(ISSN: 0973-7049(P))

The Bioscan AN INTERNATIONAL QUARTERLY JOURNAL OF LIFE SCIENCES 21(2): S.I (2), 21305-21335, 2026 www.thebioscan.com

MULTI-SCALE FEATURE LEARNING WITH SELF-ATTENTION LSTM FOR ROBUST CLASSIFICATION OF COVID-19 GENOMIC VARIANTS

[1*]Rajkumar Pandiarajan, [2]Muthu Lakshmi G

[1*]Research Scholar, Department of Computer Science and Engineering, Manonmaniam Sundaranar University, Tirunelveli-627 012, Tamil Nadu, India.

[2]Associate Professor, Department of Computer Science and Engineering, Manonmaniam Sundaranar University, Tirunelveli-627 012, Tamil Nadu, India.

[*]Corresponding Author Email Id: rajkumarpandiarajan9@gmail.com

DOI:https://doi.org/10.63001/tbs.2026.v21.i02.S.I(2).pp21305-21335

KEYWORDS

Genome Sequence Classification; K-mers; Long short-term memory, bioinformatics, and recurrent neural networks.

Received on:
31-05-2026
Accepted on:
20-06-2026
Published on:
30-06-2026

Abstract

Accurate and scalable classification of SARS-CoV-2 variants based on their genomics sequences is critical for efficient surveillance and public health management. However, conventional deep learning models suffer from issues such as modeling the complexity of high-dimensional genomics sequences, learning long-range dependencies, and ensuring generalization capabilities for newly emerging variants. In this study, a new deep learning model called Three-Level Hierarchical Deep Learning with SA-LSTM is presented for COVID-19 variants classification. Genome sequences are represented in the form of overlapping K-mers with different levels of granularity and then embedded through Continuous Bag of Words (CBOW). The designed three-level feature extraction framework utilizing convolutional and recurrent layers at low-, mid-, and high-level representations allows for the discrimination of discriminatory genomic patterns through multi-scale learning. Self-attention is then applied to refine the extracted features in order to highlight the relevant biological motifs and long-term dependencies. Extensive experimental results on SARS-CoV-2 publicly available genome data reveal that our proposed system outperforms all existing deep learning approaches, obtaining an accuracy of 99.56%, a high F1-Score, MCC, and lower inference time. These findings underscore the importance of hierarchical representation learning along with attention-guided temporal modeling for genomic sequence classification, thus providing a scalable and understandable solution for COVID-19 variant surveillance and other future applications of genomic epidemiology.

1. INTRODUCTION

Despite the extensive use of vaccines, the continued emergence of SARS-CoV-2 strains poses risks as regards the global health status since the strain might influence transmission, pathogenicity, and vaccine effectiveness [1]. There have been data-driven models for epidemiology as well as hybrid methods that play an important role in virus propagation analysis and evolution during the pandemic [2,3]. Sequencing and classification help detect novel strains, whereas light-weight diagnostic and surveillance systems play an important role in the monitoring process and hybrid detection process [4,5]. Nevertheless, there are issues concerning

the identification of novel strains as vision-based systems are influenced by different parameters [6].

Genetic pipeline models and association studies encounter issues due to lack of diversity and complicated genetic interaction, making predictions less reliable [8,9]. Whole Genome Sequencing (WGS), although very precise, is time-consuming, costly and difficult to deploy in resource-poor areas, making it not suitable for mass screening of COVID-19 variants [10]. Studies of COVID-19 variants employ deep learning (DL) algorithms due to the nature of biomedical datasets, being of high dimensionality. CNNs automatically learn the relevant features from images, biological signals or genome data [11], while LSTM and RNN learn temporal and sequential dependencies [12]. The attention mechanism focuses on important features for better classification results. The integration of CNNs and RNN/LSTM networks with feature selection increases accuracy, robustness, and interpretability [14] and the use of explainable methods enables identification of important features as well as building clinician's trust in variant prediction [15].

The appearance of new variants of COVID-19 poses limitations of existing systems in terms of accuracy, scalability, and explainability. Several deep learning models have problems in dealing with high dimensional genomics data due to inability to extract necessary sequential patterns. At the same time, expensive methods such as whole genome sequencing cannot be implemented in large scale and real-time. Therefore, there is a necessity in development of an efficient deep learning model that would be able to learn hierarchical features and concentrate on important genomic patterns. The paper's primary contributions are as follows:

- A hierarchical K-mer–based preprocessing pipeline is introduced to transform raw genome sequences into English-like textual representations,

enabling contextual feature learning using CBOW embeddings.

- A new multi-level feature learning framework is proposed by integrating convolutional and recurrent networks at low-, medium-, and high-level representations to capture complementary genomic patterns
- A Self-Attention LSTM (SA-LSTM) module is incorporated to emphasize biologically significant genomic regions as well as long-range dependencies, improving classification robustness and interpretability.
- The suggested framework performs at the cutting edge of COVID-19 variant categorization while maintaining lower inference latency compared to conventional CNN, ResNet, CNN-BiLSTM, and VGG-based models.

The following sections have already been arranged as follows: In Section 2, the review of research and literature will be done, in Section 3, the recommended framework will be introduced; in Section 4, the results and discussion will be analyzed, and finally in Section 5, the conclusion of the study will be made.

2. LITERATURE REVIEW

A selection of current studies about COVID-19 variant *were reviewed in this section.*

In 2024, Matson et al. [16] proposed a method for predicting the alteration of the neutralizing antibodies against variants using deep learning (DL), where they found a decrease in neutralizing antibodies against the new generation of XBB variants (EG.5, FL.1.5.1, XBB.1.16) because of the F456L spike mutation. Talukder et al. [17] Utilized DL on radiography images using a highly customized EfficientNetB4 model to achieve accurate classification in terms of fast COVID-19 detection and the possibilities for other medical image applications. In 2025, Fekri-Ershad & Dehkordi [18] developed a double-channel CNN architecture in terms of variant detection through an adaptive LBP feature extractor together with a spatial feature

extractor using Adam and focal loss. Hussain et al. [19] introduced the Multi-Layer Ensemble Framework (C19-MLE) for identifying COVID-19 in both early and advanced phases through coughing sound and X-ray chest image classification (2D CNN, VAE, UNet, ResNet-50, InceptionV3). Hu et al. [20] used attention mechanism and transfer learning to enhance RNN, LSTM, GRU, and TCN models in city-level COVID-19 case prediction; TLLA model outperformed other models in detecting both flattened and jagged patterns in COVID-19 case numbers.

In 2025, Azarfar et al. [21] compared DL models, LSTM, and RNNs for predicting SARS-CoV-2 vaccine efficacy in patients with chronic conditions 12 months post-vaccination, finding that traditional machine learning methods (logistic regression, SVR, random forest, gradient boosting) performed comparably, highlighting AI's potential to improve vaccination strategies for high-risk patients. Mekouar [22] adopted WHO dataset to model the transmission dynamics of COVID-19 infection across the African countries through an LSTM model with attention and data augmentation, performing better than GRU, LSTM, and BiLSTM models, having lesser validation errors. Fahim et al. [23] presented a novel hybrid architecture CNN-GRU-TabNet that made use of the structured data from India to predict the COVID-19-related fatalities with high accuracy and interpretability, wherein CNN model was used to learn the short-term dependencies, whereas GRU was used to tackle missing values, and TabNet was used to enhance the features. Antunes et al. [24] designed CTCCovid19, which is a DL model based on ResNet-50, to detect the presence of COVID-19 from the CT scan images. Dutta et al. [25] introduced QNN, a novel architecture of DL model, to diagnose COVID-19 through Raman spectroscopy, performing better than vanilla DL model like Vanilla CNN, MobileNet, Xception, and VGG-16.

