University, Changsha, Hunan, P.R. China. He is also a leader in Hunan Provincial Key Lab on Bioinformatics, Central South University, Changsha, Hunan, P.R. China. His current research interests include algorithm analysis and optimization, parameterized algorithm, bioinformatics and computer network. He has published more than 200 papers in various International journals and refereed conferences. He has been on numerous program committees and NSFC review panels, and served as editors for several journals such as the *IEEE/ACM Transactions On Computational Biology and Bioinformatics (TCBB)*, *International Journal of Bioinformatics Research and Applications*, *Current Bioinformatics*, *Current Protein & Peptide Science*, *Protein & Peptide Letters*.