UTM
UNIVERSITI TEKNOLOGI MALAYSIA
INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING
ISSN 2180-4370
Journal Homepage : https://ijic.utm.my/

Detection of Potential Viral Sequence from Next Generation Sequencing Data Using Convolutional Neural Network

Xin Ying Lim, Jia Yee Lim, Weng Howe Chan* & Hui Wen Nies
Faculty of Computing
Universiti Teknologi Malaysia
81310 UTM Johor Bahru, Johor, Malaysia
Email: cwenghowe@utm.my

Submitted: 8/9/2022. Revised edition: 28/2/2023. Accepted: 6/3/2023. Published online: 30/5/2023
DOI: https://doi.org/10.11113/ijic.v13n1.382

***Abstract*—Next Generation Sequencing (NGS) is a modern sequencing technology that can determine the sequences of RNA and DNA faster and at lower cost. The availability of NGS data has sparked numerous efforts in bioinformatics, especially in the study of genetic variation and viral sequence detection. Viral sequence detection has been one of the important processes in studying virus-induced diseases. Common methods in detecting viral sequences involve alignment of the sequence with existing databases, which remains limited as these databases might be incomplete and difficult to detect highly divergent viruses. Thus, machine learning and deep learning have been used in this regard, to unveil the patterns that distinguish viral sequences through learning from the NGS data. This study focuses on viral sequence detection using convolutional neural network (CNN). This study intended to investigate how CNN model can be used for analysis of NGS data and develop a CNN model for detecting potential viral sequences from NGS data. The CNN architecture used for this study is based on an existing design that divided into two branches namely pattern and frequency branch that cater for extracting different aspects of information from the data and lastly combined into a full model. This study further implemented slightly modified architecture that includes additional convolution layer and pooling layer. Then, parameter tuning is implemented to identify near optimal parameters for the CNN to elucidate the performance impact. The evaluation of the optimized CNN model is done using a dataset with 18,445 DNA sequences. The results show that the CNN model in this study achieved a better performance compared with existing in terms of area under receiver operating characteristics curve (AUROC) for full model (+0.1434).**

***Keywords*—Next generation sequencing, viral sequence detection, convolutional neural network, bioinformatics**

I. INTRODUCTION

Next Generation Sequencing (NGS) is a high-throughput sequencing that can sequence deoxynucleic acid (DNA) and ribonucleic acid (RNA) more quickly and cost effective than conventional sequencing methods such as Sanger Sequencing. Analysis of the entire human genome can be done in a single sequencing experiment and in a short time using NGS technology [1]. Apart from that, the determining sequence of DNA and RNA using NGS technology helps in the study of genetic variation [2-4] and viral sequence detection [5]. The entire assemble of viruses in and on the human body is known as human virome. Some viruses may cause disease in the human body. In fact, a great number of different viruses can be found in the biospecimen of humans. Nevertheless, just some human viruses have been found, and there are numerous other viruses that have not yet been reported [6]. The analysis and classification of the viral sequences remain a big challenge for researchers.

Commonly, to detect the potential viral sequences in human biospecimens, the traditional alignment-based classification such as Basic Local Alignment Search Tool (BLAST) is used. In BLAST, the sequences are compared to discovered genomes in the databases and estimate the similarity that the sequences shared. However, the limitation of BLAST is that the public databases are insufficient [7], human NGS data might contain many extremely divergent viruses that do not have homologs to the curated genomes in the databases. As a result, BLAST usually categorized these sequences as “unknown” [8].

HMMER3 [9] is another commonly used technique for detecting viral sequences in human biospecimen. This algorithm implements profile Hidden Markov Models (pHMM) and vFam, which is a HMMER3 database that is built from every viral protein that exists in RefSeq database. This technique is more likely to be used to detect distant homologs as the sequences are differentiated to the viral families. However, this method uses a reference database which resulted in similar limitation of BLAST when analyzing highly divergent viruses.

Deep learning and machine learning have been used to unveil the patterns that distinguish viral sequences through learning from the NGS data. There are various techniques that have been used such as Random Forest [10, 11], Artificial Neural Network [12, 13] and especially Convolutional Neural Network (CNN) has been shown with good performance in several existing works [5, 6, 14, 15]. For example, Ren and colleagues developed a model implementing CNN that takes DNA sequences to identify viruses from prokaryotes [14]. While Tampuu and colleagues introduced a model that implement CNN to detect viruses in human biospecimen [6]. However, the performance of CNN could be affected by how it's designed and the choice of hyperparameters setting. Thus, in this study, CNN is used for further investigation in its performance in detection of potential viruses from human NGS data by implementing different encoding methods, hyperparameter settings and the architecture design.