2.1. Research Gaps

Deep learning for the discovery and analysis of COVID-19 variant identification has improved significantly in recent times. However, there are certain key challenges that persist in this regard. Most of the existing methods have a major limitation regarding the ability to discover complementary low-, mid-, and high-level biological patterns from the genomic sequences because of the application of the single scale approach to feature extraction. Long-range interactions among distant mutations, an important factor in discriminating between closely related SARS-CoV-2 variants, are poorly represented by these models. In addition, traditional methods of encoding sequences are insufficient in terms of their ability to capture context-dependent and semantically related dependencies between subsequences of nucleotides, which leads to poor representation learning. State-of-the-art models tend to be costly and require intensive whole-genome sequencing workflows, making them unsuitable for immediate and large-scale surveillance. Besides, many deep learning models focus more on the accuracy of predictions than on attention-based interpretability, thus failing to identify biologically significant genomic segments. These aspects point to the necessity of an integrated, interpretable deep learning model that captures multi-level patterns and stresses certain variant-related characteristics.

3. PROPOSED METHODOLOGY

The proposed methodology will use an AI-based deep learning model for classification of COVID-19 genome sequence. The data on COVID-19 sequences collected from various sources (public databases such as NCBI) have been checked and labeled according to the variant. DNA sequences (represented by A, T, C, G) were divided into k-length overlapping subsequences called K-mers in order to convert genomic data into deep learning-ready data. Continuous Bag of Words (CBoW) will be used to encode the semantics of K-mers, while Label Binarize

will encode the classes of variants. The data will be fed into a three-layer model including convolutional layers (Conv11, Conv52, Conv152) as well as recurrent layers for learning temporal features. The Self-Attention Long Short Term Memory (SA_LSTM) neural network will further encode features. The whole methodology was implemented in Python programming language, and the suggested framework, depicted in Figure 1, seems to be quite appropriate for the classification problem.

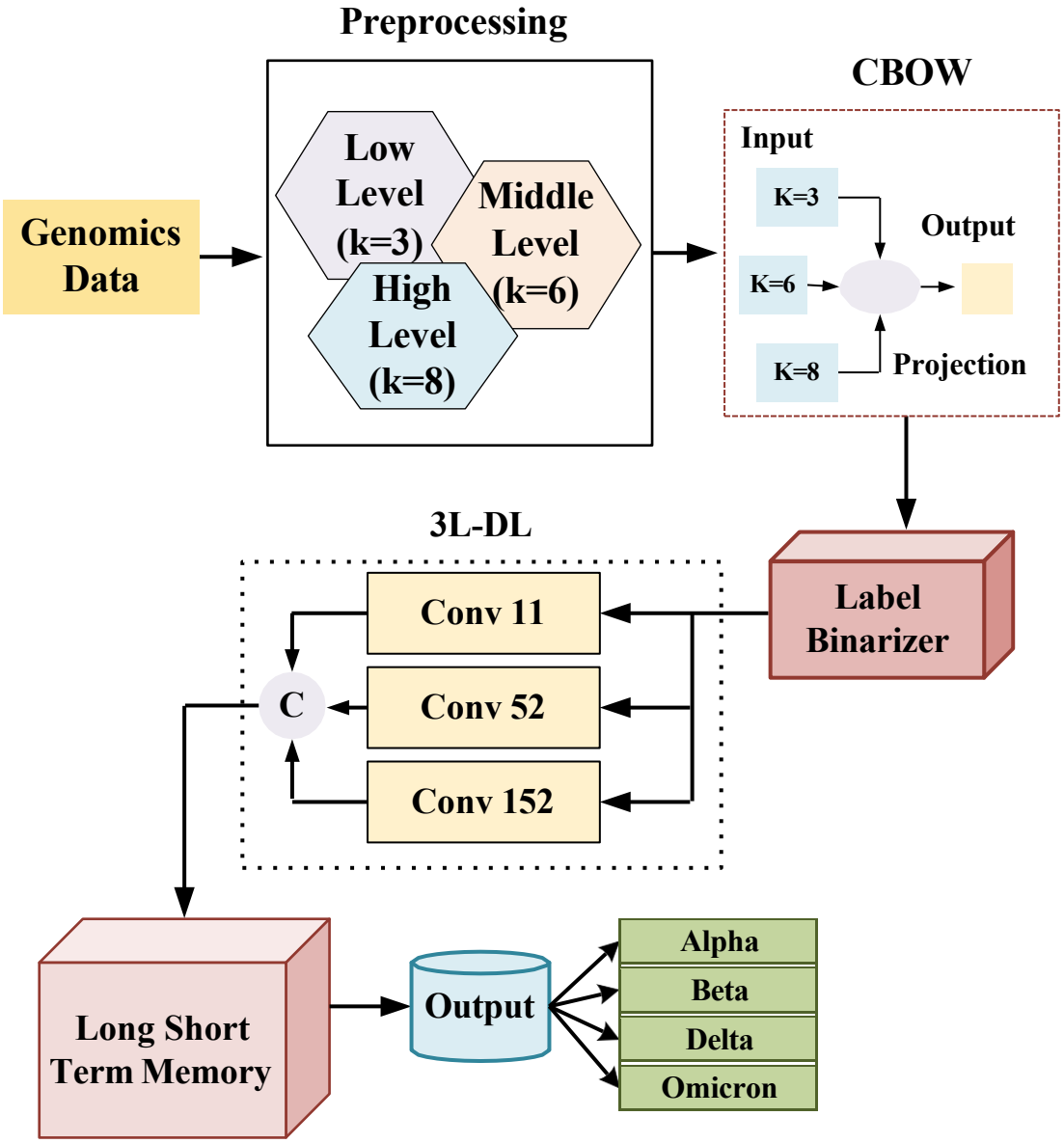


Figure 1: The general structure of the suggested approach

3.1 Data Collection

Data was obtained from https://www.ncbi.nlm.nih.gov/. It consists of genomic sequences for all four important SARS-CoV-2 variants. In each text file there are genomic sequences which are unique for each individual variant, therefore it is possible to use this data to analyze mutations and perform epidemiological studies. Omicron variation is characterized by high transmissibility, immune escape and low severity, Delta variant has high transmissibility and severity, Beta variant has mutations that provide partial immune escape and Alpha variant is associated with high transmissibility. The sequences are provided in text files, such as FASTA and could be used for further research.

3.2. Preprocessing

Preprocessing converts raw data to clean and structured data to make feature extraction and performance of the model effective. The methods used in preprocessing include K-mers, CBOW, and Label Binarize for representing data.

K-Mers

K-mers translate input sequences, and the translations are made in overlapping substrings that mimic English words in a sequential segmentation process. Length of K-mers has been classified into three categories, namely medium (k=6), low (k=3), and high (k=8), which represent sequence characteristics at different levels of granularity.

Example:	ATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAACACA

Low-level K-Mers ($K = 3$)

Low-level K-mers ($K = 3$) split a sequence into overlapping substrings, each of length three, using a stride-1 sequential segmentation approach. This captures short, local patterns in the sequence.

Example:	ATG, GAG, AGA, GAA, AAA, AAC, ACA, CAC, and ACA

Medium Level K-Mers ($K = 6$)

Medium-length K-mers with K=6 divide the sequence into overlapping substrings using a stride-1 sequential segmentation approach. This length is effective in capturing mid-level dependencies and structural patterns within the sequence.

Example:	ATGGAG, TGGAGA, GGAGAG, GAGAGC, AGAGCC, GAGCCT, GCCTTG, AACGAG, ACGAGA, CGAGAA, GAGAAA, AGAAAA, GAAAAC, AAAACA, AAACAC, and AACACA

High Level K-Mers ($K = 8$)

Higher-order K-mers ($K = 8$) capture eight-base substrings overlapping with a sequence from a stride-1 sequential segmentation approach. These capture long-range dependencies and large-scale contextual patterns.

Example:	ATGGAGAG, TGGAGAGC, GGAGAGCC, GAGAGCCT, AGAGCCTT, CGAGAAAA, GAGAAAAC, AGAAAAC, GAAAACAC, and AAAACACA

Continuous Bag of Words (CBOW)

In Natural Language Processing, the Continuous Bag of Words (CBOW) algorithm is based on neural networks and creates dense vector representations that encode contextual relations among words. The algorithm tries to predict the target word using the surrounding context inside a window. Embeddings produced by the CBOW algorithm can be used for different applications like text classification. Figure 2 shows the CBOW architecture.

