

RESEARCH

Open Access



# DeepSelectNet: deep neural network based selective sequencing for oxford nanopore sequencing

Anjana Senanayake<sup>1\*</sup>, Hasindu Gamaarachchi<sup>2,3</sup>, Damayanthi Herath<sup>1</sup> and Roshan Ragel<sup>1</sup>

\*Correspondence:  
anjana.senanayake@eng.pdn.ac.lk

<sup>1</sup> Department of Computer Engineering, University of Peradeniya, Peradeniya, Sri Lanka

<sup>2</sup> Kinghorn Centre for Clinical Genomics, Garvan Institute of Medical Research, Darlinghurst, Australia

<sup>3</sup> School of Computer Science and Engineering, University of New South Wales, Sydney, Australia

## Abstract

**Background:** Nanopore sequencing allows selective sequencing, the ability to programmatically reject unwanted reads in a sample. Selective sequencing has many present and future applications in genomics research and the classification of species from a pool of species is an example. Existing methods for selective sequencing for species classification are still immature and the accuracy highly varies depending on the datasets. For the five datasets we tested, the accuracy of existing methods varied in the range of ~ 77 to 97% (average accuracy < 89%). Here we present DeepSelectNet, an accurate deep-learning-based method that can directly classify nanopore current signals belonging to a particular species. DeepSelectNet utilizes novel data preprocessing techniques and improved neural network architecture for regularization.

**Results:** For the five datasets tested, DeepSelectNet's accuracy varied between ~ 91 and 99% (average accuracy ~ 95%). At its best performance, DeepSelectNet achieved a nearly 12% accuracy increase compared to its deep learning-based predecessor SquiggleNet. Furthermore, precision and recall evaluated for DeepSelectNet on average were always > 89% (average ~ 95%). In terms of execution performance, DeepSelectNet outperformed SquiggleNet by ~ 13% on average. Thus, DeepSelectNet is a practically viable method to improve the effectiveness of selective sequencing.

**Conclusions:** Compared to base alignment and deep learning predecessors, DeepSelectNet can significantly improve the accuracy to enable real-time species classification using selective sequencing. The source code of DeepSelectNet is available at <https://github.com/AnjanaSenanayake/DeepSelectNet>.

**Keywords:** Selective sequencing, Deep learning, Nanopore

## Background

### Problem

Oxford Nanopore Technologies (ONT) introduced *MinION*, the first portable commercial sequencer of its kind in 2014 [1–3]. *MinION*, being a small handheld device, revolutionized genomics research through its superiority in portability [4–7]. The capability of nanopore sequencers to produce ultra-long reads is another advantage, especially in achieving better genome assemblies [8–11]. Nanopore sequencers provide access to



© The Author(s) 2023. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>. The Creative Commons Public Domain Dedication waiver (<http://creativecommons.org/publicdomain/zero/1.0/>) applies to the data made available in this article, unless otherwise stated in a credit line to the data.

real-time current signals when a DNA strand traverses through the pore [12]. This real-time streaming capability foster an exciting opportunity called selective sequencing [13] (also known as targeted sequencing, adaptive sampling), which has many current and promising future applications [14–19].

ONT's *ReadUntil* API enables real-time bi-directional communication with an ONT sequencer: to obtain the current signals of the DNA strands being sequenced and to send back signals such as accept/reject. A rejection signal reverses the voltage difference across the pore so that the DNA strand is ejected from the pore. This can be used to quickly free a pore from sequencing an unwanted DNA strand and allow a new DNA strand to be sequenced. This can also be used to sequence selected DNA strands from a pool of different DNA species.

Successful selective sequencing is in many ways beneficial given its effective flow-cell utilization and reduced cost for sequencing. Efficient and accurate real-time analysis of the current signals streamed from the nanopore sequencer is the key to successful selective sequencing. However, existing methods used in selective sequencing to accurately compare the current signal to a target reference genome are still computationally expensive. They hence would negate the true real-time sequencing capability. Selective sequencing being relatively new, potential improvements to the performance of selective sequencing in many areas are yet to be discovered. In this paper, we investigate methods to improve the accuracy and speed of selective sequencing compared to current methodologies and present a deep neural network-based method called *DeepSelectNet*.

### Related works

The very first attempt to realize the concept of selective sequencing used the Dynamic Time Warping (DTW) algorithm to align the raw current signal directly to a synthetic signal generated from the reference sequence [13]. However, due to the high computational complexity of DTW, this method could not scale beyond references that are longer than a few mega bases. Then, a method called RUBRIC was introduced by Edwards et al. [20], which performed real-time basecalling using Nanonet basecaller, followed by read alignment to the conventional nucleic acid references. RUBRIC delivered significant benefits in speed, scalability, and flexibility over the DTW [13] approach. Similar to RUBRIC, Readfish [21] is another basecalling based approach for selective sequencing. Readfish [21] utilizes the GPU-accelerated Guppy basecaller from ONT and Minimap2 mapper [22]. Readfish [21] is flexible and scales well to larger genomes when a powerful GPU is available.

UNCALLED [23] is a recent method that revitalises the raw signal alignment in the signal space for selective sequencing. UNCALLED probabilistically derive k-mers that can be presented in a signal and performs a comparison with a reference encoded using a FM-index [24]. Raw signals are first converted to events and the probability of each event is matched with a possible k-mer in a probabilistic k-mer model provided by ONT. UNCALLED [23] scaled much better than the DTW-based method [13], however, does not support large (> 1 Gbase) or highly repetitive references.

In recent years, researchers increasingly turned their direction to deep learning approaches to improve basecalling accuracy. DeepNano [25], Chiron [26], Base-cRAWler [27], Nanonet, RODAN [28], Radian [29], Albacore, Guppy, Bonito and

Dorado are such examples. According to Wick et al. [30], modern basecalling tools were able to achieve a significant improvement in basecalling with the introduction of deep learning approaches (from  $\sim 85$  to 90%). Deep Neural Networks based methods are also now being increasingly used for improved variant calling [31, 32].

Recently, deep learning-based methods for selective sequencing started to emerge. SquiggleNet [33] is the first deep learning-based tool to classify reads based on electrical signals to perform selective sequencing. SquiggleNet adopts the convolution architecture from ResNet [34], modified to perform convolution in one dimension. It claims 90% accuracy in classifying human DNA over bacterial DNA. Also, considering the computational resources, SquiggleNet outperforms alignment-based approaches with much lesser memory consumption. However, when we benchmarked SquiggleNet across several viral and bacterial datasets, we observed that on some dataset combinations, SquiggleNet performed worse than base alignment-based methods. For instance, on datasets comprised of Corona Virus & Zymo metagenome and Corona Virus & Yeast, SquiggleNet classification accuracy could not go beyond 79%, where alignment-based approaches reached 90% (discussed under Results).

