VTyper - Flavivirus Sequence Classification/Prediction Model

By: Elizabeth Lim

Updated Log: 03/07/2025

Summary

The objective of this experiment was to train and implement a model capable of predicting or classifying viral species based on a given input cDNA or genomic sequence. A convolutional neural network (CNN) model was built to classify seven flavivirus species. The outcome could offer a faster alternative to computationally intensive methods like Blast for batch screening and sorting of contigs or assembled sequences.

Sequences from seven flavivirus species were downloaded from NCBI Refseq and Genbank. A dedicated final test set was set aside to prevent data contamination during training. Sequences were vectorized using one-hot encoding, that is, mapping nucleotides to unique binary vectors so that they can be input into the model for training and classification. The model architecture was a basic convolutional neural network (CNN), featuring a single convolutional 2D layer, with MaxPooling2D layer, and dense layers culminating in a SoftMax output layer.

The initial model achieved 96% test accuracy. Further simplification of the model, by removing a dense layer, improved accuracy to 97.8%. When evaluated with the unseen final test set, only 7 out of 160 predictions did not match true labels. It was also found that the model was able to detect a possible misidentified sequence in Genbank. One sequence (MZ284953.1) that was stated as Dengue virus Type 3 in the Genbank database was predicted by the model as Dengue virus Type 1 (99% confidence). The sequence from Genbank was checked using Blastn which confirmed the prediction result.

In conclusion, a CNN model was successfully trained for classifying flavivirus genomes or sequences. This model may be applied to quick screening for identification of sequences or contigs from complex samples, and finding errors in public databases.

Keywords: Flavivirus, sequence classification, convolutional neural network, deep learning

Contents

1. Introduction

The gold standard Blast method for screening contigs and assemblies is based on sequence alignment for comparison of sequences for applications such as identification, species typing and phylogenetic analysis. The Blast method is highly accurate and yields important high-resolution information enabling base to base sequence comparisons, analysis of regions of match/mismatch, etc. However, this method can be time consuming and computationally intensive if hundreds or thousands of contigs need to be identified. This is a possible issue when metagenomic assemblies from complex samples yield fragmented genomes from both target and non-target organisms or agents (Liu et al 2022, Tampuu et al 2019).

Deep learning can enable neural network models to identify patterns in genomic sequences, allowing classification of viral species by their unique features (Chen et al 2024). Genomic sequences can be viewed as distinctive portraits made up of bases (A, C, G, T) instead of pixels, that define each species. Seen in this way, genomic sequences can be vectorized (just as image data can be vectorized) and then used to train a convolutional neural network (CNN) for the purpose of classification or prediction of species type.

The use of CNN models for sequence classification and feature extraction is not a novel idea. A search on Google Scholar turned up 11000 entries (search term: ‘CNN models for nucleotide sequence classification’, since: ‘2024’).

In this learning project, a CNN model will be trained to classify 7 flavivirus species based on a given input nucleotide sequence.

2. Objective

Train and implement a model that can predict or classify viral species based on a given input cDNA or genomic sequence.

3. Approach

The model will have a very basic convolutional neural network architecture for feature learning and a dense network for classification with a SoftMax activation function in the final output layer.

Sequence data of 7 flaviviruses were downloaded from NCBI Refseq and Genbank was used to train a CNN-based model.

4. Methods and Results

4.1. Downloading and data organization

Sequences from 7 flavivirus species were downloaded from NCBI Refseq and Genbank through the NCBI Virus community portal (https://www.ncbi.nlm.nih.gov/labs/virus/vssi/#/, accessed June 2025).

The virus species included:

i) Dengue Virus Type 1 (Den1)
ii) Dengue Virus Type 2 (Den2)
iii) Dengue Virus Type 3 (Den3)
iv) Dengue Virus Type 4 (Den4)
v) Zika Virus (ZKV)
vi) West Nile Virus (WNV)
vii) Japanese Encephalitis Virus (JEV)

Selected parameters for downloading: Nucleotide Completeness: Complete, Sequence Length Min:10,000.