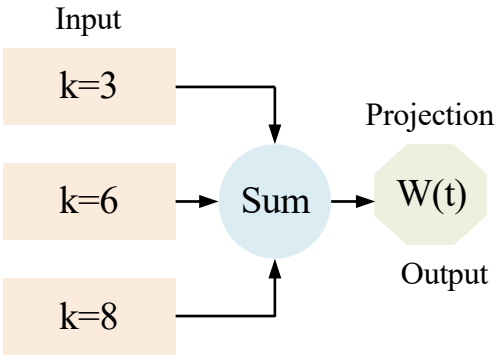


Figure 2: CBOW architecture

Label binarizes

Label Binarize is a preprocessing application that converts class labels to a binary (one-hot) format. It converts category or sequential data to a numeric format by encapsulating each unique label as a binary vector with one bit set to 1 (indicating the presence of a given class) and the rest set to 0. This transformation enables machine learning algorithms to efficiently handle and learn from categorical inputs, particularly in classification challenges. Label Binarize is described in Table 1, 2 and 3 demonstrates the label binarize with k=3, k=6, and k=8.

Table 1 (a): Input for Label Binarize k=3

Label	kmer_1	kmer_2	kmer_3
Alpha	ATG	TTC	TGT
Alpha	ATG	GAG	TGG

The input sequences for the Alpha variant with k = 3 are shown in Table 1(a). The DNA sequences have been transformed into overlapping 3-mers. The k-mers enable the acquisition of short-range nucleotide patterns for the purpose of encoding and classification.

Table 1 (b): Output for Label Binarize k=3

Label	kmer_1_ATG	kmer_2_TTC	kmer_2_GAG	kmer_3_TGT	kmer_3_TGG
Alpha	1	1	0	1	0
Alpha	1	0	1	0	1

Table 1(b) illustrates the binarized output for the Alpha variant with k=3, where features for each unique k-mer have been binarized. A value labeled '1' means that the corresponding k-mer is present in the corresponding sequence and allows for efficient input encoding for model training.

Table 2 (a): Input for Label Binarize k=6

Label	kmer_1	kmer_2	kmer_3
Alpha	ATGTTC	TGTTCG	GTTCGT
Alpha	ATGGAG	TGGAGA	GGAGAG

In Table 2(a), DNA sequences from the Alpha variant are divided into overlapping k-mers of size 6. Each row is a sequence split along three k-mers in genomic order and with respect to the context of that sequence when training a model and recording labels.

Table 2 (b): Output for Label Binarize k=6

Label	kmer_1_ ATGTTC	kmer_1_ ATGGAG	kmer_2_ TGTTCG	kmer_2_ TGGAGA	kmer_3_ GTTCGT	kmer_3_ GGAGAG
Alpha	1	0	1	0	1	0
Alpha	0	1	0	1	0	1

The binary label encoding for the k=6 k-mers of the Alpha variant (Table 2(b)) lists a unique binary column for each k-mer, with a presence denoted by '1'. This format reads the input data ready to use for training the classification model.

Table 3 (a): Input for Label Binarize k=8

Label	kmer_1	kmer_2	kmer_3
Alpha	ATGTTCGT	TGTTCGTG	GTTCGTGT
Alpha	ATGGAGAG	TGGAGAGC	GGAGAGCC

Table 3(a) shows the input sequences for Label Binarization with k=8 for the Alpha version. Each sequence is broken down into overlapping k-mers that have a size of 8, thus giving nucleotide dependencies at a longer range.

Table 3 (b): Input for Label Binarize k=8

Label	kmer_1_ ATGTTCGT	kmer_1_ ATGGAGAG	kmer_2_ TGTTCGTG	kmer_2_ TGGAGAGC	kmer_3_ GTTCGTGT	kmer_3_ GGAGAGCC
Alpha	1	0	1	0	1	0
Alpha	0	1	0	1	0	1

Table 3(b) provides the binary-encoded output of the k=8 k-mer sequences for the Alpha variant via a process called Label Binarization. Each unique k-mer gets an allocation in the output, and values of either zero or one determine the non-appearance or appearance, respectively, of each k-mer in the order of the output binary

vector. This makes multi-class classification easy to perform.

3.3. Three-Level deep learning-based feature learning (3L-DL)

Hierarchical patterns in COVID-19 genomic sequences are intended to be detected by Three-Level Deep Learning-Based Feature Learning. The model takes into account the global and local dependency in sequence by combining the low, mid, and high-level convolution layers along with RNNs at all levels. The multi-layer structure allows the extraction of holistic features which improves the model's ability to classify and detect complex structures in the genome. The 3L-DL with SA_LSTM is presented in Figure 3.

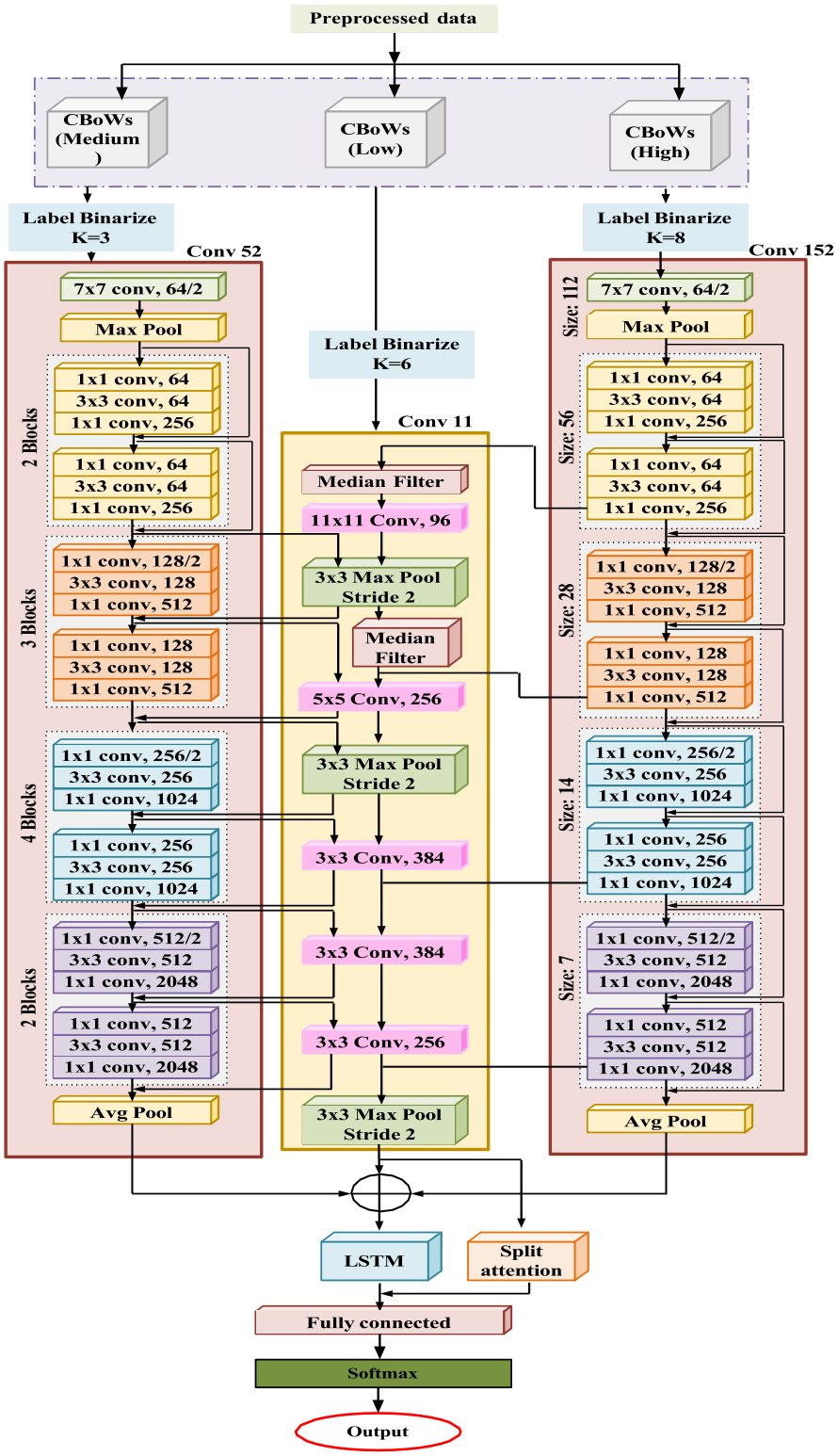


Figure 3: Framework of 3L_DL with SA_LSTM

Low-level feature learning

This layer uses Conv11 and RNN models for modeling the short-range dependencies between the sequences. In this, low-dimensional CBOW is used for representing local k-mers, whereas, median filtering is done to reduce the noise level. In Conv11 layer, 11×11 (96 filters), 5×5 (256 filters), and 3×3 (384/256 filters) convolutions are used to derive features for local and larger sequence lengths. Stride-2 max-pooling is used for downsampling the maps, and the derived features are given as input to RNN.