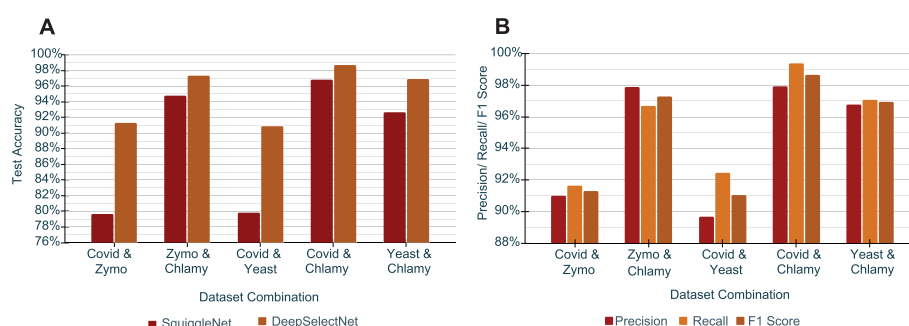
Another deep learning-based approach for selective sequencing is discussed in Danilevsky et al. [35]. The mentioned work applies deep learning models to classify mitochondrial DNA against human genomic DNA, with no reliance on labelled data to classify between the two DNA groups. However, the applicability of the mentioned method for complex inter-species classification is yet to be explored. The recent work named *baseLess* is also a deep learning approach that relies on features of salient k-mers rather than reads as a whole, explicitly designed to work with MinION [36]. The authors have reported a performance similar to SquiggleNet, which does not surpass the performance of Guppy+Minimap2. Another very recent work is RISER [37], a deep learning-based adaptive sampling method for nanopore direct RNA.

### Our contribution

The existing techniques for selective sequencing have room to improve effectiveness and efficiency. We introduce DeepSelectNet, a far superior deep learning-based method, which utilizes an improved deep neural network. Also, improved techniques of raw data preprocessing enable better feature extraction with limited training data. DeepSelectNet was able to achieve over 90% (on average  $\sim 95\%$ ) classification accuracy for five datasets that we tested.

## Results

DeepSelectNet is a deep neural network-based method capable of classifying species DNA directly using nanopore current signals with superior classification accuracy. DeepSelectNet is built on a convolutional architecture based on ResNet's residual blocks [34]. The ResNet neural network architecture has been shown to perform better in feature extraction and semantic segmentation compared to Multilayer Perceptrons (MLP), Fully Convolutional Networks (FCN), and Recurrent Neural Networks (RNN) [38]. ResNet-based neural networks are efficient and address the degradation problem by maintaining a low error rate in much deeper layers in the network [34]. Furthermore, SquiggleNet's exploratory work in different neural network architectures has



**Fig. 1** A Test Accuracy comparison of SquiggleNet vs DeepSelectNet across five dataset combinations. B DeepSelectNet's performance metrics across five dataset combinations

demonstrated that ResNet is higher in accuracy and faster in training compared to other architectures such as RNN [34]. Similar to SquiggleNet, DeepSelectNet also utilizes one-dimensional convolutional layers to perform 1D convolution over nanopore current signals in the time domain. Additionally, DeepSelectNet relies on neural net regularization to minimise model complexity thereby reducing the overfitting of data (see [Methods and Materials](#)).

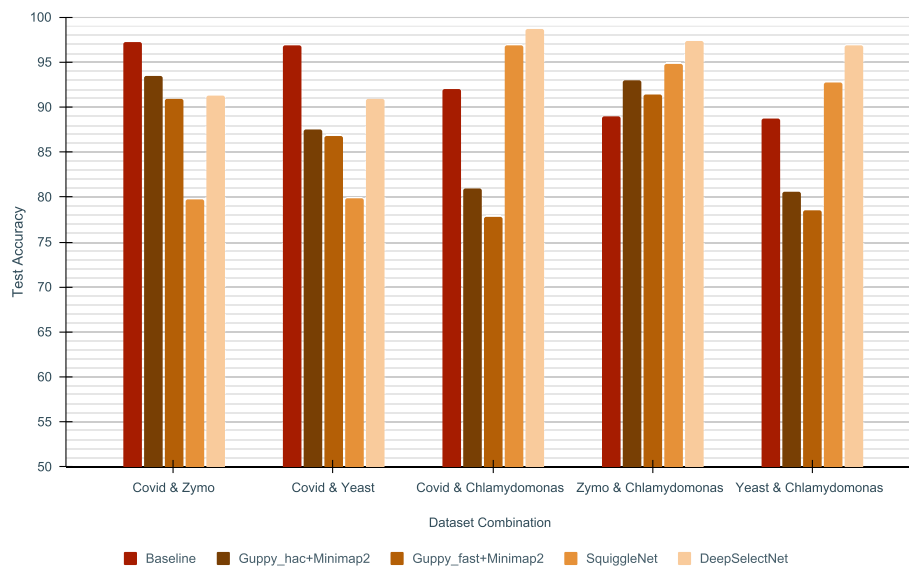
### Experiments details

Four publicly available datasets sequenced on ONT MinION/GridION, namely, SARS-CoV-2 virus (*Cov*), Zymo bacterial mixture (*Zymo*), *Saccharomyces cerevisiae* (*Yeast*), and *Chlamydomonas algae* (*Chlamy*) were used for evaluation. We prepared five combinations out of these datasets to measure the performance of DeepSelectNet: *Cov & Zymo*, *Cov & Yeast*, *Cov & Chlamy*, *Zymo & Chlamy*, and *Yeast & Chlamy* (detailed in Materials and methods). For each combination, training of DeepSelectNet was done using 20000 reads from each species (thus a balanced dataset). Before training, reads were preprocessed by trimming each read's first 1500 signal samples to eliminate the adaptor sequence (also, stall and barcodes if present; Additional file 1: Fig. S1). This 1500 value that corresponds to  $\sim 160$  bases is an overestimation for certain reads, however, it is a safe approach to eliminate unwanted portions from most of the reads [33]. Out of the remaining signal, 3000 signal samples were normalized and taken for the training. Therefore, a read to qualify for the training should have at least 4500 signal samples. The labels of the reads—from which species a given signal segment is from—were fed to the training algorithm.

### Accuracy comparison of DeepSelectNet against existing methods

DeepSelectNet outperformed SquiggleNet in terms of test accuracy (defined in Materials and methods) for all the dataset combinations (Fig. 1, Additional file 1: Table S1). The best accuracy for DeepSelectNet was observed for *Cov & Chlamy* (98.65%), while the best improvement compared to SquiggleNet was observed for *Cov & Zymo* (accuracy improvement by  $\sim 12\%$ ) as depicted in Fig. 1A. For all dataset combinations, DeepSelectNet achieved a test accuracy above 90%.

