Triplex-Forming Potential Prediction in Long Non-coding RNAs Using Convolutional and Long Short Term Memory Networks

Renee Manohara Bethapudy
Department of Information Technology
BVRIT HYDERABAD College of Engineering for Women
Hyderabad, India
22wh1a12a2@bvrithyderabad.edu.in

***Abstract*—Long non-coding RNAs (lncRNAs), longer than 200 nucleotides, play key roles in gene regulation by forming DNA:RNA triplexes through Hoogsteen base pairing with purine-rich regions of dsDNA. Experimental methods are expensive and condition sensitive, highlighting the need for computational pre-screening. This work proposes a multiscale deep learning model combining 1D-CNN and LSTM to classify triplex-forming lncRNAs, capturing both local and long-range dependencies. The model is trained on a benchmark dataset and evaluated using accuracy, sensitivity, specificity, AUROC, AUPRC, F1-score, and harmonic mean. With 5-fold cross-validation, it achieved 90.11% accuracy, 85.71% sensitivity, 93.88% specificity, 0.9689 AUROC, 0.9683 AUPRC, 0.8889 F1-score, and 0.8961 harmonic mean, demonstrating strong generalization.This approach provides a reliable computational framework for identifying lncRNAs with DNA:RNA triplex-forming capability, supporting further investigation into their regulatory and functional implications.**

***Index Terms*—Deep Learning, Prediction, Long Non-coding RNAs, Triplex-Forming Potential, CNN, LSTM**

I. Introduction

DNA and RNA can form a triple helix structure known as a triplex, which plays important roles in DNA repair, transcriptional regulation, and RNA metabolism [1]. These DNA:RNA triplexes arise when a single-stranded RNA binds to the purine-rich strand of duplex DNA via forward or reverse Hoogsteen base pairing rules [2]. Triplex forming oligonucleotides (TFOs) are RNA sequences capable of binding DNA to form such structures, while triplex target sites (TTSs) are the complementary regions in DNA that enable this interaction [3]. Experimental identification relies on biophysical techniques such as NMR spectroscopy [4], which detects imino-proton peaks under mildly acidic conditions, and circular dichroism (CD) spectroscopy [5], which reveals characteristic shifts near 270–280 nm. Electrophoretic mobility shift assays (EMSA) [6] further assess triplex formation and RNA–DNA binding kinetics. These methods, however, are time-consuming and require significant laboratory effort. RNA–DNA interaction data can be categorized by experimental scope. One-to-all methods such as ChIRP [7] and its variants identify genome-wide binding sites of a specific RNA, while all-to-all methods like RADICL-seq [8] and RedC [9] capture RNA–DNA interactions across the genome. Notably, Sentürk Cetin et al. [6] retrieved RNA and DNA sequences associated with triplex structures, but their method does not detect direct RNA–DNA base pairing. Triplex formation is governed by base pairing rules, broadly categorized into canonical and non-canonical types. Canonical rules typically involve purine-rich Hoogsteen interactions, while non-canonical pairing may include mismatches, bulges, or non-standard hydrogen bonding patterns. To address the challenges in triplex identification, computational tools have been developed based on different underlying principles. Some tools such as Triplexator [10] and Triplex Domain Finder [12] rely on canonical Hoogsteen pairing rules to identify triplex forming regions. Others like LongTarget [11] and Fasim LongTarget [14] incorporate non-canonical base pairing, expanding the scope of detectable triplex configurations. Tools such as 3plex [16] combine both canonical and non-canonical rules along with additional features like RNA secondary structure masking to improve predictive power. TriplexFPP [13] employs deep learning models trained on sequence data, while TriplexAligner [15] uses machine learning and statistical algorithms to predict DNA:RNA triplex formation. Unlike the traditional methods, these do not rely on predefined pairing rules, enabling flexible data-driven prediction of triplex interactions.

This work proposes a CNN-LSTM model to predict the global triplex forming potential of lncRNAs. This enables scalable screening of triplex forming lncRNAs evaluated using metrics such as accuracy, sensitivity, specificity, AUROC, AUPRC, F1 score, and harmonic mean.