Middle Level Feature Learning

In this layer, Conv52 and RNN have been used to detect mid-level intermediate-range patterns in genome sequences. The medium resolution CBOW embeddings are used to understand the motifs and conservation of the sequence of nucleotides in the multiple k-mers. In this Conv52 network, the inspiration is taken from ResNet-50. This network starts from 7 × 7 convolution followed by max-pooling. The next part involves four stages of residual block having filters (1 × 1,3 × 3,1 × 1) in the numbers of filters 64 to 2048. Average pooling is used to reduce the size of feature map, and skip connections keep computations efficient.

High-Level Feature Learning

In this layer, Conv152 and an RNN are used to extract long-range and high-level patterns from the genomic sequences. The high resolution CBOW representations will help the model to learn about broader k-mer contexts which include complex structure dependencies. In terms of Conv152, the ResNet-152 architecture is applied to begin with a max pooling and 7x7 convolution (64 filters, stride 2), followed by 4 residual blocks stages (3,8,36,3), having 1×1,3×3,1×1 convolutions, and increasing the number of feature maps to 2048 filters. The extracted features will be passed through an RNN to maintain the time dependency and learn about long-range dependencies within the sequence.

The outputs of all the three layers that extract the low-level, medium, and high-

level features respectively are merged to form a single feature vector. This way the model will learn different spatial and context features of the genome sequence.

3.4. Self-Attention Long Short-Term Memory (SA_LSTM)

The sequential nature of genomic data is captured using an LSTM network as it is important that the order of the biological sequences (A, T, C, G) must be maintained. The feature maps obtained from the Conv11, Conv52, and Conv152 layers are concatenated together to make a time series dataset, through which the long-term dependencies of the genome are learned by the LSTM. This helps in capturing the variant specific signatures such as mutation and insertion of genetic data, which might be in a non-linear fashion. In order to boost the representation, we use self-attention using parallel average and max pooling and then shared MLP to create attention-based features.

$$\mathcal{R}_T = \sigma(\omega_{\mathcal{R}} * [hid_{T-1}, inp_T]) \quad (1)$$

$$\mathcal{U}_T = \sigma(\omega_{\mathcal{U}} * [hid_{T-1}, inp_T]) \quad (2)$$

$$\mathbb{C}hid_T = tanh(\omega_{hid} * [\mathcal{R}_T,* hid_{T-1}, inp_T]) \quad (3)$$

$$hid_T = (1 - \mathcal{U}_T)hid_{T-1} + \mathcal{U}_T\mathbb{C}hid_T \quad (4)$$

The weight matrices are hid_T, $\omega_{\mathcal{U}}$, and ω_{hid}. The reset gate ($\mathcal{R}_T$) in Eq. (1), the update gate ($\mathcal{U}_T$) in Eq. (2), the candidate hidden state ($\mathbb{C}hid_T$) in Eq. (3), and the hidden state (4). The previous hidden state is represented by hid_{T-1}, and the current concealed state is represented by hid_T, inp_T is the input vector at time step T, σ is the sigmoid activation function, $tanh$ is the hyperbolic tangent activation function, $*$ is element-wise multiplication, and $[\cdot]$ denotes vector concatenation.

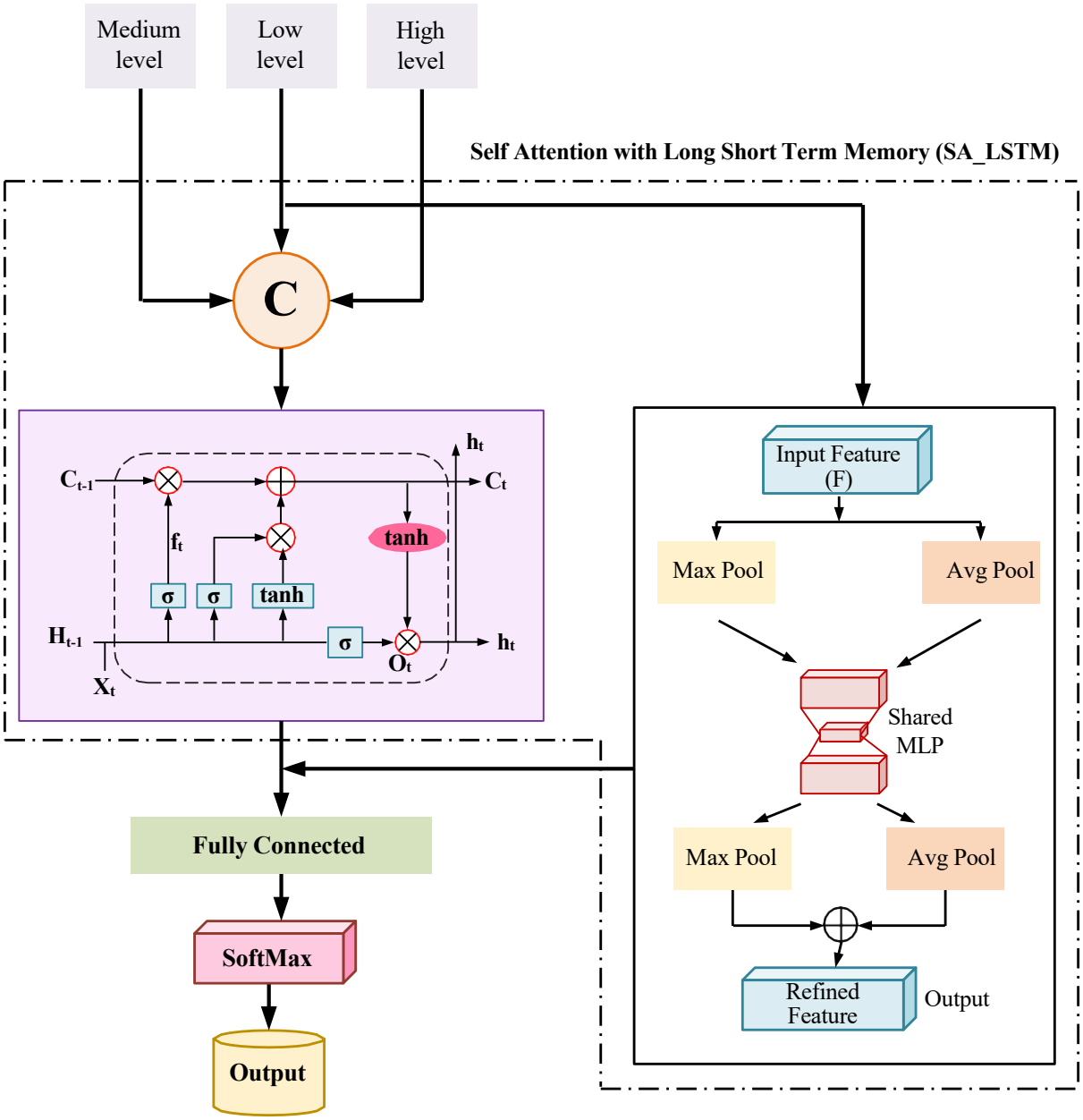


Figure 4: Framework of SA_LSTM

Figure 4 shows the SA_LSTM framework. After attention refinement, the final fully connected layer maps the learned, attention-weighted features to output neurons, with SoftMax converting scores into probabilities for COVID-19 variant classification. The SA_LSTM model captures the dependencies and hierarchical feature representation, while the attention mechanism highlights the biological significance in the genome. This step-wise method guarantees high classification

accuracy in analyzing the high dimensional

genome of the COVID-19 virus.