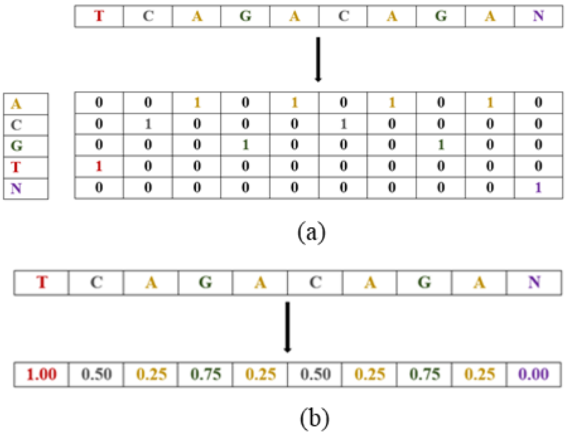
II. Materials and Methods

A. Dataset

The dataset used in this study is extracted from metagenomics assembled contigs collected from 19 different experiments [6]. Sequences are given in letters "ATCG…" with the length of 300 and the labels are 1 or 0 to represent virus or non-virus, respectively. There are 18,445 DNA sequences in the dataset used in the proposed solution, with 17,713 (96.03%) classed as non-viral sequences and 732 (3.97%) classified as viral sequences. Following that, the dataset is divided into three sections: training (80%, 14756 sequences), testing (10%, 1845 sequences), and validation (10%, 1844 sequences).

B. Preprocessing

The previous work by Tampuu and colleagues [6] have shown that utilizing the one-hot encoding method to transform DNA sequences into matrix format of numerical numbers can get good results in the CNN model. However, other studies employed ordinal sequencing, substituting one-hot encoding due to the high dimensionality of the one-hot encoded input data [17]. In this study, both preprocessing methods are used to encode the DNA sequences to construct different CNN models and the performance of both models is compared. Basically, in one-hot encoding, the nucleotide bases are transformed into a matrix of 0s and 1s with the size of 300 (size of the sample) x 5 (five unique nucleotide base) for each sample, it converts nucleotide A as [1, 0, 0, 0, 0], nucleotide C as [0, 1, 0, 0, 0], nucleotide G as [0, 0, 1, 0, 0], nucleotide T as [0, 0, 0, 1, 0] and N as [0, 0, 0, 0, 1], resulted a final matrix of 18445 x 300 x 5 at the end. The letter 'N' basically represents detected nucleotide that wasn't A, T, C, and G, or also known as noise in this case. While ordinal encoding transforms nucleotide bases of a sample into ordered numerical values where nucleotides A, C, G, T, N are represented as 0.25, 0.5, 0.75, 1.0 and 0, respectively. This resulted in a final matrix with the size of 18445 x 300 x 1. Here, during the ordinal encoding, 'N' is represented as 0 since it represents noise (any bases other than A, T, G, C) so that in the encoded outcome, 'N' will not play any role in the analysis. Fig. 1 illustrates the example of both preprocessing methods while Fig. 2 depicts the dimensions of the input data that were applied to both preprocessing methods.



T	C	A	G	A	C	A	G	A	N

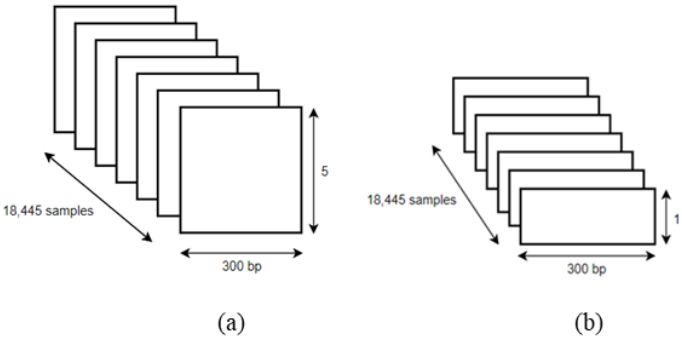
A	0	0	1	0	1	0	1	0	1	0
C	0	1	0	0	0	1	0	0	0	0
G	0	0	0	1	0	0	0	1	0	0
T	1	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	1