Similar to Accuracy, DeepSelectNet shows scores  $> 89\%$  on other performance metrics: precision, recall, and F1 as depicted in Fig. 1B (Additional file 1: Table S4, Additional



**Fig. 2** Test Accuracy comparison of DeepSelectNet against existing alignment methods across five dataset combinations

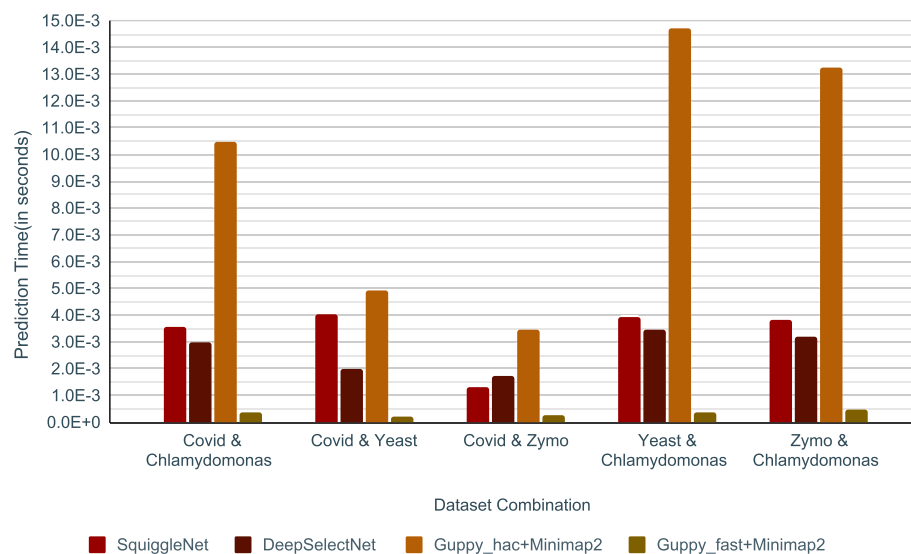
file 1: Fig. S4). Given all the datasets used for training and testing are balanced as explained earlier, being able to observe both high precision and recall demonstrates that the classification is not biased<sup>1</sup> towards a certain species in a dataset combination while also having a low variance<sup>2</sup> of predictions (Additional file 1: Fig. S4).

In addition to the comparison with SquiggleNet, DeepSelectNet's performance was then compared against: Guppy super accuracy basecalling on complete reads followed by Minimap2 [22] mapping (*baseline*); Guppy high accuracy basecalling on the first 4500 signal samples of each read<sup>3</sup> followed by Minimap2 mapping (*Guppy\_hac+Minimap2*); and, Guppy fast basecalling on the first 4500 signal samples of each read followed by Minimap2 mapping (*Guppy\_fast+Minimap2*). Accuracy results are shown in Fig. 2. The *baseline* (Fig. 2) that uses the complete read (impractical for selective sequencing) was to get an estimate of the upper margin accuracy a certain computational method could potentially achieve. *Guppy\_hac+Minimap2* mirrors a real-time scenario as in DeepSelectNet's intended use, where the whole read is not available for classification. *Guppy\_fast+Minimap2* follows the approach used in Readfish [21]. While we expected the accuracy of DeepSelectNet to be lower than the baseline, interestingly, DeepSelectNet outperformed the baseline method in three (*Zymo & Chlamy*, *Cov & Chlamy* and *Yeast & Chlamy*) out of five dataset combinations (Fig. 2, Additional file 1: Table S2). Compared to *Guppy\_hac+Minimap2* that considers the realtime scenario (unlike the baseline method that uses complete reads), DeepSelectNet was performing better in four out of five dataset combinations (except *Cov & Zymo*; Fig. 2). *Guppy\_fast+Minimap2* accuracy was somewhat in close proximity to *Guppy\_hac+Minimap2*, yet lower

<sup>1</sup> The difference between the average prediction of the model and the actual target—high bias means under-fitting.

<sup>2</sup> The variability between the actual target and predicted target—high variance means over-fitting.

<sup>3</sup> translates to ~ 300 bases when the adaptor and barcode are removed.



**Fig. 3** Inference runtime comparison of DeepSelectNet against other methods across five dataset combinations

than DeepSelectNet in all dataset combinations. Also, it was observed that *Guppy\_fast+Minimap2* has the worst performance in three out of five dataset combinations. As the predecessor of DeepSelectNet, the SquiggleNet performed as the second-best method in three out of five dataset combinations, when considering real-time scenarios.

### Runtime performance comparison

In addition to accuracy benchmarking, we also benchmarked the execution time for inference (see Fig. 3 where the per-read execution time is plotted, Additional file 1: Table S3). A setup similar to that discussed earlier (previous subsection) was used for these experiments. Among the methods experimented with, *Guppy\_fast+Minimap2* had the best execution time throughout all datasets. *Guppy\_hac+Minimap2* was the slowest (Fig. 3). This drastic difference can be accounted for by the guppy configuration of the model being used in the two methods. Compared to SquiggleNet, DeepSelectNet has the best runtime performance in four out of five datasets, except *Cov & Zymo* where SquiggleNet was faster compared to DeepSelectNet. Even though DeepSelectNet is not as fast as *Guppy\_fast+Minimap2*, DeepSelectNet's superior accuracy must be noted (Additional file 1: Fig. S7). Also, note that Guppy which is the ONT's production base-caller written in C/C++ is likely to be well-optimised for execution performance, unlike DeepSelectNet which was a prototype developed using Python. While it is not in the scope of this work to optimise DeepSelectNet for execution performance, it is logical to believe that reimplementing using C/C++ and performing optimisations could lead to improved execution time.

Additionally, we also benchmarked against Guppy super accuracy base-calling on the first 4500 signal samples of each read followed by Minimap2 mapping (*Guppy\_sup+Minimap2*; Additional file 1: Fig. S6). The prediction time for super accuracy basecalling was impractical to be used in selective sampling (more than 4X slower than *Guppy\_hac+Minimap2* that is already slow; Additional file 1: Fig. S8). However,

**Table 1** Comparison of performance of DeepSelectNet against other methods in related works

Dataset	Accuracy (%)			Inference Time (milliseconds)		
	DeepSelect Net	Squiggle Net	Guppy_fast+Minimap2	DeepSelect Net	Squiggle Net	Guppy_fast+Minimap2
Covid & Zymo	91.28	79.69	90.95	1.75	1.33	0.38
Covid & Chlamy	98.65	96.82	86.79	3.00	3.55	0.20
Covid & Yeast	90.90	79.83	77.83	1.98	4.03	0.28
Zymo & Chlamy	97.31	94.81	91.46	3.18	3.85	0.37
Yeast & Chlamy	96.93	92.69	78.49	3.45	3.95	0.45

the accuracy was better when using Guppy\_sup than Guppy\_hac (Additional file 1: Fig. S5). DeepSelectNet was still performing better in three out of five dataset combinations (except *Cov & Zymo* and *Cov & Yeast*, Additional file 1: Fig. S6). These observations denote that, DeepSelectNet has performed closely equal or better, against the benchmark methods, irrespective of the basecaller types that were used.