II. Related Work

Identifying triplex structures remains a complex problem due to diverse base pairing patterns and context-dependent variability. This complexity is further compounded by the lack of strict formation rules and the presence of both canonical and non-canonical interactions. Computational tools address these challenges by leveraging predefined rules, heuristic extensions,

or data-driven models to predict triplex-forming regions with greater flexibility.

In [10], Triplexator predicts triplex-forming potential in TFOs and TTSs, and identifies matching TFO–TTS triplex pairs using the X-drop algorithm [19] with q-gram filtering. It uses ENCODE K562 chromatin-associated RNAs for TFO prediction and RefSeq gene annotations for TTS identification, achieving 90% confidence and statistically significant triplex predictions ($p \leq 10^{-4}$), though with nonlinear and computationally intensive runtime.

In [11], LongTarget ranks optimal TFOs within lncRNAs and identifies their corresponding target sites (TTSs) in DNA. It applies 24 base pairing rule sets, including non-canonical ones, and uses midpoint overlap clustering, evaluated on full exon lncRNA–DNA datasets. However, TFOs other than the best-ranked one are typically false positives.

In [12], TRIPLEXES improves Triplexator to detect RNA-DNA binding sites in lncRNAs. It uses Myers' bit-parallel algorithm [20] with a heuristic k-mismatch search. At 20% mismatch, it fares 1.86 times faster than Triplexator. However, it restricts mismatches based on a preset error rate, which may cause it to miss some potential predictions.

In the same study [12], Triplex Domain Finder (TDF) integrates TRIPLEXES predictions to statistically identify triplex-forming lncRNAs during cardiac differentiation. Using Fisher's Exact Test with FDR correction [21] on promoter and genomic regions, TDF analyzed 75 up-regulated lncRNAs, and reported significant enrichment ($p < 0.05$) with more than 80% precision among the top 15 ranked genes. The top candidate, GATA6-AS, showed an AUROC of 0.54. However, TDF is incompatible with protocols involving mixed RNA–DNA interaction types.

In [13], TriplexFPP employs a deep learning approach using a two-layer convolutional neural network with CD-HIT clustering to predict the triplex-forming potential of lncRNA and DNA sequences. Custom datasets were curated from Sentürk et al. (HeLa S3 cells), GENCODE, and Ensembl.The model achieved AUROC and AUPRC scores of 0.9649 and 0.9996 for lncRNA, and 0.8705 and 0.9671 for DNA. However, the evaluation is limited by the small number of positive samples and class imbalance in the dataset.

In [14], Fasim-LongTarget, an improved version of LongTarget, provides fast and accurate genome-wide lncRNA-DNA binding prediction by employing the Striped Smith-Waterman [22] and Waterman-Eggert algorithms with Huang-Miller improvements [23]. Tested on experimental data from MEG3, NEAT1, and MALAT1 variants, it achieved AUROC values of 0.74, 0.65, and 0.60, respectively. However, genome-wide application remains limited due to high computational time requirements.

In [15], TriplexAligner predicts RNA:DNA:DNA triplex formation using an expectation-maximization algorithm combined with Karlin–Altschul statistics [24]. Trained on triplexRNA-seq and triplexDNA-seq data from HeLa cells, it was evaluated on Red-C and RADICL-seq datasets, achieving AUROC scores exceeding 0.90. However, the method requires 11,270 runs to avoid local minima, shows variable performance across datasets, and disregards higher-order molecular structures by assuming linear molecules.

In [16], 3plex predicts triplex forming lncRNAs by identifying TFOs, TTSs and the binding potential of target regions. It is built on Triplexator combined with LongTarget for stability scoring and RNAplfold [25] for RNA secondary structure prediction to mask potential RNA regions. Using datasets from GEO, ENA, PubMed and GENCODE, 3plex was tested on 7 experimentally validated triplex forming lncRNAs achieving an AUROC of 0.660 ± 0.002 with a statistically significant ($p \leq 10^{-10}$) corrected for false discovery rate. However, 3plex does not consider non canonical pairing rules and limitations in RNA secondary structure prediction may affect overall accuracy.