(a)

T	C	A	G	A	C	A	G	A	N
1.00	0.50	0.25	0.75	0.25	0.50	0.25	0.75	0.25	0.00

(b)

Fig. 1. Preprocessing method: (a) One-hot encoding; (b) Ordinal encoding



(a) (b)

Fig. 2. The dimension of the input data that applied: (a) One-hot encoding; (b) Ordinal encoding

C. CNN Architecture

The CNN architecture in this study is based on the existing ViraMiner by Tampuu and colleagues [6]. It comprises two branches of models namely pattern branch and the frequency branch. The pattern branch model intends to learn and return the degree to which certain DNA sequence patterns were utmost matched across the entire DNA sequence. Therefore, in this branch, the convolution layer is followed by global max pooling which indicates that only one utmost activation value can be passed on to the next layer from each filter. The downside of this branch is that other data, for instance how frequently an excellent match is discovered, is not covered.

As for the frequency branch, it is used to solve the pattern branch's shortcoming. As its name implies, the frequency branch returns the frequency of the pattern. Thus, the frequency branch implements the global average pooling after the convolution layer. Although the maximal activation information is lost, information regarding the frequency is gained in this branch because if just a few good matches are found, the average cannot be high.

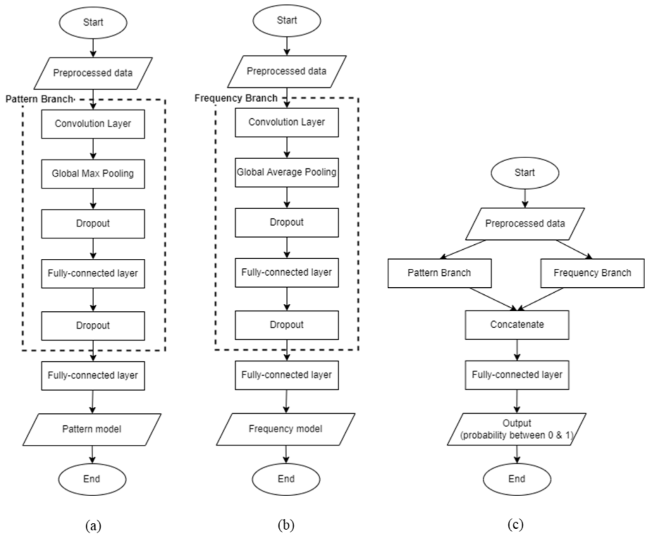


Fig. 3. Workflow of (a) Pattern model, (b) Frequency model and (c) Full model

Both pattern and frequency branches provide different types of information that are merged into the full model where the trained network from both branches are fed into the concatenate layer in the Keras library that merges as the full model. The output node with the sigmoid activation function is then added. The activation function converts the weighted sum of inputs into a probability that ranges between 0 and 1 (Eq.1).

$$\sigma(x) = \frac{1}{1+e^{-1}} \qquad (1)$$

The workflow is illustrated in Fig. 3. For model training, both pattern and frequency branches are trained individually, which implies that each branch generates its own model, which is then integrated to form a full model. Finally, there are three models that can provide the performance report separately (pattern model, frequency model, and whole model).

D. Hyperparameter Tuning

Hyperparameter tuning is used to determine which combination of hyperparameters delivers the greatest result for the pattern and frequency model. This study focus on four hyperparameters namely filter size, layer size, droupout rate and learning rate. Filter size and layer size impacts on how much information is extracted during the convolution. Dropout rate is crucial for avoiding overfitting while learning rate would determine the speed of the model adapts to the problem. The HParams Dashboard in Google Colab is used to tune the hyperparameters. Each hyperparameter combination from the given range undergoes model training for 5 epochs during hyperparameter tuning. The finest model is then picked according to the highest validation AUROC. The hyperparameters that involve in the tuning are filter size, layer size, dropout rate and learning rate. The range of each parameter used is as follows:

- Filter size: 6-14 (Pattern branch); 8-16 (Frequency branch)
- Layer size: 1000, 1200, 1500
- Dropout rate: 0.1, 0.5
- Learning rate: 0.01, 0.001, 0.0001

The results for each parameter are presented in separate graphs that show the mean and max value of validation AUROC of every value in the selected range after the hyperparameter tuning is completed. If the validation AUROC is continuously increasing or decreasing within the selected range based on the visible result for parameter filter size, a new parameter range is picked for another hyperparameter tuning. This is because if the performance is fluctuating continuously, it indicates that the optimal value for that parameter to achieve the best result may be outside of the range selected. The hyperparameter tuning and model training workflow is depicted in Fig. 4.