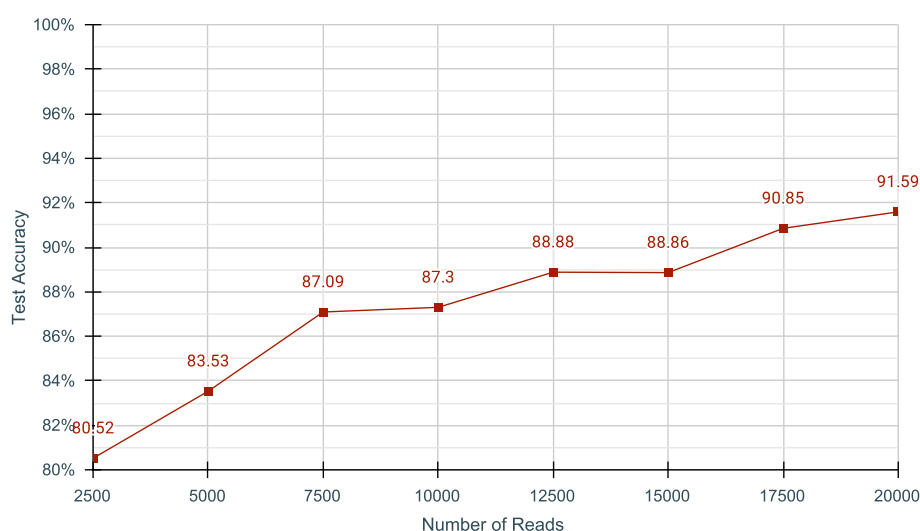
## Discussion

DeepSelectNet shows promising improvements in accuracy compared to existing deep learning-based and alignment-based approaches for selective sequencing. DeepSelectNet is a result of observing that the existing deep learning-based methods, such as SquiggleNet, had room to improve further. The model's ability to classify at higher accuracy, precision, and recall demonstrates that the classification is unbiased for a particular species. Table 1, summarises the accuracy and inference time comparison of DeepSelectNet with existing methods. As it is evident, DeepSelectNet stands out to be the best in terms of accuracy. In contrast, the *Guppy\_fast+Minimap2* (the approach used in Readfish [21]) stands to be the best in speed. However, DeepSelectNet's high accuracy could offer more benefits and the speed could be potentially improved through re-engineering (e.g., rewriting in C/C++), enabling a better realization of selective sequencing.

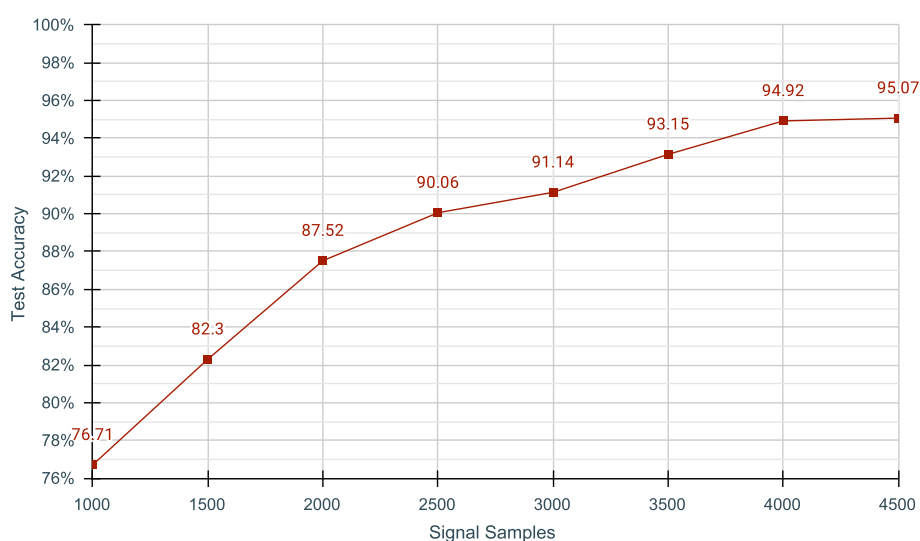
### Impact of number of reads and segment size on accuracy

We measured the model accuracy while increasing the number of reads used for training. We observed that the number of reads used for training improves the classification accuracy (Fig. 4). It is likely that the increase in accuracy is due to the increased genome coverage when the number of reads is increased (coverage is defined in Methods and Materials). When the coverage of the genome by a dataset is high, the model will have more features to learn about the species (and also the effect due to noise is diminished) and classify them better against the other species.

We also measured the model accuracy against different signal segment lengths (no. of signal samples in each read) and noticed that the accuracy improves with longer segments (Fig. 5). However, the improvement becomes less significant after  $\sim 4000$  signal samples (Fig. 5). We selected a segment size of 3000 signal samples for our experiments above ( $\sim 330$  bases), considering the time to train, real-time inference requirement and fair comparison with SquiggleNet [33] (SquiggleNet also used 3000 as the segment length). A longer signal segment means having a larger k-mer size that allows



**Fig. 4** Accuracy of DeepSelectNet against varying number of reads for Cov & Zymo dataset. The segment size (excluding the 1500 samples trimmed at the beginning) was kept constant at 3000 samples. Refer Additional file 1: Table S14 for details Genome Coverage in each case



**Fig. 5** Accuracy of DeepSelectNet against the signal segment length for Cov & Zymo dataset. The number of reads used for training was kept constant at 20,000. Note that the segment length excludes the first 1500 signal samples that are trimmed. Refer Additional file 1: Table S13 for details Genome Coverage in each case

distinguishing species better, thereby the model may classify better with longer segments. Having a longer segment increases the genome coverage, which also may improve the model performance as discussed before.

### DeepSelectNet with human datasets

Inter-species experiments we performed with *Human & Zymo* and *Human & Covid*, produced 92.26% and 96.27% test accuracies, respectively (Additional file 1: Table S9). Human data were from the publicly available NA12878 reference Genome [39]. For these experiments, 20,000 reads from each species were used (relates to 0.008x, 400x,

and 0.4× genome coverage for human, covid and zymo respectively, Materials and methods, Additional file 1: Table S12). Interestingly, even without seeing the complete genome (coverage < 0.008× for human), the model could classify accurately. This could be because, even though the coverage is less, still the features are sufficiently different (an analogy would be k-mers distribution between humans and bacteria/viruses being different). However, it also raised a concern if the model is picking some other difference amongst the datasets than the species (for instance, differences in library preparation method and/or minor differences in the flowcells in the publicly available datasets sequenced by different labs at different times). To eliminate this concern, we simulated synthetic nanopore raw signal data for the two species Yeast and Chlamydomonas using *Squigulator* [40] that relies on a pore-model indicating the expected current level for each k-mer. When this simulated data was used for training and testing, DeepSelectNet could achieve a test accuracy of 97.10% (Additional file 1: Table S11). Therefore, this experiment demonstrates that DeepSelectNet actually classifies based on features of species, rather than being misled by any other non-species related difference. To support this further, an experiment against two species present in Zymo dataset, the *Bacillus subtilis* & *Saccharomyces cerevisiae* was performed. The experiment achieved 82.51% of test accuracy (Additional file 1: Table S15), which is sufficient to demonstrate that DeepSelectNet can distinguish different species in the same sample. Note that we relied on Guppy+Minimap2 results to generate the truth set for DeepSelectNet (as there is no other way to isolate the reads from the zymo mix), which could have negatively affected the accuracy.