Number of sequences (samples) per virus species (class)

```
Number of samples per class:
- Dengue1: 1924
- Dengue2: 1656
- Dengue3: 1020
- Dengue4: 256
- JapaneseEncephalitisVirus: 538
- WestNileVirus: 2089
- Zika: 370
```

Box 1: Number of files downloaded from NCBI Virus Portal. The sequences originate from NCBI Genbank.

4.2. File processing

Each downloaded fasta file contained multiple sequences in each file.

A python script was used to extract each sequence from the downloaded file and save it as a single fasta sequence file.

From the datapool, for each species, 20-25 sequences were put aside as a final test set for post training evaluation. (Table 1)

The training set will be split into training, validation and test sets in the process of training. The final test is set aside for final test evaluation and will not be part of the training process. It is set aside in this manner to reduce change of data contamination.

Table 1: Number of training and testing sequence files

Flavivirus species	**Training Set**	**Final Test Set**
Dengue Virus Type 1 (Den1)	1899	25
Dengue Virus Type 2 (Den2)	1631	25
Dengue Virus Type 3 (Den3)	995	25
Dengue Virus Type 4 (Den4)	236	20
Zika Virus (ZKV)	350	20
West Nile Virus (WNV)	2064	25
Japanese Encephalitis Virus (JEV)	518	20

4.3. Data preparation: Vectorization of the sequences

Sequences are vectorized using one-hot encoding approach as follows:

```
NUCLEOTIDE_MAP = {
    'A': [1, 0, 0, 0],
    'C': [0, 1, 0, 0],
    'G': [0, 0, 1, 0],
    'T': [0, 0, 0, 1],
    'N': [0, 0, 0, 0] # Represent 'N' as all zeros
}
# Default encoding for any character not in NUCLEOTIDE_MAP (e.g., 'R', 'Y', etc.)
UNKNOWN_NUCLEOTIDE_ENCODING = [0, 0, 0, 0]
```

(Note1: This method of encoding is suitable for small genomes such as viruses but may not be efficient for larger genomes (bacteria, eukaryotes etc). In published works, Kmer encoding is more commonly used (Ftci 2024).)

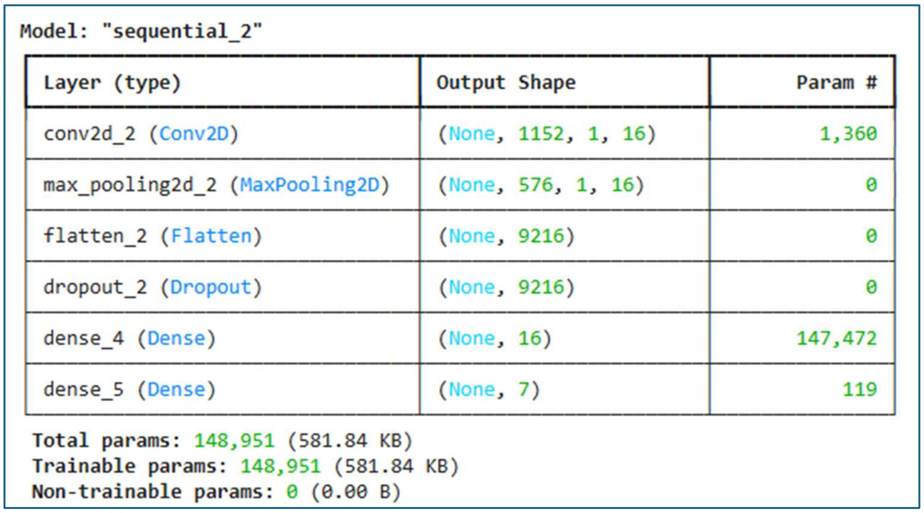
Data was split into training (70%), validation (15%) and testing (15%) datasets.

4.4. Initial model design, training and evaluation

Model design, training and evaluation are performed using Jupyter Notebook running in Google Colab (with GPU). Python libraries included numpy, TensorFlow/Keras, and matplotlib.

The model is based on a paper by Lopez-Rincon (2021) which included details of the convolutional neural network (CNN) used in that study.