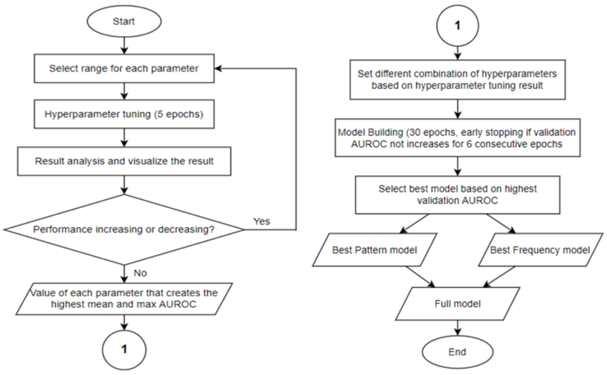

Start
Select range for each parameter
Hyperparameter tuning (5 epochs)
Result analysis and visualize the result
Performance increasing or decreasing?
Yes
No
Value of each parameter that creates the highest mean and max AUROC
1
1
Set different combination of hyperparameters based on hyperparameter tuning result
Model Building (30 epochs, early stopping if validation AUROC not increases for 6 consecutive epochs
Select best model based on highest validation AUROC
Best Pattern model
Best Frequency model
Full model
End

Fig. 4. The workflow of hyperparameter tuning and model training

E. Model Building

The way to set the combinations of hyperparameters is based on the value of each parameter that produces the highest mean and max of the validation AUROC. If the highest mean and maximum values are different, each value will form a combination with the other parameters. All the combinations are then applied to train different models. The models were then trained for a maximum of 30 epochs, however, if the validation AUROC does not improve for 6 continuous epochs, the training process stops. The model is saved if validation AUROC is increased after each epoch. After each combination of hyperparameters has gone through the model training, the best model was chosen based on the highest validation AUROC. The best model from the Pattern branch and

Frequency branch are then utilized to train the full model. Fig. 4 also illustrates the workflow of model training.

F Implement of modified CNN structure

A modified CNN model architecture is used in this study to train the same dataset to learn more about CNN in analyzing NGS data. The initial architecture that based on Tampuu and colleagues [6] is modified by adding a pair of convolution and pooling layer (max pooling layer for Pattern branch, average pooling layer in Frequency branch). Some researchers also used a similar CNN architecture in their findings, which comprises two convolutional layers and two pooling layers [18-19]. The new CNN architecture comprises 2 convolution layers, 2 pooling layers, 2 dropout layers and 2 fully connected layers as shown in Fig. 6.

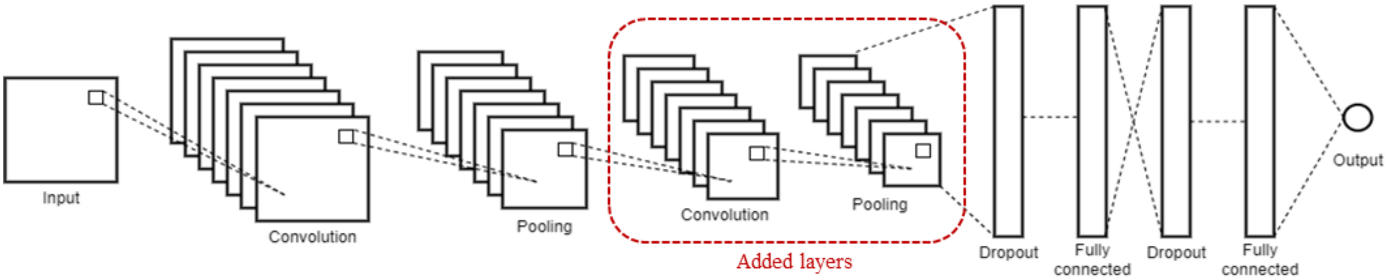


Fig. 6. Modified architecture of CNN in this study

III. Results

The results of the hyperparameter tuning for both pattern branch and frequency branch models in different preprocessing methods (one-hot encoding and ordinal encoding) are shown in this section. Next, the identified sets of hyperparameters are used to train the models for both branches and compared in terms of the validation AUROC. Lastly, this section also shows the performance differences between the existing and the modified CNN architecture in analyzing the same dataset and evaluated in the aspect of validation AUROC, accuracy, sensitivity, and specificity.