Algorithm 1: Pseudocode for our proposed work

BEGIN

Step 1*: Data Collection*

Load genomic sequences for COVID-19 variants from NCBI

Step 2: *Preprocessing - K-mer Segmentation*

FOR each sequence IN genomic_data:

low_level_kmers ← extract_kmers (sequence, k=3, stride=1)

mid_level_kmers ← extract_kmers (sequence, k=6, stride=1)

high_level_kmers ← extract_kmers (sequence, k=8, stride=1)

combined_kmers ← concatenate (low_level_kmers, mid_level_kmers, high_level_kmers)

Step 3*: CBoW Embedding*

embedded_kmers ← CBOW (combined_kmers)

Step 4: *Label Binarization*

encoded_labels ← LabelBinarizer(variant_labels)

Step 5: *Feature Extraction using Three-Level Deep Learning*

conv_low ← ConvLayer11(embedded_kmers)

conv_mid ← ConvLayer52 (embedded_kmers)

conv_high ← ConvLayer152 (embedded_kmers)

Step 6: *Feature Fusion*

fused_features ← concatenate (conv_low, conv_mid, conv_high)

Step 7: *Temporal Sequence Learning with SA_LSTM*

lstm_output ← SA_LSTM (fused_features)

Step 8: *Classification Output*

final_output ← Softmax (Dense(sa_lstm_output))

Step 9: *Variant Prediction*

predicted_variant ← argmax (final_output)

RETURN predicted_variant

END

4. RESULTS AND DISCUSSIONS

The performance of the suggested model was evaluated through the application of several metrics, including Matthews Correlation Coefficient (MCC), Sensitivity, False Negative Rate (FNR), Accuracy, F1 score, Mean Absolute Error (MAE) and Mean Square Error (MSE), False Positive Rate (FPR), Specificity and so on. In order to assess the efficiency of the recommended architecture, the performance of the suggested model was compared with that of the other architectures, including DT, XGBoost, KNN, and RF.

4.1. System Configurations

4.1.1. Hardware Requirements

The ideal hardware for deep learning and security studies will include either the AMD Ryzen 5 or the Intel Core i5 (10th Gen). In case one wishes to use deep learning technology, then an NVIDIA GTX 1650 graphics card which has 4 GB of VRAM is essential in enhancing processing and training of the models. The computer ought to have at least 16 GB of RAM so as to facilitate easy handling of large data and complex calculations. The ideal storage space will be 256 GB SSD.

4.1.2. Software Requirements

The suggested software architecture uses Windows 10 (64-bit), which can work with most of the development environments. The version 3.11 of the Python programming language must be used together with Anaconda that will make it

easier to manage dependencies and packages. The Anaconda tool will help to easily manage packages and dependencies, and that is why it can be used effectively in such areas as deep learning and cybersecurity. Additionally, Jupyter Notebook must be installed to have an environment where it is easy to test and visualize the code. This configuration provides a productive and adaptable workspace for data analysis and model development.

4.2. Performance Evaluation

Performance evaluation uses a variety of metrics, such as sensitivity, MAE, FPR, accuracy, MSE, Specificity, FNR, MCC, and F1 Score.

- ***Accuracy:*** It refers to how closely a quantity's measurements match its actual value.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (5)$$

- ***Precision:*** It describes the total number of actual samples that were appropriately taken into account during the classification process.

$$Precision = \frac{TP}{FP+TP} \quad (6)$$

- ***F-Measure:*** The demand that each definition defines a specific type of information item is carefully balanced with the need to fully identify every piece of data.

$$F - Measure = 2 \times \frac{Presision.Recall}{Presision + Recall} \quad (7)$$

- ***Specificity:*** The number of negatively expected outcomes among all correctly predicted bad occurrences is a reliable indicator of specificity.

$$Specificity = \frac{TN}{FP+TN} \quad (8)$$

- ***Sensitivity:*** It is computed by dividing the percentage of accurate positive forecasts by the total number of positive forecasts.

$$Sensitivity = \frac{TP}{TP+FN} \quad (9)$$

- ***MCC:*** Since it considers FN, TP, FP, and TN, it is a valid statistic for assessing the performance of binary classifiers.

$$MCC = \frac{(TP*TN)-(FP*FN)}{\sqrt{(TP+FP)(TP+FN)(FP+TN)(TN+FN)}} \quad (10)$$

- ***NPV:*** It is a statistic used to evaluate the performance of a binary classification model. Negative projected values indicate the proportion of negative projections that materialize.

$$NPV = \frac{TN}{TN+FN} \quad (11)$$

- ***FNR:*** It is defined as the percentage of cases that are incorrectly classified and inadvertently assigned a negative label relative to the total number of cases that are classified positively.

$$FNR = \frac{FN}{FN+TP} \quad (12)$$

- ***FPR:*** It can be expressed by dividing total amount of negative data by percentage of correctly classified positive data.

$$FPR = \frac{FP}{TN+FP} \quad (13)$$

- ***G-Mean:*** The geometric mean (G-Mean), as indicated in Eq. (14), is used when both TP and TN performance indicators are significant. It acts as a barometer for the classifier's performance in both classes.

$$G - mean = \sqrt{TP(1-FP)} \quad (14)$$

Here, False Positive FP, False Negative FN, True Negative TN, and True Positive TP. Figure 5 shows Confusion matrix.

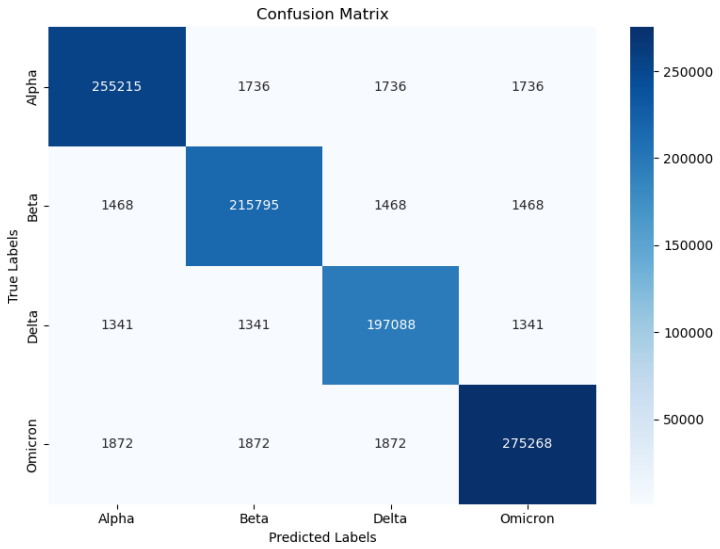


Figure 5: Confusion matrix

4.3. Performance analysis

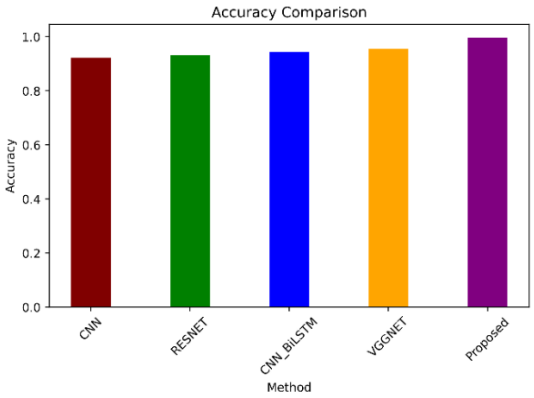
The suggested approach is contrasted with CNN, ResNet, CNN_BiLSTM, and VGGNet with a variety of performance metrics. The performance comparison is shown in the Table 3. The proposed method outperformed all classifiers regarding accuracy, sensitivity, specificity, precision, F-measure, and other measures.