For this learning project, the CNN model has a single convolutional 2D layer `[Conv2D(16, kernel_size=(21, 1), strides=(10, 1), padding='same', activation='relu')]` followed by a MaxPooling2D layer `[MaxPooling2D(pool_size=(2, 1)]`. This is followed by a flattening layer, a dropout layer (set to 0.3 dropout rate), a dense layer with 16 output channels and a final layer with 7 output channels (one per virus species 'class') and SoftMax activation function. (Figure 1)



```
Model: "sequential_2"

 Layer (type)                    Output Shape               Param #
 conv2d_2 (Conv2D)               (None, 1152, 1, 16)          1,360
 max_pooling2d_2 (MaxPooling2D)  (None, 576, 1, 16)               0
 flatten_2 (Flatten)             (None, 9216)                     0
 dropout_2 (Dropout)             (None, 9216)                     0
 dense_4 (Dense)                 (None, 16)                 147,472
 dense_5 (Dense)                 (None, 7)                      119

 Total params: 148,951 (581.84 KB)
 Trainable params: 148,951 (581.84 KB)
 Non-trainable params: 0 (0.00 B)
```

Figure 1: Model architecture

According to a few other papers, models used for sequence classification tends to keep to a simple architecture. In testing and adjusting, additional layers and nodes were added, and it was found that model complexity was detrimental to model performance and resulted in extreme overfitting. Adam optimizer was used with 0.0001 learning rate. Categorical cross entropy loss function was used, and accuracy was used.

(Note2: In machine learning -- more complex models, more nodes do not necessarily equate to better performance. The lesson one learns is that there is a 'sweet spot' and the job of training is not just to make the model better but to find that spot that is just right.)

Each iteration of the model was trained for 10 epochs and evaluated using test data set. To monitor training, accuracy and loss graphs were generated and analysed. Confusion matrix was used to visualize and evaluate test data results.

4.5. Testing results of initial design and try-out

Our model (Model 5a) was found to have a test accuracy of 0.9610 (96%) with a test loss of 0.1784.

The confusion matrix generated (Figure 2) shows there is few misclassifications.

(Note3: Dengue 4, JEV and Zika Virus constituted smaller datasets compared to the others, yet it is interesting that there is quite a few samples that were misclassified as Dengue 1 when they were labelled as 'Dengue 2'.

It would be possible to increase the data pool for Dengue 4, ZKV and JEV. I am hesitant in using incomplete genomic / nucleotide assemblies. The viruses are quite close (especially the four dengue types, and between WNV and JEV) and if they are incomplete, there is a higher chance of misclassification.)

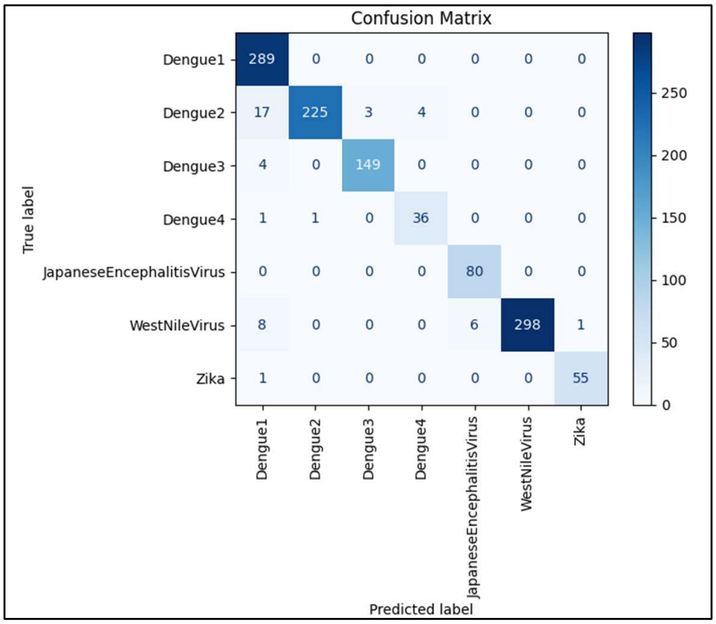
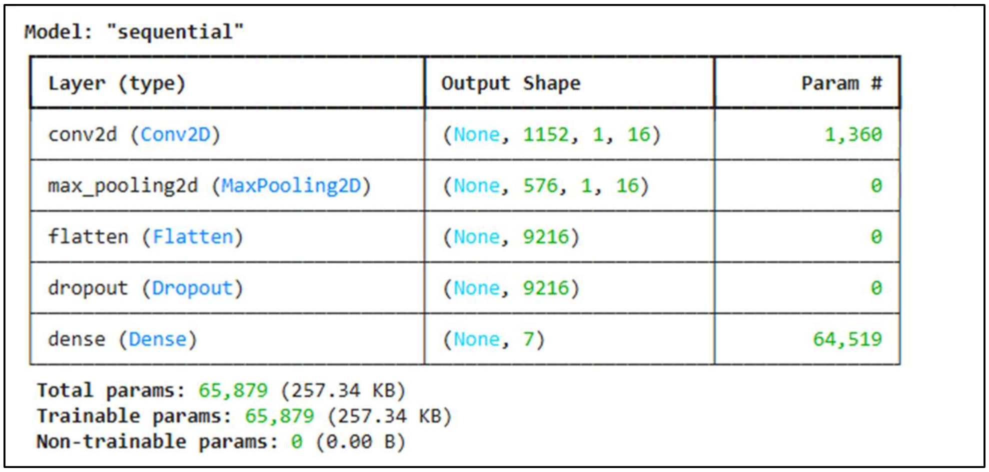