A. Hyperparameter Tuning Result

The results of the optimal value for each parameter are shown and the best combination of hyperparameters used for each model is compared according to the different preprocessing methods. The values of each parameter that give the highest AUROC mean and max value for each model based on different preprocessing methods are shown in Table I.

TABLE I. Values of each parameter that produces the best mean and max AUROC value for both pattern and frequency model

Hyperparameter	**Pattern Model**		**Frequency Model**	
	Mean	**Max**	**Mean**	**Max**
One-hot encoding method				
Filter size	10	8	7	7
Layer size	1500	1500	1200	1000
Dropout rate	0.1	0.1	0.1	0.1
Learning rate	0.001	0.001	0.01	0.01
Ordinal encoding method				
Filter size	14	8	7	7
Layer size	1500	1200	1200	1200
Dropout rate	0.1	0.1	0.5	0.5
Learning rate	0.001	0.001	0.01	0.01

In Table I, the column of “Mean” represents the set of hyperparameters that produces the highest mean AUROC for both models using both encoding methods. Similarly, the column of “Max” shows the set of hyperparameters values that produce highest AUROC.

In the one-hot encoding method, the maximum AUROC of the Pattern model is reached with a filter size of 8, whereas the highest average AUROC is achieved with a filter size of 10. For the layer size, dropout rate, and learning rate, the highest mean, and highest max are achieved when the values are 1500, 0.1, and 0.001 respectively. On the other hand, the values of the parameters that yield the highest mean and highest max for the Frequency branch are 7, 0.1, and 0.01 for filter size, dropout rate, and learning rate, respectively. While the highest AUROC is achieved when the layer size is 1000, the highest mean is achieved when the layer size is 1200.

For the ordinal encoding method, both the highest mean and max for filter size and layer size in Pattern Model are different, which are 14 and 8 for filter size, and 1500 and 1200 for layer size. For the rest, the dropout rate and learning rate have the same values for the highest mean and max, which are 0.1 and 0.001 respectively. Surprisingly, the values of the highest mean and max are consistent in all parameters: 7 for filter size, 1200 for layer size, 0.5 for dropout, rate, and 0.01 for learning rate.

Based on the results shown, the dropout rate of 0.1 works best in all models that used different preprocessing methods, except for the Frequency model applied ordinal encoding achieves the best performance when the dropout rate is 0.5. Learning rate can be said to be more consistent in similar CNN architecture, because in the Pattern model for both preprocessing methods, the learning rate of 0.001 works the best, while the optimal learning rate in the Frequency model for both preprocessing methods is 0.01. The filter size in the Frequency model for both preprocessing methods are having the same value which is 7 in both the highest mean and max.

B. Model Training Result

From Table I, these sets of hyperparameters are further used to train corresponding models for 30 epochs with early stopping criteria if the AUROC does not improve for 6 continuous epochs. The performances of each combination of hyperparameters based on AUROC for every model are displayed in Table II.

The best combination for the Pattern model that applied the one-hot encoding method reached the best AUROC, 0.9174 when filter size is 8, layer size is 1500, the dropout rate is 0.1 and learning rate is 0.001. While the filter size 7, layer size 1200, dropout rate 0.1 and learning rate 0.01 work best in the Frequency branch that applied the one-hot encoding method which achieved a higher AUROC (0.9261) than the Pattern model. On the other hand, the performances for the models that employed ordinal encoding is not ideal. The best performance for the Pattern model is 0.7737 with the combination of filter size 14, layer size 1500, dropout rate 0.1 and learning rate 0.001. While the only combination for frequency model, 7 for filter size, 1200 for layer size, 0.5 dropout rate and 0.01 for dropout rate obtained an even lower AUROC which is 0.6025.