Table 3: A comparison between the suggested and current works

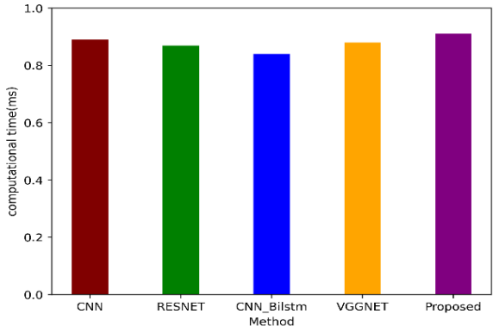
Metrices	CNN	RESNET	CNN_BiLSTM	VGGNET	Proposed
Accuracy	0.9212	0.9323	0.9433	0.9543	0.99569
Precision	0.9143	0.9212	0.9321	0.94212	0.98765
Sensitivity	0.9143	0.9433	0.9343	0.9417	0.98776
Specificity	0.9221	0.9421	0.9234	0.9432	0.99567
F-Measure	0.9132	0.9344	0.9321	0.94213	0.98546
MCC	0.9345	0.9311	0.9345	0.9443	0.98212
NPV	0.9321	0.9245	0.9433	0.9432	0.99213
FPR	0.095	0.11	0.124	0.153	0.0253

FNR	0.091	0.134	0.183	0.193	0.0343
Inference time(ms)	26	32	21	34	12
GMean	0.8921	0.9067	0.9212	0.9323	0.9721

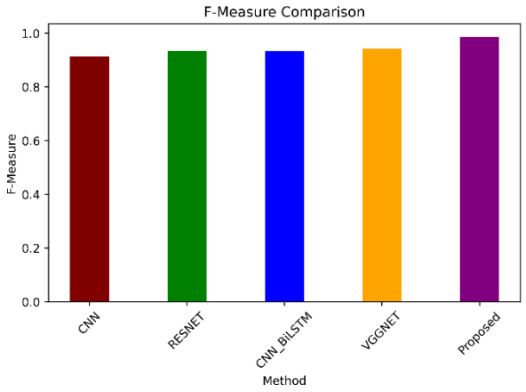
Table 3 provides a detailed A comparison of the proposed COVID-19 variant classification model relative to several existing deep learning architectures (CNN, ResNet, CNN_BiLSTM, VGGNet) based on a variety of performance metrics. The proposed model outperforms all baseline models on all performance measures. For example, Regarding accuracy, the suggested model's accuracy is 0.99569, which exceeds VGGNet (0.9543), CNN_BiLSTM (0.9433), ResNet (0.9323), and CNN (0.9212). Precision is equal to 0.98765 and sensitivity 0.98776 for the proposed model shows a dominant number of true positive predictions. The specificity is also quite high at 0.99567 and indicates the strength of the model in classifying negatives correctly. F-Measure and MCC are also highest with respect to the proposed model and are 0.98546 and 0.98212, respectively, and reiterate its high overall performance and balanced classification strength. In addition, the suggested model achieves the largest NPV of 0.99213, and the smallest FPR and FNR of 0.0253 and 0.0343, respectively, which highlights its accuracy in reducing misclassifications. Importantly, the inference time of the suggested model is the shortest at only 12 milliseconds, whereas other models take 21–34 milliseconds, which indicates its efficiency. Finally, the GMean of 0.9721 also confirms how well the model handles data that is unbalanced. In conclusion, these findings show that, in contrast with the most advanced models, the suggested method not only achieves superior classification performance but also enables a quicker and more accurate COVID-19 variant detection.



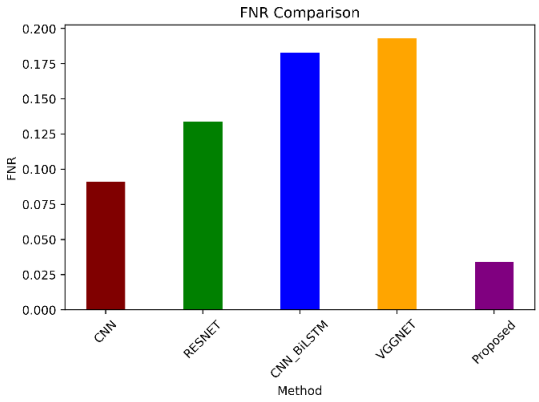
a) Accuracy



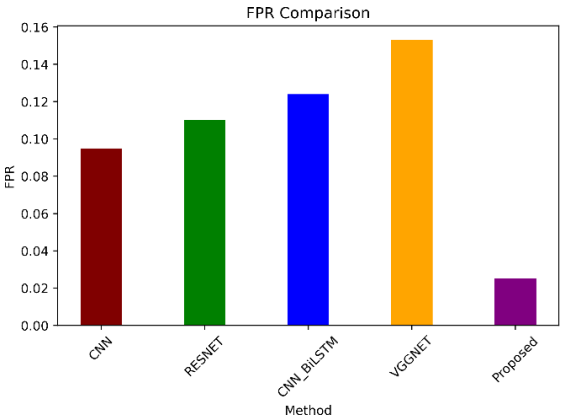
b) Computation time



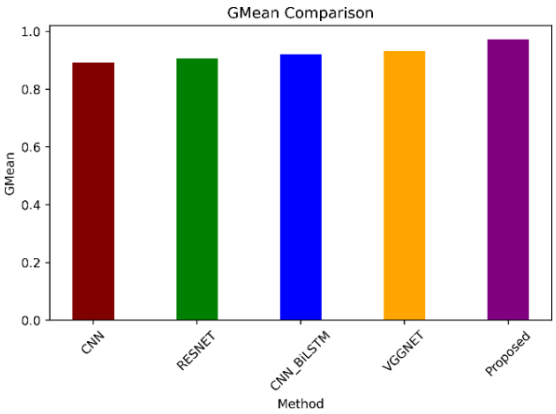
c) F-Measure



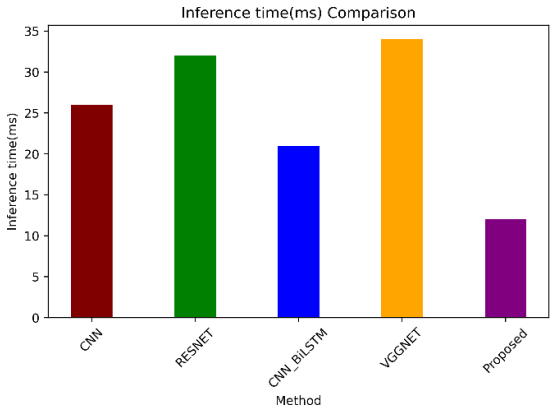
d) FNR



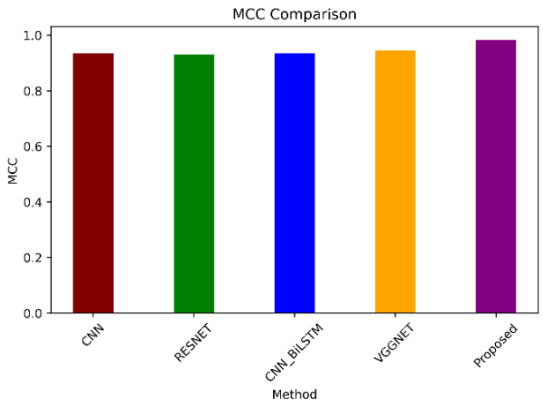
e) FPR



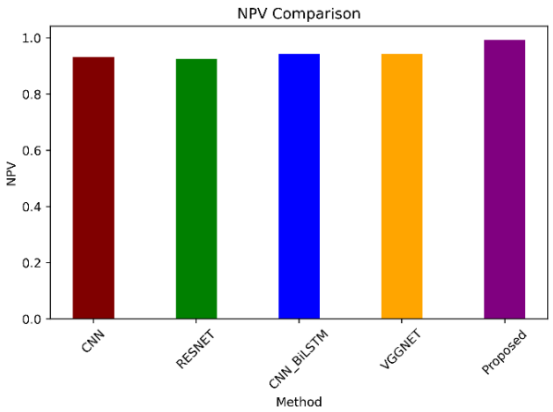
f) G-Mean



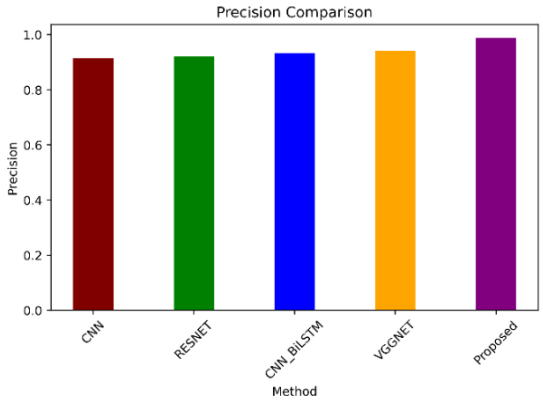
g) Inference time (ms)