Figure 2: Confusion Matrix for Model 5

4.6. Final model and test results

The model was further simplified with the removal of one dense layer. (Figure 3)



```
Model: "sequential"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                   ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ conv2d (Conv2D)                │ (None, 1152, 1, 16)    │         1,360 │
├────────────────────────────────┼────────────────────────┼───────────────┤
│ max_pooling2d (MaxPooling2D)   │ (None, 576, 1, 16)     │             0 │
├────────────────────────────────┼────────────────────────┼───────────────┤
│ flatten (Flatten)              │ (None, 9216)           │             0 │
├────────────────────────────────┼────────────────────────┼───────────────┤
│ dropout (Dropout)              │ (None, 9216)           │             0 │
├────────────────────────────────┼────────────────────────┼───────────────┤
│ dense (Dense)                  │ (None, 7)              │        64,519 │
└────────────────────────────────┴────────────────────────┴───────────────┘
 Total params: 65,879 (257.34 KB)
 Trainable params: 65,879 (257.34 KB)
 Non-trainable params: 0 (0.00 B)
```

Figure 3: Model Architecture: Model5a

Training was carried out for 10 epochs with a batch size of 32. Adam Optimizer was used with a learning rate set at 0.0001. Model training was monitored using accuracy and loss graphs. (Figure 4)

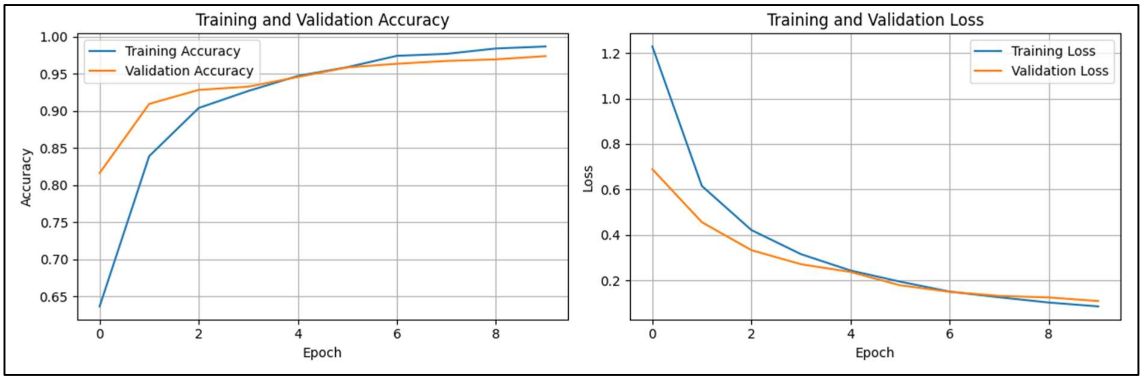


Figure 4: Training and validation accuracy and loss graphs for Model 5a.