TABLE II. AUROC OF EACH COMBINATION OF HYPERPARAMETER FOR PATTERN AND FREQUENCY BRANCH

Model	**Hyperparameter**				**AUROC**
	Filter size	**Layer size**	**Dropout rate**	**Learning rate**	
One-hot encoding method					
Pattern model	**8**	**1500**	**0.1**	**0.001**	**0.9174**
	10	1500	0.1	0.001	0.9035
Frequency model	**7**	**1200**	**0.1**	**0.01**	**0.9261**
	7	1000	0.1	0.01	0.9243
Ordinal encoding method					
Pattern model	**14**	**1500**	**0.1**	**0.001**	**0.7737**
	8	1200	0.1	0.001	0.7487
Frequency model	**7**	**1200**	**0.5**	**0.01**	**0.6025**

After building all the models, the best Pattern and Frequency models with tuned hyperparameters in each preprocessing method are utilized to generate the full model. The comparison of the performance of the full models generated is shown in Table III.

TABLE III. RESULT OF EACH MODEL WITH DIFFERENT ENCODED INPUTS

Model	**AUROC**	
	Ordinal encoded input	**One-hot encoded input**
Pattern model	0.7737	0.9174
Frequency model	0.6025	0.9261
Full model	0.7782	0.9216

Based on the results, the AUROC of pattern, frequency and full models that employed ordinal encoded input are less than 0.8, meanwhile the full model that employed one-hot encoded input obtains the highest AUROC which is more than 0.9 in the three models.

C. Convention CNN architecture vs new CNN architecture

There are several sets of layer sizes used to determine which is the best for the Pattern and Frequency models that applied the new CNN architecture. After running each layer size set, the best layer sizes for the Pattern model are 256 for the first convolutional layer and 128 for the second convolutional layer which obtained 0.9174 AUROC. On the other hand, the best layer sizes set for the Frequency model are 38 and 19 for first and second convolutional layers respectively. The AUROC of the optimal Frequency model is slightly higher than that of the Pattern model which is 0.9059. After identifying the best models in both Pattern and Frequency models, these models are then merged and built into the full model. The Full model greatly improved from the previous models which achieved highest AUROC which is 0.9612.

To identify the better CNN architecture, the comparison of the three models that applied different CNN architectures are shown in Table IV. Although the AUROC of Pattern model and the Frequency model that applied new CNN architecture is lower than that of the convention architecture, the Full model that applied new CNN architecture obtains the highest AUROC among all the models.

TABLE IV. RESULT OF EACH MODEL WITH DIFFERENT CNN ARCHITECTURES

Model	**AUROC**	
	Existing architecture	**Modified architecture**
Pattern model	0.9174	0.9021
Frequency model	0.9261	0.9059
Full model	0.9216	0.9612

The detailed data of each performance measurement which are the accuracy, sensitivity and specificity of Pattern, Frequency and Full models are displayed in Table V.

TABLE V. The accuracy, sensitivity and specificity of Pattern, Frequency and Full Models with Convention and New CNN Architectures

CNN architecture	Model		
	Pattern Model	**Frequency Model**	**Full Model**
Accuracy (%)			
Convention	97.56	97.77	98.10
New	97.50	94.69	**98.27**
Sensitivity (%)			
Convention	72.58	76.67	91.30
New	**93.55**	38.94	90.20
Specificity (%)			
Convention	99.04	99.21	99.77
New	**99.89**	96.10	99.72

From the data shown in Table V, the accuracy obtained from both CNN architectures for the 3 models are quite good with more than 90%. The accuracy of the full models is the highest in both convention and new CNN architecture among the three models. This indicates that the full model improves from the Pattern and Frequency models as it is the combination of these two models.

Sensitivity is the ability of the models to correctly predict those DNA sequences that are viral. Based on the data, for conventional CNN architecture, the full model achieves the highest sensitivity (91.30%) among the models, while the pattern model reaches the highest sensitivity (93.55%) in the Pattern model among the models that applied new CNN architecture. High sensitivity is important in the predictive model as it can be used to identify the viral DNA sequences. High sensitivity means that the model can predict most of the viral sequences correctly while only a small portion is left undetected.