Though the DeepSelectNet performed great for inter-species classification as explained throughout the paper (accuracy ~ 95% on average), its performance was not as great for intra-species classification (that is, classifying reads based on a genomic region of the same species). For example, when chromosome 21 and chromosome 22 from the human genome (publicly available NA12878 data available under [41]) were used to train DeepSelectNet (chr21 as the positive dataset and chr22 as the negative dataset), the training accuracy was below 60% in all cross folds. Increasing the number of reads for each chromosome from 20,000 to 100,000 (signal segment length of 3000 samples), improved the training accuracy by a small margin up to 64%. With 35,000 reads from each chromosome at a signal segment length of 10,000 (~ 1100 bases), the training accuracy improved to 73%. It is possible that DeepSelectNet is reaching its performance limitations to learn fine features among different chromosomes or different regions of a species. Unfortunately, our computational infrastructure restricted us from exploring further on larger datasets than this.

## Materials and methods

### DeepSelectNet model architecture

The first layer in DeepSelectNet is a one-dimensional Convolutional Neural Network (CNN) with 20 channels. The rest of the residual blocks (Similar to ResNet) are implemented additionally with dropout layers [42] to effectively regularize any over-fitting effects that the layer complexity might introduce. This helps to mitigate the effect of the model picking up on statistical noise in the training data, which could result in poor performance when the model is evaluated on unseen data. During training, 10% layer

**Table 2** Datasets and their sources

Species	Dataset	Source
SARS-CoV-2	Cov	<a href="https://community.artic.network/t/links-to-raw-fast5-fastq-data-for-artic-protocol/17">https://community.artic.network/t/links-to-raw-fast5-fastq-data-for-artic-protocol/17</a>
Zymo Metagenome	Zymo	<a href="https://github.com/LomanLab/mockcommunity">https://github.com/LomanLab/mockcommunity</a>
Chlamydomonas	Chlamy	<a href="https://sra-download.ncbi.nlm.nih.gov/traces/era20/ERZ/003237/ERR3237140/Chlamydomonas_0.tar.gz">https://sra-download.ncbi.nlm.nih.gov/traces/era20/ERZ/003237/ERR3237140/Chlamydomonas_0.tar.gz</a>
Saccharomyces cerevisiae	Yeast	<a href="https://www.ncbi.nlm.nih.gov/bioproject/PRJNA510813">https://www.ncbi.nlm.nih.gov/bioproject/PRJNA510813</a>

outputs are randomly ignored so that the layer will be treated like a layer with a different number of nodes and edges than the layer before. Therefore, each update to a layer during training is performed differently in comparison to the updated layer. The four residual layers, which include two bottleneck blocks, are stacked together following a mean pooling layer and lastly, a fully-connected layer activated with a sigmoid function that flattens the output for classification.

### DeepSelectNet implementation

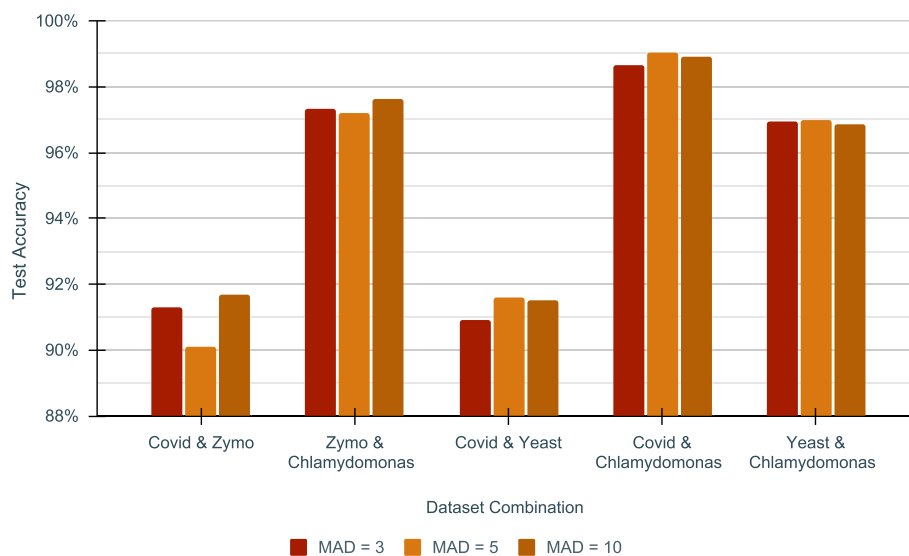
DeepSelectNet was developed in Python programming language using the Keras framework.<sup>4</sup> Core functionalities of DeepSelectNet are distributed among three scripts:

- *preprocessor.py* that pre-processes datasets prior to training (read filtering, signal trimming, signal segmentation, signal normalization and splitting to batches).
- *trainer.py* that trains the pre-processed data (change hyperparameters of the model including classifier network, loss function, train to validation split ratio, number of cross folds, epochs, data batch size, etc.).
- *inference.py* that inferences test data using a trained model (data preprocessing similar to *preprocessor.py* is done in this script before the inference).

### Data Pre-processing

Four publicly available datasets sequenced on ONT MinION/GridION were used for the experiments (Table 2). These datasets contained raw signal data in single-FAST5 format (one file per each read), which we converted to SLOW5 format using *slow5tools* [41] to enable convenient and efficient file manipulation. Then, 40,000 reads containing at least 4500 signal samples were extracted from each dataset. Using these extracted reads, five dataset combinations were created, namely *Cov* & *Zymo*, *Cov* & *Chlamy*, *Cov* & *Yeast*, *Yeast* & *Chlamy* and *Zymo* & *Chlamy*. Each dataset combination was partitioned equally for training and testing. For instance, the *Cov* & *Yeast* dataset combination which contains 40,000 from *Cov* and 40,000 from *Yeast* is partitioned such that 20,000 *Cov* reads and 20,000 *Yeast* reads are for training, and the rest for testing. Thus, the created datasets were balanced datasets where a species' contribution to each dataset combination was equal. Note that this count of 20000 reads was a value that we empirically

<sup>4</sup> Keras is a high-level API built up on Tensorflow to develop deep neural networks.