This model was found to have a test accuracy of 97.8% (Test Loss: 0.0995 Test Accuracy: 0.9779). An improvement over the initial model. This demonstrates that a simpler model had better performance.

A confusion matrix was plotted to see if the model had any obvious biases (Figure 5). The four Dengue Virus types are closely related and as such it was not unexpected that there are some disparities between predicted and true labels with regards to that.

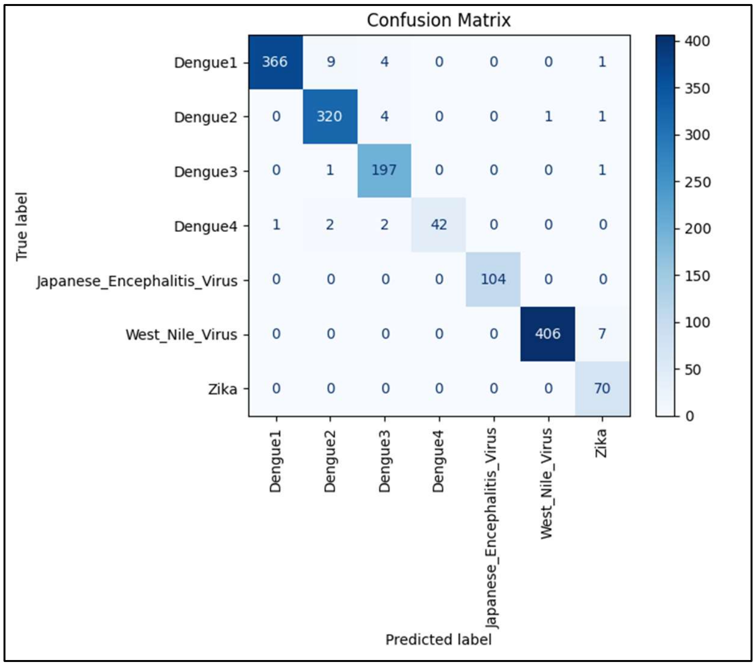


Figure 5: Confusion matrix based on the testing results for Model 5a.

4.7. Evaluation using final test set

Finally, the model was tested using the final test set of sequences that was set aside prior to model training.

The test sequences were similarly vectorized so that it can be processed by the model.

The results seem to be very good (only 7 out of 160 predictions didn't match the label).

The results were also compiled into a pandas data frame with accession numbers, true/predicted labels and prediction scores. This is exported as a csv file.

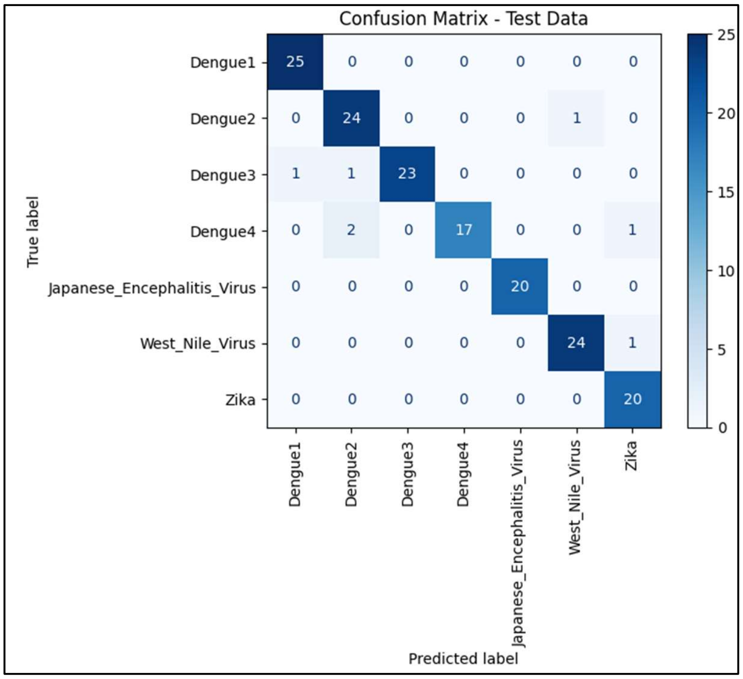


Figure 6: Confusion matrix of final test evaluation.