The specificity among the three different models of both CNN architectures achieves a great value which all are more than 90%. The highest specificity for models that implemented convention and new CNN architectures fall on Full model (91.30%) and Pattern model (93.55%) respectively. Specificity refers to the ability of the models to precisely predict the non-viral sequences. As all the models having 95% specificity and above, it indicates that the models correctly predict these 95% and above non-viral DNA sequences as non-viral, and only not more than 5% non-viral DNA sequences are incorrectly predicted as viral sequences which is also known as false positives. The model that has low sensitivity but high specificity such as the Frequency model that employed new CNN architecture, results in many DNA sequences that are viral being predicted as the non-viral sequences and are left out for further investigation.

IV. Disscussion

From the result of hyperparameter tuning, each model has the different sets of hyperparameter combinations which demonstrates that different models employ different hyperparameter combinations to produce the best results, and that is why hyperparameter tuning is required in the experiment. The result also reveals that the one-hot encoding preprocessing approach for DNA sequences produced a better result than ordinal encoding. This shows the capability of one-hot encoding to transform the raw DNA sequences into a more informative computer-readable format than ordinal encoding. This might be due to the one-hot encoding method converting each nucleotide base into a matrix form, whereas ordinal encoding only converts the nucleotide bases into different numerical numbers. It also proves that one-hot encoding is a more suitable preprocessing method for DNA sequences to be used in CNN architecture. Despite the fact that the ordinal encoding method creates a minimal dimension for the input data, the values provided to the nucleotide bases create a manual ordering of the input elements, which may bias the representation and, in turn, reduce the model's performance [22]. Furthermore, the new CNN model design produces superior results than the conventional CNN architecture. This means that when additional layers are added to the CNN design, more information is collected, and, as a consequence, better results are produced, despite the longer processing time.

V. Conclusion

Different models have different combinations of hyperparameters to achieve the highest performance for the CNN model in predicting viral sequences. Results from this study also shows that one-hot encoding is a more suitable preprocessing method for NGS data compared to ordinal encoding in CNN. Moreover, adding more layers to the CNN structure improves the performance of CNN models to detect the viral sequences. This study investigated the potential of CNN in analysis of NGS data for viral sequence detection. However, it does not cover all possibilities that can be further explored.

The recommendation for future work is to increase the sample size. In this study, only 18,455 of the DNA sequences are extracted and used in the experiment from the total 264,029 samples due to the time constraint. When the number of samples rises, the accuracy of the model also increases. The application of different DNA sequences length can also be another suggestion for future work. In this study, all the DNA sequences used are the length 300. Other sequence lengths can also be used if the computational resources are allowed. Meanwhile, implementation of different optimizer such as stochastic gradient descent (SGD), Adagrad and others that might further improve the performance.

Acknowledgment

Authors wish to thank Universiti Teknologi Malaysia for supporting this work. This work was supported by the Ministry of Higher Education under Fundamental Research Grant Scheme (FRGS/1/2019/ICT02/UTM/02/12).

References

[1] Behjati, S., & Tarpey, P. S. (2013). What is next generation sequencing? *Arch Dis Child Educ Pract Ed.,* 236-238. Doi: 10.1136/archdischild-2013-304340.

[2] Akker, J. v., Mishne, G., Zimmer, A. D., & Zhou, A. Y. (2018). A machine learning model to determine the accuracy of variant calls in capture-based next generation sequencing. *BMC Genomics*. Doi: https://doi.org/10.1186/s12864-018-4659-0.

[3] Li, Z., Wang, Y., & Wang, F. (2018). A study on fast calling variants from next-generation sequencing data using decision tree. *BMC Bioinformatics.* Doi: https://doi.org/10.1186/s12859-018-2147-9.

[4] Spinella, J.-F., Mehanna, P., Vidal, R., Saillour, V., Cassart, P., Richer, C., Sinnett, D. (2016). SNooPer: A machine learning-based method for somatic variant identification from low-pass next-generation sequencing. *BMC Genomics.* Doi: https://doi.org/10.1186/s12864-016-3281-2.

[5] Lopez-Rincon, A., Tonda, A., Mendoza-Maldonado, L., Claassen, E., Garssen, J., & Kraneveld, A. D. (2020). Accurate identification of SARS-CoV-2 from viral genome sequences using deep learning. *bioRxiv*. Doi: https://doi.org/10.1101/2020.03.13.990242.

[6] Tampuu, A., Bzhalava, Z., Dillner, J., & Vicente, R. (2019). ViraMiner: Deep learning on raw DNA sequences for identifying viral genomes in human samples. *Plos One.* Doi: https://doi.org/10.1371/journal.pone.0222271.