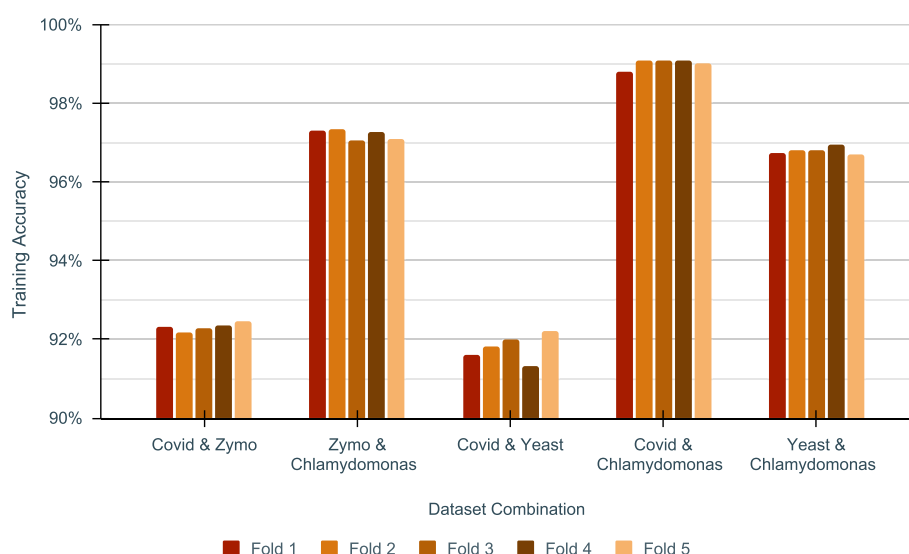
**Fig. 6** Impact of Median Absolute Deviations(MAD) on DeepSelectNet's Test Accuracy

determined to adequately provide genome-wide coverage for these dataset combinations (See the [Discussion](#) for more information).

Each read from the dataset is first trimmed by removing the initial 1500 signal samples. As discussed earlier, trimming is essential to get rid of the read adaptor (and barcode if exists, Additional file 1: Fig. S1). Then, the raw signal samples (16-bit integers) are converted to pico-amperes using the equation:  $pA\_signal = (\frac{range}{digitization}) \times (raw\_signal + offset)$ . Next, four random segments (where each segment is 3000 signal samples) are obtained from this converted pico-ampere signal.<sup>5</sup> Taking multiple signal segments from a single long read provides more feature space than taking a single segment and directly affects on model performance. This technique helped DeepSelectNet to increase its accuracy to a great extent (Additional file 1: Table S10). Value four as the number of segments was derived empirically, considering the model accuracy, the average length of the reads and the time to train.

Each such segment is then normalized with modified version of z-score that uses median absolute deviation (MAD):  $Modified\_Z\_Score = \frac{0.6745 \times (X_i - \tilde{X})}{MAD}$ ; where  $MAD = median(\|X_i - \tilde{X}\|)$  and  $\tilde{X} = median(X)$ . The median absolute deviation used in the modified z-score is tolerant to extreme outliers (Additional file 1: Fig. S2) present in the data. The threshold used for MAD in data preprocessing had an effect on model training (Fig. 6, Additional file 1: Table S7). The MAD threshold affects the number of outliers being filtered, producing a differently performing model. Therefore, the better the outliers are filtered, the better the model trains. The best MAD threshold could depend on many factors such as the species, sequencing device and sample purity and thus we determined this threshold empirically (Additional file 1: Fig. S3). For this, We

<sup>5</sup> there could be overlaps when randomly taking these segments; if the remaining signal length is only 3000 samples, all the four segments will be identical.



**Fig. 7** Training Accuracy of DeepSelectNet's across five cross folds

conducted experiments with  $MAD = 3, 5$ , and  $10$  as the thresholds and picked the value that produced the best accuracy for a given dataset combination (Fig. 6).

### Training

In a given dataset combination, the training data partition (explained above) was again partitioned in 7:3 ratio, where 70% of training data was used for training and the rest for validation. As an example, *Cov & Yeast* dataset combination containing 20,000 training reads from *Cov* and 20,000 training reads from *Yeast* is partitioned such that 14,000 *Cov* reads and 14,000 *Yeast* reads are for training, and the rest for validation.

Additionally, cross-fold validation was used to generalise training to ensure a minimally biased model. The training accuracy of trained models in each of the 5 folds lies in close proximity (Fig. 7, Additional file 1: Table S8), demonstrating that training has generalised. The training data was fed to the model in batches of 1000 reads at a time. The full dataset was trained for a maximum of 200 epochs in one cross-fold. Learning Rate (LR) factor for the training was 0.5. The LR patience factor was 20 epochs and was monitored against the binary accuracy metric. To avoid over-fitting, an Early Stopping Monitor was used with an Early Stopping Patience of 30 epochs.

### Evaluation

The best performing model of the five training cross folds was selected for the evaluation. All evaluations were done with previously unseen test data created in the preprocessing step. The Test data is also first preprocessed: the first 1500 signal samples in each read are trimmed off; the next 3000 signal samples are taken (note that only the first segment that represents the beginning of the read is taken for testing; we do not take 4 segments as in training); converted to pA and normalized using the modified z-score as done for training.

The inference was performed in batches of 1000 reads to ensure higher GPU occupancy and also indicates a scenario where reads are being sequenced in parallel on a nanopore sequencer with 1000 channels. The model training and the accuracy-related benchmarks were conducted on a system powered by an Intel Xeon CPU E5-1630 v3 @ 3.70GHz CPU, 32 GB of RAM, 2TB of HDD storage, and an Nvidia Quadro K40 GPU. As this was a shared workstation unsuitable for reliable timing measurements, runtime benchmarks were performed on another dedicated workstation powered by an Intel Xeon Silver 4114 CPU @ 2.20GHz CPU, 384 GB RAM, 6TB NVME SSD storage and a Tesla V100-16 G GPU.

#### Modifications to SquiggleNet for comparison against DeepSelectNet

We modified SquiggleNet to accept SLOW5 file format [41] as an input (SquiggleNet only accepted single-FAST5 files and manipulation of those files is inefficient and inconvenient). This modification does not affect the model classification accuracy as the underlying raw signal data is identical despite the file format used. The modified source code of SquiggleNet is available in the link provided under Availability of data and materials.

#### Baseline, Guppy\_hac+Minimap2 and Guppy\_fast+Minimap2

The *baseline* approach uses complete reads whereas *Guppy\_hac+Minimap2* and *Guppy\_fast+Minimap2* uses only the first 4500 signal samples in a read (see Results). Guppy version 6.1.3 and Minimap2 version 2.20 were used for the experiments. For *baseline* and *Guppy\_hac+Minimap2*, the high-accuracy basecalling mode *dna\_r9.4.1\_450bps\_hac* was used for Guppy and Minimap2 was executed with base alignment enables (with -c option). For *Guppy\_fast+Minimap2*, fast basecalling mode was used with Guppy and Minimap2 was executed without base alignment to mirror the approach used in Readfish [21].

#### Genome coverage computation

Genome Coverage denotes the ratio between the number of bases appearing in the given dataset to the actual number of bases in its reference genome. The formula we used to compute the genome coverage seen by the model during training for a given dataset is given below.

$$\text{Genome Coverage} = \frac{\text{signal segment length} \times \text{No. of reads}}{\text{samples per base} \times \text{reference genome length}}$$

The signal segment length is the number of samples in each signal segment used for training (excluding the samples trimmed at the beginning of the read). Samples per base refer to the number of signal samples that correspond to a single base, which is determined by the translocation speed of the DNA strand through the pore and the sampling rate of data acquisition. For DNA on R9.4.1, 450 bases/s at 4000 Hz gives ~ 8–10 Samples per base.