The seven samples where the predicted labels do not match the true labels were further investigated.

Using the accession numbers, the original sequences of these seven samples are checked using Blastn (https://blast.ncbi.nlm.nih.gov/Blast.cgi) to verify their identity. (Table 2)

The model makes the call on the identity based on the highest prediction score. It was observed that for 6 of these samples, the prediction scores were not very high (0.26 – 0.38 or 26-38%). In most cases, correct calls tend to have higher prediction scores (>0.50 or 50%).

For sequence with accession number MZ284953.1, it was found that although it was deposited and labelled as Dengue 3 in NCBI Genbank database, yet Blastn revealed that this sequence is most likely Dengue 1 with 100% match to Dengue 1

NC_001477.1. As such this simple model was able to detect a possible misidentified sequence deposited in Genbank.

Table 2: Prediction scores for the 7 samples that were misclassified.

Accession Number	True Label	Predicted Label	Prediction scores (Probability %)							
			Den 1	**Den 2**	**Den 3**	**Den 4**	**JEV**	**WNV**	**ZKV**	**Blastn Results**
KP188554.1	Dengue2	West Nile Virus	0.083	0.150	0.122	0.021	0.099	0.284	0.241	Dengue 2
KU509279.1	Dengue3	Dengue2	0.269	0.329	0.221	0.059	0.017	0.028	0.078	Dengue 3
MZ284953.1	Dengue3	Dengue1	0.993	0.003	0.001	0.001	0.000	0.001	0.001	Dengue 1 NC_001477.1
KP188560.1	Dengue4	Dengue2	0.095	0.360	0.345	0.127	0.009	0.022	0.042	Dengue 4
LC069810.1	Dengue4	Dengue2	0.214	0.306	0.253	0.104	0.015	0.050	0.058	Dengue 4
MG601754.1	Dengue4	Zika	0.061	0.151	0.051	0.012	0.119	0.225	0.381	Dengue 4
HQ671691.1	West Nile Virus	Zika	0.062	0.183	0.108	0.051	0.040	0.242	0.314	West Nile Virus

5. Conclusion

A CNN model that can be used for classification of flavivirus genomes or sequences was successfully trained and implemented.

The development of a CNN model for classification of genomes or sequences based on may be useful for the following applications:

i) Quick screening for identification of sequences or contigs.
ii) Finding errors in the database. Quick screening of sequences in a database for mislabelled or misidentified sequences.

6. References

1. Çi Ftçi B, Teki N R. Prediction of viral families and hosts of single-stranded RNA viruses based on K-Mer coding from phylogenetic gene sequences. Comput Biol Chem. 2024 Oct;112:108114. doi: 10.1016/j.compbiolchem.2024.108114. Epub 2024 May 31. PMID: 38852362.

2. Chen Z, Ain NU, Zhao Q, Zhang X. From tradition to innovation: conventional and deep learning frameworks in genome annotation. Brief Bioinform. 2024 Mar 27;25(3):bbae138. doi: 10.1093/bib/bbae138. PMID: 38581418; PMCID: PMC10998533.

3. Liu, S., Moon, C.D., Zheng, N. *et al.* Opportunities and challenges of using metagenomic data to bring uncultured microbes into cultivation. *Microbiome* **10**, 76 (2022). https://doi.org/10.1186/s40168-022-01272-5

4. Lopez-Rincon A, Tonda A, Mendoza-Maldonado L, Mulders DGJC, Molenkamp R, Perez-Romero CA, Claassen E, Garssen J, Kraneveld AD. Classification and specific primer design for accurate detection of SARS-CoV-2 using deep learning. Sci Rep. 2021 Jan 13;11(1):947. doi: 10.1038/s41598-020-80363-5. PMID: 33441822; PMCID: PMC7806918.

5. Tampuu A, Bzhalava Z, Dillner J, Vicente R. ViraMiner: Deep learning on raw DNA sequences for identifying viral genomes in human samples. PLoS One. 2019 Sep 11;14(9):e0222271. doi: 10.1371/journal.pone.0222271. PMID: 31509583; PMCID: PMC6738585.