In [17], a comparison of classical machine learning algorithms including SVM, kNN, Random Forest, Decision Tree, and XGBoost for TFO prediction was conducted using data from Kaufmann et al. [26] XGBoost outperformed others, achieving accuracy of 0.961, precision of 0.930, recall of 0.994, F1-score of 0.962, and AUROC of 0.961. However, the comparison was limited to only five models, and performance was dependent on computationally expensive grid search hyperparameter tuning.

In [18], Easy Triplex is presented as an online tool for DNA–RNA triple helix prediction using a sliding window approach under various error rates. Validated with lncRNA HOTAIR and promoter PCDH7, it predicted three TFOs and five corresponding TTSs. However, the tool is based solely on four canonical Hoogsteen rules and lacks an ideal evaluation framework for its predictions.

III. Proposed Method

A. Datasets

This study uses the benchmark dataset curated by Zhang et al. [13], comprising 531 triplex-forming and 36022 non-triplex-forming lncRNA sequences in FASTA format. However, to avoid data imbalance, 531 triplex-forming and 700 non-triplex-forming sequences were utilized. Each FASTA header follows a structured, pipe-separated format with fields including the Ensembl Transcript and Gene IDs, HAVANA gene and transcript IDs, isoform label, gene symbol, and transcript length. The sequences vary in length and consist of the canonical RNA bases adenine (A), guanine (G), cytosine (C), and uracil (U). However, in the provided FASTA files, uracil is already represented as thymine (T). This is because the FASTA format was originally designed for DNA sequences, and many bioinformatics tools expect nucleotides to be represented using the DNA alphabet: A, T, C, and G.

Fig. 1: A Typical FASTA Entry from the lncRNA Dataset

In Fig. 1, the Ensembl Transcript ID, ENST00000574016.5, uniquely identifies a specific transcript isoform, with annotation version .5, while the Ensembl Gene ID ENSG00000186594.14 specifies the associated gene with .14 version. Similarly, The HAVANA Gene ID OTTHUMG00000132197.7 and HAVANA Transcript ID OTTHUMT00000438379.2 represent manually curated gene and transcript identifiers by the GENCODE project. The transcript alias MIR22HG-207 identifies a particular splice variant of the gene, while MIR22HG is the standardized gene symbol. Lastly, the sequence length is indicated as 557 base pairs, reflecting the nucleotide count of the sequence.

B. Feature Extraction and Preprocessing

Adopting the feature extraction strategy of TriplexFPP [13], each sequence is represented as a 90-dimensional vector that integrates *k-mer* frequency and *kmerscore* features, capturing both compositional and statistical sequence characteristics [27], [28]. Input sequences are parsed from FASTA format using the SeqIO module [29] from Biopython.

(i) k-mer Frequency Features: For $k = 1$ to 3, the frequencies of all possible k-length nucleotide substrings (k-mers) are computed and normalized by the sequence length. The nucleotide alphabet is defined as $\Sigma = \{A, C, G, T\}$, and all possible combinations of nucleotides of length k are generated, yielding Equation 1:

$$4^1 + 4^2 + 4^3 = 84 \text{ distinct k-mers} \tag{1}$$

For a given sequence s of length L, let $\mathcal{K}_k$ be the set of all possible k-mers of length k, given by Equation 2:

$$\mathcal{K}_k = \{k_1, k_2, \ldots, k_{4^k}\}, \quad k_i \in \Sigma^k \tag{2}$$

Let $\text{freq}(s, k_i)$ denote the number of occurrences of k-mer k_i in s. The normalized frequency vector is then defined using Equation 3:

$$\mathbf{f}^{(k)}(s) = \left[\frac{\text{freq}(s,k_1)}{L}, \frac{\text{freq}(s,k_2)}{L}, \ldots, \frac{\text{freq}(s,k_{|\mathcal{K}_k|})}{L}\right] \tag{3}$$

(ii) Mer Score Features: To capture class-specific k-mer biases, mer scores are computed for $k = 1$ to 6. Let $F_c(k_i)$ and $F_{nc}(k_i)$ denote the frequency of k-mer k_i in the positive and negative training classes, respectively. Using a sliding window of size k, the local log-ratio at position i in sequence s is defined in in Equation 4 as:

$$\log_r(i) = \begin{cases} 0 & \text{if } F_c = 0 \text{ and } F_{nc} = 0 \\ -1 & \text{if } F_c = 0 \text{ and } F_{nc} > 0 \\ +1 & \text{if } F_c > 0 \text{ and } F_{nc} = 0 \\ \log\left(\dfrac{F_c}{F_{nc}}\right) & \text{otherwise} \end{cases} \tag{4}$$