### Accuracy metrics

Accuracy metrics and related terms used for evaluating the classification performance of the models in this work are defined below.

An outcome where the model predicts a read as positive when the read is actually from the targeted (positive) dataset/species is considered a *True Positive (TP)*. An outcome where the model predicts a read as negative when the read is actually from the non-targeted (negative) dataset/species is considered a *True Negative (TN)*. An outcome where the model predicts a read as positive when the read is actually from the non-targeted (negative) dataset/species is considered a *False Positive (FP)*. An outcome where the model predicts read as negative when the read is actually from the targeted (positive) dataset/species is considered a *False Negative (FN)*. Accuracy that denotes the overall accuracy of the model with correct predictions over total predictions is computed as  $\frac{TP+TN}{TP+TN+FP+FN}$ ; precision is computed as  $\frac{TP}{TP+FP}$ ; Recall is computed as  $\frac{TP}{TP+FN}$ ; and F1 score  $\frac{2 \times (Precision \times Recall)}{Precision + Recall}$ . However, in experiments from baseline method, Guppy\_hac+Minimap2 and Guppy\_fast+Minimap2, the unmapped reads accounted neither as false positives nor false negatives. Therefore in such cases, Precision, Recall and F1 Score are approximations with available prediction statistics.

### Conclusion

Nanopore selective sequencing is an emerging area of genome sequencing that is gaining popularity due to its efficiency and low cost. There have been several attempts to realize selective sequencing with better accuracy and speed. In this paper, we presented DeepSelectNet, a deep learning-based approach for species classification using selective sequencing that outperforms existing methods. Amongst five inter-species datasets evaluated, DeepSelectNet achieved > 90% accuracy (on average ~ 95%) for all datasets. DeepSelectNet was bench-marked against existing methods (both deep-learning and non-deep-learning methods) and produced exceedingly top performances across the vast majority of the datasets. In some cases, DeepSelectNet on partial reads (4500 signal samples at the beginning of each read) outperformed the alignment-based approach (high accuracy basecalling + alignment) done on complete reads. DeepSelectNet also performed well in classifying human DNA from bacteria/viral data (> 90% accuracy). However, DeepSelectNet could only achieve < 70% for intra-species classification such as classifying based on different regions of the same species. Thus, the architecture of DeepSelectNet is more suitable for inter-species classification. In terms of execution time, DeepSelectNet outperformed the existing deep-learning-based method. However, DeepSelectNet being a Python-based prototype was several times slower than the C/C++-based Guppy (fast basecalling) + Minimap2 combination. Re-implementation of DeepSelectNet in C/C++ followed by performance tuning would potentially achieve comparable runtime performance.

### Supplementary Information

The online version contains supplementary material available at <https://doi.org/10.1186/s12859-023-05151-0>.

**Additional file 1.** Supplementary Information.

# Acknowledgements

Not applicable

# Author contributions

HG and RR conceived the study. AS designed, developed and optimised the methods. AS, HG and DH planned the experiments. AS and HG conducted the experiments and benchmarks. AS and HG wrote the manuscript. DH and RR revised the manuscript. All authors read and approved the final manuscript.

# Funding

Not applicable.

# Availability of data and materials

The source code for DeepSelectNet is available on Github at <https://github.com/AnjanaSenanayake/DeepSelectNet>. The source code for modified SquiggleNet is available on Github at <https://github.com/AnjanaSenanayake/SquiggleNet/tree/slow5-support>. All instructions for using the tool can be found in Additional file 1: Note 1. Datasets used are publicly available datasets and their links can be found in Table 2. Additionally, a curated dataset used for DeepSelectNet experiments can be found at <https://doi.org/10.5281/zenodo.7111366>.

# Declarations

## Ethics approval and consent to participate

Not applicable.

## Consent for publication

Not applicable.

## Competing interests

H.G. has received travel and accommodation expenses to speak at Oxford Nanopore Technologies conferences. The authors declare that they have no other competing financial or nonfinancial interests.

Received: 24 October 2022 Accepted: 17 January 2023

Published online: 28 January 2023

# References

1. Jain M, Olsen HE, Paten B, Akeson M. The oxford nanopore minion: delivery of nanopore sequencing to the genomics community. *Genome Biol.* 2016;17(1):239–239.
2. Lu H, Giordano F, Ning Z. Oxford nanopore minion sequencing and genome assembly. *Genomics Proteomics Bioinform.* 2016;14(5):265–279. <https://doi.org/10.1016/j.gpb.2016.05.004>. SI: Big Data and Precision Medicine
3. Mikheyev AS, Tin MM. A first look at the oxford nanopore minion sequencer. *Mol Ecol Resour.* 2014;14(6):1097–102.
4. Castro-Wallace SL, Chiu CY, John KK, Stahl SE, Rubins KH, McIntyre AB, Dworkin JP, Lupisella ML, Smith DJ, Botkin DJ, et al. Nanopore DNA sequencing and genome assembly on the international space station. *Sci Rep.* 2017;7(1):1–12.
5. Hoenen T, Groseth A, Rosenke K, Fischer RJ, Hoenen A, Judson SD, Martellaro C, Falzarano D, Marzi A, Squires RB, et al. Nanopore sequencing as a rapidly deployable ebola outbreak tool. *Emerg Infect Dis.* 2016;22(2):331.
6. Xu F, Ge C, Luo H, Li S, Wiedmann M, Deng X, Zhang G, Stevenson A, Baker RC, Tang S. Evaluation of real-time nanopore sequencing for salmonella serotype prediction. *Food Microbiol.* 2020;89: 103452.
7. Gigante CM, Yale G, Condori RE, Costa NC, Long NV, Minh PQ, Chuong VD, Tho ND, Thanh NT, Thin NX, et al. Portable rabies virus sequencing in canine rabies endemic countries using the oxford nanopore minion. *Viruses.* 2020;12(11):1255.
8. Jain M, Koren S, Miga KH, Quick J, Rand AC, Sasani TA, Tyson JR, Beggs AD, Dillthey AT, Fiddes IT, et al. Nanopore sequencing and assembly of a human genome with ultra-long reads. *Nat Biotechnol.* 2018;36(4):338–45.
9. Miga KH, Koren S, Rhie A, Vollger MR, Gershman A, Bzikadze A, Brooks S, Howe E, Porubsky D, Logsdon GA, et al. Telomere-to-telomere assembly of a complete human x chromosome. *Nature.* 2020;585(7823):79–84.
10. Wang S, Song Q, Li S, Hu Z, Dong G, Song C, Huang H, Liu Y. Assembly of a complete mitogenome of chrysanthemum nankingense using oxford nanopore long reads and the diversity and evolution of asteraceae mitogenomes. *Genes.* 2018;9(11):547.
11. Frei D, Veekman E, Grogg D, Stoffel-Studer I, Morishima A, Shimizu-Inatsugi R, Yates S, Shimizu KK, Frey JE, Studer B, et al. Ultralong oxford nanopore reads enable the development of a reference-grade perennial ryegrass genome assembly. *Genome Biol Evol.* 2021;13(8):159.
12. Wang Y, Zhao Y, Bollas A, Wang Y, Au KF. Nanopore sequencing technology, bioinformatics and applications. *Nat Biotechnol.* 2021;39(11):1348–65.
13. Loose M, Malla SM. Real-time selective sequencing using nanopore technology. *Nat Methods* 2016;751–754.
14. Ulrich J-U, Lutfi A, Rutzen K, Renard BY. Readbouncer: Precise and scalable adaptive sampling for nanopore sequencing. *bioRxiv.* 2022.
15. Stevanovski I, Chintalapathi SR, Gamaarachchi H, Ferguson JM, Pineda SS, Scriba CK, Tchan M, Fung V, Ng K, Cortese A, et al. Comprehensive genetic diagnosis of tandem repeat expansion disorders with programmable targeted nanopore sequencing. *Sci Adv.* 2022;8(9):5386.
16. Martin S, Heavens D, Lan Y, Horsfield S, Clark MD, Leggett RM. Nanopore adaptive sampling: a tool for enrichment of low abundance species in metagenomic samples. *Genome Biol.* 2022;23(1):1–27.