[7] Bzhalava, Z., Hultin, E., & Dillner, J. (2018). Extension of the viral ecology in humans using viral profile hidden Markov models. *Plos One.* Doi: https://doi.org/10.1371/journal.pone.0190938.

[8] Labonté, J. M., & Suttle, C. A. (2013). Previously unknown and highly divergent ssDNA viruses populate the oceans. *The ISME Journal*, 2169-2177. Doi: https://doi.org/10.1038/ismej.2013.110.

[9] Mistry, J., Finn, R. D., Eddy, S. R., Bateman, A., & Punta, M. (2013). Challenges in homology search: HMMER3 and convergent evolution of coiled-coil regions. *Nucleic Acids Research*, e121. Doi: https://doi.org/10.1093/nar/gkt263.

[10] Bzhalava, Z., Tampuu, A., Bała, P., Vicente, R., & Dillner, J. (2018). Machine learning for detection of viral sequences in human metagenomic datasets. *BMC Bioinformatics.* Doi: https://doi.org/10.1186/s12859-018-2340-x.

[11] Kandpal, M., & Davuluri, R. V. (2020). Identification of geographic specific SARS-Cov-2 mutations by random forest classification and variable selection methods. *Stat Appl.*, 253-268.

[12] Brion, G., Viswanathan, C., Neelakantan, T. R., Lingireddy, S., Girones, R., Lees, D., Vantarakis, A. (2005). Artificial neural network prediction of viruses in shellfish. *Public Health Microbiology*, 5244-5253. Doi: 10.1128/AEM.71.9.5244-5253.2005.

[13] Seguritan, V., Jr., N. A., Arnoult, M., Raymond, A., Lorimer, n., Jr., A. B., Segall, A. M. (2012). Artificial neural networks trained to detect viral and phage structural proteins. *Plos Computational Biology*. Doi: https://doi.org/10.1371/journal.pcbi.1002657.

[14] Ren, J., Song, K., Deng, C., Ahlgren, N. A., Fuhrman, J. A., Li, Y., Sun, F. (2020). Identifying viruses from metagenomic data using deep learning. *Quantitative Biology,* 64-77. Doi: https://doi.org/10.1007/s40484-019-0187-4.

[15] Nguyen, N. G., Tran, V. A., Ngo, D. L., Phan, D., Lumbanraja, F. R., Faisal, M. R., Satou, K. (2016). DNA sequence classification by convolutional neural network. *Biomdesical & Life Sciences.* Doi: 10.4236/jbise.2016.95021.

[16] Spanhol, F. A., Oliveira, L. S., Cavalin, P. R., Petitjean, C., & Heutte, L. (2017). Deep features for breast cancer histopathological image classification. *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 1868-1873. Doi: 10.1109/SMC.2017.8122889.

[17] Choong, A. C., & Lee, N. K. (2017). Evaluation of convolutionary neural networks modeling of DNA sequences using ordinal versus one-hot encoding method. *2017 International Conference on Computer and Drone Applications (IConDA)*, 60-65.

[18] Gunasekaran, H., Ramalakshmi, K., Arokiaraj, A. R., Kanmani, S. D., Venkatesan, C., & Dhas, C. S. (2021). Analysis of DNA sequence classification using CNN and hybrid models. *Computational and Mathematical Methods in Medicine*. Doi: https://doi.org/10.1155/2021/1835056.

[19] Aoki, G., & Sakakibara, Y. (2018). Convolutional neural networks for classification of alignments of non-coding RNA sequences. *Bioinformatics*, 237-244. Doi: https://doi.org/10.1093/bioinformatics/bty228.

[20] Ronao, C. A., & Cho, S.-B. (2015). deep convolutional neural networks for human activity recognition with smartphone sensors. *Neural Information Processing,* (pp. 46-53). Springer, Cham. Doi: https://doi.org/10.1007/978-3-319-26561-2_6.

[21] Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. *Computer Vision and Pattern Recognition.*

[22] Lovino, M., Urgese, G., Macii, E., Cataldo, S. D., & Ficarra, E. (2019). A deep learning approach to the screening of oncogenic gene fusions in humans. *International Journal of Molecular Sciences*, 1645. Doi: https://doi.org/10.3390/ijms20071645.