The average mer score across all valid windows in sequence s is then given by in Equation 5:

$$\mu^{(k)}(s) = \frac{1}{L-k+1} \sum_{i=1}^{L-k+1} \log_r(i) \tag{5}$$

This yields one scalar feature per k, contributing a total of 6 features. All possible k-mers of lengths 1 to 6 are enumerated by computing the Cartesian product over the nucleotide alphabet $\Sigma = \{A, C, G, T\}$, resulting in in Equation 6:

$$4 + 16 + 64 + 256 + 1024 + 4096 = 5460 \text{ unique } k\text{-mers} \tag{6}$$

The final feature vector for each sequence is formed by concatenating the frequency and mer score features, as shown in Equation 7:

$$\begin{aligned} \mathbf{F}(s) = \big[&\mathbf{f}^{(1)}(s), \mathbf{f}^{(2)}(s), \mathbf{f}^{(3)}(s), \\ &\mu^{(1)}(s), \mu^{(2)}(s), \ldots, \mu^{(6)}(s) \big] \in \mathbb{R}^{90} \end{aligned} \tag{7}$$

The resulting feature matrix is reshaped to $(N, 90, 1)$, where N is the number of sequences, to match the input format required by the deep learning model. For five-fold cross-validation, the training, validation, and test sets followed an approximate ratio of 74:19:7. Without cross-validation, the split was approximately 68:25:7.

C. Model Architecture

(i) Convolutional Neural Network (CNN):

A 1D-CNN [30] is effective for extracting local patterns from sequential data such as nucleotide sequences. It applies sliding filters to detect features such as motifs and conserved regions. The one-dimensional convolution operation, shown in Equation 8, slides a filter over the input to produce a localized activation map:

$$c_t = \sum_{i=1}^{m} h_i \cdot z_{t+i-1} \tag{8}$$

where $\mathbf{z} = [z_1, z_2, \ldots, z_n]$ is the input sequence, $\mathbf{h} = [h_1, h_2, \ldots, h_m]$ is the convolutional filter of width m, and c_t is the convolved output at position t.

(ii) Long Short-Term Memory (LSTM):

To model long-range dependencies, an LSTM network is employed [31], which mitigates the vanishing gradient problem common in traditional RNNs.The network computes the forget gate (Equation 9), input gate (Equation 10), candidate cell state (Equation 11), and cell state update (Equation 12),