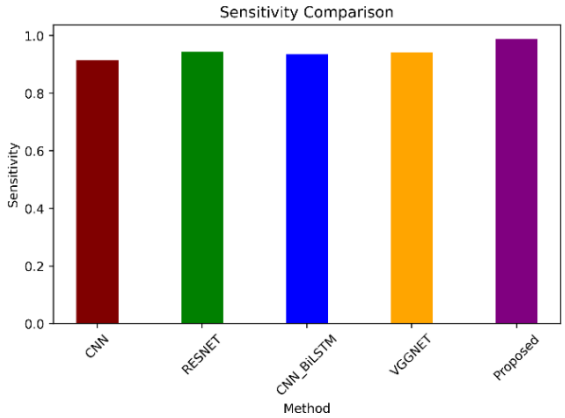
h) MCC



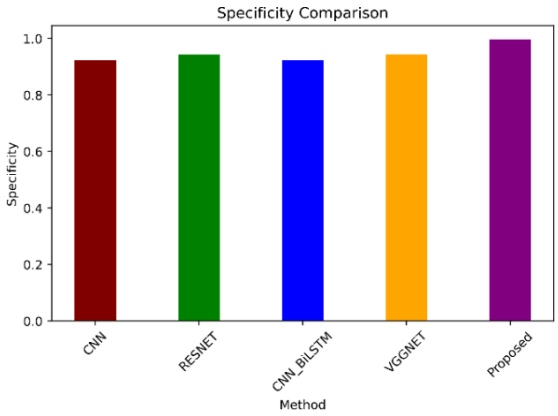
i) NPV



j) Precision



k) Sensitivity



l) Specificity

Figure 6 (a-l): Visual representation of Accuracy, Computation time, F-Measure, FNR, FPR, G-Mean, Inference time (ms), MCC, NPV, Precision, Sensitivity, and Specificity

The performance measures that have been

used for comparison include Accuracy, Computation time, F-Measure, FNR, FPR, G-Mean, Inference time (ms), MCC, NPV, Precision, Sensitivity, and Specificity. They have been shown through graphs as Figures 6 (a) to (l) with various methods, such as CNN, ResNet, and CNN_BiLSTM

4.4. Performance of Ablation Study

A study on ablation was performed to measure the contribution of the main architectural elements by removing the attention layers and LSTM units from the proposed model. This enables a clearer assessment of the impact of each module on classification performance. The results demonstrate the contribution of each module to overall model performance and robustness. Table 4 shown the performance of the ablation study

Table 4: Performance analysis for the ablation study

Metrices	without Attention layers	without LSTM	Proposed
Accuracy	0.91	0.92	0.99569
Precision	0.9	0.9	0.98765
Sensitivity	0.89	0.91	0.98776
Specificity	0.88	0.92	0.99567
F-Measure	0.91	0.91	0.98546
MCC	0.91	0.92	0.98212
NPV	0.9	0.91	0.99213
FPR	0.09	0.08	0.0253
FNR	0.1	0.09	0.0343
Inference time(ms)	18	14	12
GMean	0.9	0.91	0.9721

The ablation study results in summary in Table 4 highlight the impact of the critical components in the model proposed in the previous sections. Without the attention layers and without the LSTM, the full proposed model performs much worse across all metrics when compared with the full proposed framework. The recommended model exhibits the greatest amounts of specificity (0.99567), sensitivity (0.98776), accuracy (0.99569), and precision (0.98765), as both attention mechanisms and LSTM layers play a significant role in the model. Additionally, it achieves the worst FPR (0.0253) and FNR (0.0343) along with best GMean (0.9721) and best inference time (12 ms), demonstrating that the full architecture demonstrates efficacy and efficiency.

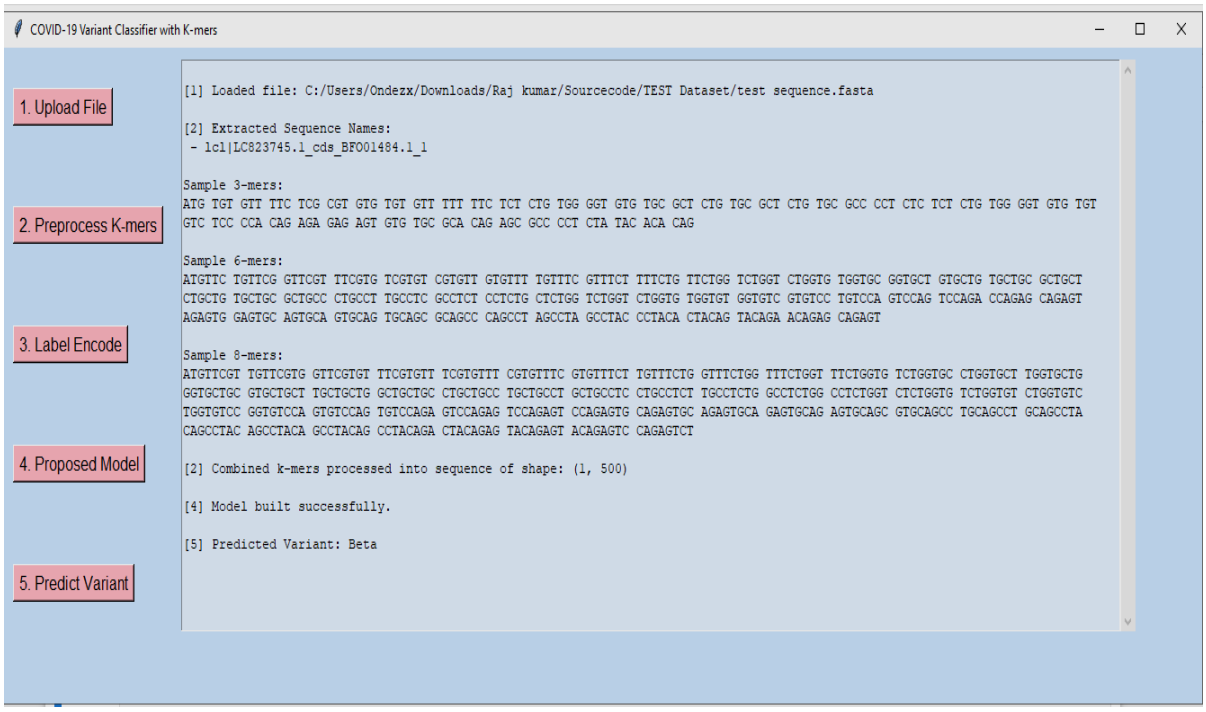


Figure 7: Visual representation of the Predicted image

The predicted image that is illustrated in Figure 7 was the final output from the processes performed and these include: inputting, opening the file and sequential extraction. This process enables the model to have its input as the genomics and to generate its output in the form of a valid classification.