17. Lin Y, Dai Y, Liu Y, Ren Z, Guo H, Li Z, Li J, Wang K, Yang L, Zhang S, et al. Rapid PCR-based nanopore adaptive sequencing improves sensitivity and timeliness of viral clinical detection and genome surveillance. *Front Microbiol.* 2022;13.
18. Mariya T, Kato T, Sugimoto T, Miyai S, Inagaki H, Ohye T, Sugihara E, Muramatsu Y, Mizuno S, Kurahashi H. Target enrichment long-read sequencing with adaptive sampling can determine the structure of the small supernumerary marker chromosomes. *J Hum Genet.* 2022;67(6):363–8.
19. Marquet M, Zöllkau J, Pastuschek J, Viehweger A, Schleußner E, Makarewicz O, Pletz MW, Ehrlich R, Brandt C. Evaluation of microbiome enrichment and host DNA depletion in human vaginal samples using oxford nanopore's adaptive sequencing. *Sci Rep.* 2022;12(1):1–10.
20. Edwards H, Krishnakumar R, Sinha A, Bird S, Patel K, Bartsch M. Real-time selective sequencing with rubric: Read until with basecall and reference-informed criteria. *Sci Rep.* 2019. <https://doi.org/10.1038/s41598-019-47857-3>.
21. Payne A. Readfish enables targeted nanopore sequencing of gigabase-sized genomes. *Nat Biotechnol.* 2021. <https://doi.org/10.1038/s41587-020-00746-x>.
22. Li H. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics.* 2018;34(18):3094–100. <https://doi.org/10.1093/bioinformatics/bty191>.
23. Kovaka S, Fan Y, Ni B, Timp W, Schatz MC. Targeted nanopore sequencing by real-time mapping of raw electrical signal with uncalled. *Nat Biotechnol.* 2021;39(4):431–41.
24. Cheng H, Wu M, Xu Y. FMtree: a fast locating algorithm of FM-indexes for genomic data. *Bioinformatics.* 2017;34(3):416–24. <https://doi.org/10.1093/bioinformatics/btx596>.
25. Boža V, Brejová B, Vinař T. Deepnano: Deep recurrent neural networks for base calling in minion nanopore reads. *PLoS ONE.* 2017;12(6):1–13. <https://doi.org/10.1371/journal.pone.0178751>.
26. Teng H, Cao MD, Hall MB, Duarte T, Wang S, Coin LJ. Chiron: translating nanopore raw signal directly into nucleotide sequence using deep learning. *GigaScience.* 2018;7(5):037.
27. Stoiber M, Brown J. Basecrawler: Streaming nanopore basecalling directly from raw signal. *BioRxiv.* 133058. 2017.
28. Neumann D, Reddy AS, Ben-Hur A. Rodan: a fully convolutional architecture for basecalling nanopore RNA sequencing data. *BMC Bioinform.* 2022;23(1):1–9.
29. Sneddon A, Shirokikh NE, Eyra E. Language-informed basecalling architecture for nanopore direct rna sequencing. *bioRxiv.* 2022.
30. Wick RR, Judd LM, Holt KE. Performance of neural network basecalling tools for oxford nanopore sequencing. *Genome Biol.* 2019;20(1):1–10.
31. Luo R, Wong C-L, Wong Y-S, Tang C-I, Liu C-M, Leung C-M, Lam T-W. Exploring the limit of using a deep neural network on pileup data for germline variant calling. *Nat Mach Intell.* 2020;2(4):220–7.
32. Su J, Zheng Z, Ahmed SS, Lam T-W, Luo R. Clair3-trio: high-performance nanopore long-read variant calling in family trios with trio-to-trio deep neural networks. *bioRxiv.* 2022.
33. Bao Y, et al. SquiggleNet: real-time, direct classification of nanopore signals. *Genome Biol.* 2021. <https://doi.org/10.1186/s13059-021-02511-y>.
34. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. *CoRR arXiv:1512.03385* 2015.
35. Danilevsky A, Polsky AL, Shomron N. Adaptive sequencing using nanopores and deep learning of mitochondrial DNA. *Brief. Bioinform.* 2022;23(4):bbac251. <https://doi.org/10.1093/bib/bbac251>.
36. Noordijk B, Nijland R, Carrion VJ, Raaijmakers JM, de Ridder D, de Lannoy C. baseless: lightweight detection of sequences in raw minion data. *bioRxiv.* 2022. <https://doi.org/10.1101/2022.07.10.499286>.
37. Sneddon A, Ravindran A, Hein N, Shirokikh NE, Eyra E. Real-time biochemical-free targeted sequencing of RNA species with riser. *bioRxiv.* 2022.
38. Elharrouss O, Akbari Y, Almaadeed N, Al-Maadeed SA. Backbones-review: Feature extraction networks for deep learning and deep reinforcement learning approaches. *ArXiv arXiv:2206.08016* 2022.
39. Whole Human Genome Sequencing Project. <https://github.com/nanopore-wgs-consortium/NA12878/blob/master/Genome.md#rel6-data>
40. Gamaarachchi H. Squigulator. <https://github.com/hasindu2008/squigulator>
41. Gamaarachchi H, Samarakoon H, Jenner SP, Ferguson JM, Amos TG, Hammond JM, Saadat H, Smith MA, Parameswaran S, Deveson IW. Fast nanopore sequencing data analysis with slow5. *Nat Biotechnol.* 2022;1–4.
42. Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving neural networks by preventing co-adaptation of feature detectors. 2012. <https://doi.org/10.48550/ARXIV.1207.0580>

# Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.