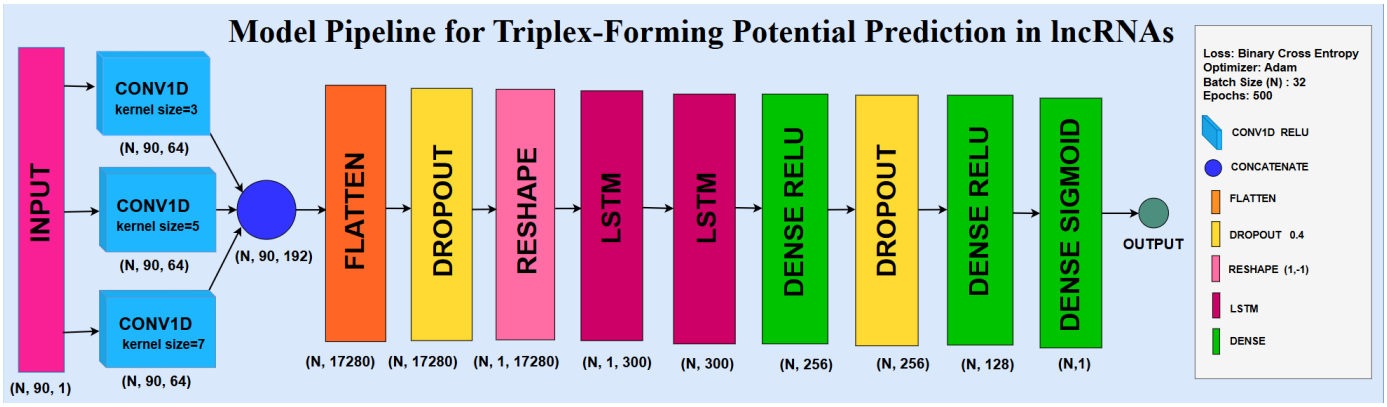

Fig. 2: End-End Framework for Triplex-Forming Potential Prediction in lncRNAs

followed by the output gate (Equation 13) and hidden state (Equation 14):

$$f_t = \sigma(W_f a_t + R_f h_{t-1} + b_f) \tag{9}$$
$$i_t = \sigma(W_i a_t + R_i h_{t-1} + b_i) \tag{10}$$
$$\tilde{s}_t = \tanh(W_g a_t + R_g h_{t-1} + b_g) \tag{11}$$
$$s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s}_t \tag{12}$$
$$o_t = \sigma(W_o a_t + R_o h_{t-1} + b_o) \tag{13}$$
$$h_t = o_t \odot \tanh(s_t) \tag{14}$$

In this formulation, f_t, i_t, and o_t are the forget, input, and output gates, respectively, each modulated via a sigmoid activation. The candidate cell state $\tilde{s}_t$ introduces new information through a tanh nonlinearity. The memory cell s_t (Equation 12) aggregates historical and current inputs, and the hidden state h_t (Equation 14) serves as the LSTM's output.

W_*, R_*, b_* represent the trainable parameters, and $\odot$ denotes element-wise multiplication. The subscripts f, i, g, o correspond to the forget gate, input gate, candidate generator, and output gate, respectively, with each gate having its own input weights W_*, recurrent weights R_*, and bias b_* to independently regulate information flow within the LSTM.

(iii) Multi-Scale CNN–LSTM Hybrid Architecture:

Three parallel Conv1D layers with kernel sizes 3, 5, and 7, each comprising 64 filters with ReLU activation and same padding have been used in our work as shown in Fig. 2 to extract local patterns at multiple receptive fields. Their outputs are concatenated along the feature axis and flattened, followed by a dropout layer with a 0.4 rate to mitigate overfitting. The flattened vector is reshaped into a single-timestep sequence $(1, N)$ for subsequent processing by two stacked LSTM layers with 300 units each; the first returns full hidden sequences, while the second outputs the final hidden state. This output feeds into two dense layers with 256 and 128 units respectively, both using ReLU activation and separated by a 0.4 dropout layer. The final output layer applies a sigmoid activation to yield a scalar probability for binary classification.

D. Evaluation Metrics

Model performance is assessed using several metrics: Accuracy, Sensitivity, Specificity, AUROC, AUPRC, F1 Score, and Harmonic Mean, as defined in Equations 15 to 23. These metrics are based on true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). TP and TN indicate correct classification of triplex and non-triplex sequences, while FP and FN represent misclassifications.

1. Accuracy: Accuracy quantifies the proportion of correct predictions over the total number of samples and serves as a general measure of correctness. It is defined in Equation 15:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{15}$$

2. Sensitivity (Recall or True Positive Rate): Sensitivity measures the model's ability to correctly identify triplex sequences and is especially critical when false negatives are costly, as given by Equation 16:

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{16}$$

3. Specificity (True Negative Rate): Specificity evaluates the model's capacity to correctly recognize non-triplex sequences, as given in Equation 17:

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{17}$$

4. AUROC (Area Under the ROC Curve): AUROC reflects the model's discrimination ability across thresholds. It integrates the True Positive Rate (TPR) over the False Positive Rate (FPR), as defined in Equation 18. The AUROC value is computed as shown in Equation 19:

$$\text{TPR} = \frac{TP}{TP + FN}, \quad \text{FPR} = \frac{FP}{FP + TN} \tag{18}$$

$$\text{AUROC} = \int_0^1 \text{TPR}(\text{FPR}^{-1}(x))\, dx \tag{19}$$

AUROC is computed using scikit-learn's roc_auc_score function with trapezoidal integration over the ROC curve.