5. CONCLUSION

A new three-layer hierarchical deep learning framework along with self-attention-based LSTM was designed to classify variants of SARS-CoV-2 based on their genomes. The model uses multi-level K-mer encoding of raw genome data, and the representation is embedded using CBOW, to identify local nucleotide patterns as well as contextual information from longer sequences of the genome. In addition, hierarchical convolutional-recurrent architecture is used to perform multi-scale feature extraction and self-attention-based mechanism is employed to highlight biological patterns of the genome. Experimental studies show that the proposed method outperforms the existing deep learning models in inference efficiency and performance metrics including accuracy, F1-Score, and MCC. The results clearly demonstrate the efficacy of hierarchical learning of representations and attention-based temporal modeling in dealing with genomic data. The proposed model is a promising approach that is scalable and interpretable for COVID-19 variants' monitoring in real time and can easily be adapted for other virus genomics studies and emerging infectious diseases monitoring.

DECLARATIONS:

DATA AVAILABILITY STATEMENT

All the data is collected from the simulation reports of the software and tools used by the authors. Authors are working on implementing the same using real world data with appropriate permissions.

FUNDING

No fund received for this project

CONFLICTS OF INTEREST

The authors declare that they have no conflict of interest.

ETHICAL APPROVAL AND HUMAN PARTICIPATION

No ethics approval is required.

REFERENCES

[1] Singh, P., Anand, A., Rana, S., Kumar, A., Goel, P., Kumar, S., ... & Singh, H. 2024. Impact of COVID-19 vaccination: a global perspective. *Frontiers in public health*, *11*, 1272961.

[2] Li, R., Song, Y., Qu, H., Li, M. and Jiang, G.P., 2024. A data-driven epidemic model with human mobility and vaccination protection for COVID-19 prediction. *Journal of Biomedical Informatics*, *149*, p.104571.

[3] Cheng, C., Aruchunan, E. and Noor Aziz, M.H., 2025. Leveraging dynamics informed neural networks for predictive modeling of COVID-19 spread: a hybrid SEIRV-DNNs approach. *Scientific Reports*, *15*(1), p.2043.

[4] Qayyum, A., Benzinou, A., Saidani, O., Alhayan, F., Khan, M.A., Masood, A. and Mazher, M., 2024. Assessment and classification of COVID-19 DNA sequence using pairwise features concatenation from multi-transformer and deep features with machine learning models. *SLAS technology*, *29*(4), p.100147.

[5] Bae, H.J., Kim, J. and Jeong, D., 2026. Real-Time Fluorescence-Based COVID-19 Diagnosis Using a Lightweight Deep Learning System. *Sensors*, *26*(1), p.339.

[6] Tabassum, T., Talukder, M.A., Rahman, M.M., Rashiduzzaman, M., Kabir, M.Z., Islam, M.M. and Uddin, M.A., 2025. A Parallel Convolutional Neural Network for Accurate Face Mask Detection in the Fight Against COVID-19. *Biomedical Materials & Devices*, pp.1-11.

[7] Li, X., Li, J., Liu, H., Mínguez-Alarcón, L., van Loosdrecht, M.C. and Wang, Q., 2025. Lifting of travel restrictions brings additional noise in COVID-19 surveillance through wastewater-based epidemiology in post-pandemic period. *Water Research*, *274*, p.123114.

[8] Walker, D.M., Smith, W.A., Gale, L., Wolff, J.T., Healy, C.P., Van Hollebeke, H.F., Stephenson, A. and Kim, M., 2025. CoMIT: a bioinformatic pipeline for risk-based prediction of COVID-19 test inclusivity. BMC bioinformatics, 26(1), p.51.

[9] Zheng, X., Li, J., Ma, Q., Gong, J. and Pan, J., 2025. Integrative analyses of mendelian randomization and bioinformatics reveal casual relationship and genetic links between COVID-19 and knee osteoarthritis. *BMC Medical Genomics*, *18*(1), p.2.

[10] Nawi, S.F., Tajudin, N., Wang, S.M. and Mohamad, M., 2025. Detection of COVID-19 Outbreaks with Whole Genome Sequencing by Oxford Nanopore Technology in a Tertiary Centre, Malaysia. *International Journal of Infectious Diseases*, *152*, p.107452.

[11] Kathiravan, M., Shanthi, S., Alqahtani, M.S., Abbas, M. and Ayalew, F., 2026. Hybrid feature selection with novel deep learning model for COVID-19 risk prediction. *Scientific Reports*.

[12] Mutawa, A.M., 2025. Attention-based hybrid deep learning models for classifying COVID-19 genome sequences. *AI*, *6*(1), p.4.

[13] Nkengue, M.J., Zeng, X., Koehl, L. and Tao, X., 2024. X-RCRNet: An explainable deep-learning network for COVID-19 detection using ECG beat signals. *Biomedical Signal Processing and Control*, *87*, p.105424.

[14] Baihaqi, A.D., Yudistira, N. and Santoso, E., 2024. An investigation of multivariate data-driven deep learning models for predicting COVID-19 variants. *Healthcare Analytics*, *5*, p.100331.

[15] Henna, S., Alcaraz, J.M.L., Rathnayake, U. and Amjath, M.,

2025. An interpretable deep learning framework for medical diagnosis using spectrogram analysis. *Healthcare Analytics*, *8*, p.100408.

[16] Matson, R.P., Comba, I.Y., Silvert, E., Niesen, M.J., Murugadoss, K., Patwardhan, D., Suratekar, R., Goel, E.G., Poelaert, B.J., Wan, K.K. and Brimacombe, K.R., 2024. A deep learning approach predicting the activity of COVID-19 therapeutics and vaccines against emerging variants. *NPJ Systems Biology and Applications*, *10*(1), p.138.

[17] Talukder, M.A., Layek, M.A., Kazi, M., Uddin, M.A. and Aryal, S., 2024. Empowering covid-19 detection: Optimizing performance through fine-tuned efficientnet deep learning architecture. *Computers in Biology and Medicine*, *168*, p.107789.

[18] Fekri-Ershad, S. and Dehkordi, K.B., 2025. A Flexible Multi-Channel Deep Network Leveraging Texture and Spatial Features for Diagnosing New COVID-19 Variants in Lung CT Scans. *Tomography*, *11*(9), p.99.

[19] Hussain, S., Amran, G.A., Alabrah, A., Alkhalil, L. and Al-Bakhrani, A.A., 2024. C19-MLE: A Multi-Layer Ensemble Deep Learning Approach for COVID-19 Detection Using Cough Sounds and X-ray Imaging. *IEEE Access*, *12*, pp.197151-197167.

[20] Hu, B., Han, Y., Zhang, W., Zhang, Q., Gu, W., Bi, J., Chen, B. and Xiao, L., 2024. A prediction approach to COVID-19 time series with LSTM integrated attention mechanism and transfer learning. *BMC Medical Research Methodology*, *24*(1), p.323.

[21] Azarfar, G., Sun, Y., Pasini, E., Sidhu, A., Brudno, M., Humar, A., Kumar, D., Bhat, M., Ferreira, V.H., Delisle, J.S. and Hosseini-Moghaddam, S., 2025. Using machine learning for personalized prediction of longitudinal coronavirus disease

2019 vaccine responses in transplant recipients. *American Journal of Transplantation*, *25*(5), pp.1107-1116.

[22] Mekouar, S., 2025. Enhanced LSTM model with integrated attention mechanism and data augmentation for projecting COVID-19 trends in Africa. *Scientific African*, *28*, p.e02617.

[23] Fahim, A., Osman, A.M., Tarek, Z. and Elshewey, A.M., 2025. A hybrid deep learning framework based on CNN-GRU-TabNet for the predictive modeling of COVID-19 mortality. *Engineering, Technology & Applied Science Research*, *15*(5), pp.28057-28062.

[24] Antunes, C., Rodrigues, J. and Cunha, A., 2025. CTCOVID19: automatic COVID-19 model for computed tomography scans using deep learning. *Intelligence-Based Medicine*, *11*, p.100190.

[25] Dutta, S.S., Sandeep, S., Sridevi, S., Mistry, R., Ortega, J. and Kar, G., 2025. Novel Quanvolutional Neural Network based COVID-19 Diagnosis from Raman Spectroscopic Signals of Human Serum. *IEEE Access*.