5. AUPRC (Area Under the Precision-Recall Curve): AUPRC emphasizes model performance on imbalanced data. Precision and recall are defined in Equation 20, and the AUPRC is given by Equation 21:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN} \tag{20}$$

$$\text{AUPRC} = \int_0^1 \text{Precision}(\text{Recall}^{-1}(r))\, dr \tag{21}$$

This metric is computed using average_precision_score from scikit-learn.

6. F1 Score: F1 Score is the harmonic mean of precision and recall, providing a balanced measure of classification performance, especially useful in imbalanced datasets. It is defined in Equation 22:

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{22}$$

7. Harmonic Mean: The Harmonic Mean of sensitivity and specificity offers a balanced evaluation of true positive and true negative rates. It is suitable for assessing models on skewed datasets, as given by Equation 23:

$$\text{Harmonic Mean} = 2 \cdot \frac{\text{Sensitivity} \cdot \text{Specificity}}{\text{Sensitivity} + \text{Specificity}} \tag{23}$$

IV. Results and Discussion

A. Experimental Setup

JupyterLab (v2.3.2) was used as the initial development environment for building and deploying the prediction system. Development was carried out on a Windows 11 machine equipped with an NVIDIA RTX 6000 GPU hosted on a high-performance computing (HPC) server, enabling efficient model training, experimentation, and debugging. The environment included Python 3.8.10, TensorFlow 1.15.5, and Keras 2.2.4, along with scientific libraries such as NumPy, Pandas, Matplotlib, Bio, scikit-learn, seaborn, and livelossplot, and standard Python libraries like csv and math. This configuration supported legacy model architectures and allowed flexible prototyping. The project was eventually migrated to JupyterLab (v4.3.5) with TensorFlow 2.18.0 to align with modern machine learning practices.

After feature extraction, the dataset was shuffled using a fixed random seed of 42 to ensure reproducibility, followed by a stratified split into training, validation, and test sets. The model was optimized using the Adam optimizer with a learning rate of 1×10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$; the AMSGrad variant was not used. Training was performed for up to 500 epochs with a batch size of 32, monitoring validation performance throughout (Fig. 3).

During inference, the model generated probabilities for unseen sequences and classified them as Triplex or Non-Triplex based on a threshold of 0.5. Predictions, along with sequence identifiers and confidence scores, were saved to a CSV file. Gradio (v5.31.0), a Python library for building web-based machine learning interfaces, was employed to host a local server that visually presented the test results.

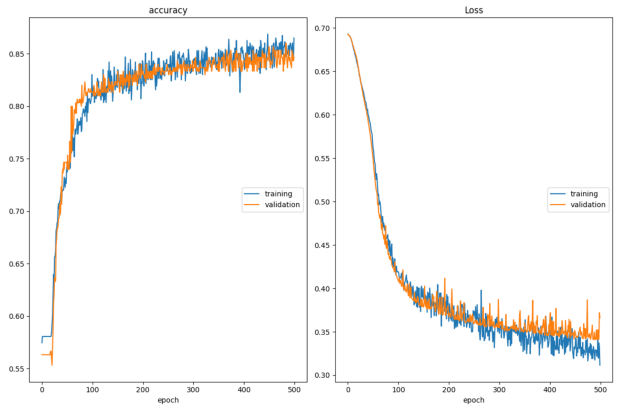

Fig. 3: Training and validation accuracy (left) and loss (right) over 500 epochs, demonstrating stable learning and good convergence for both metrics

B. Results

Out of the total 1,231 lncRNA sequences, 91 were reserved as an independent test set with preserved class distribution, and 10 were used exclusively for benchmarking against TriplexFPP [13]. The remaining 1,130 sequences were used for stratified five-fold cross-validation, ensuring class balance across folds. In each iteration, four folds were used for training and one for validation.

The cross-validation results, as shown in Fig. 4, showed an average accuracy of 90.11%, sensitivity of 85.71%, specificity of 93.88%, F1 score of 0.8889, and harmonic mean of 0.8961. AUROC and AUPRC were 0.9689 and 0.9683, respectively. Sensitivity across folds ranged from 80.95% to 88.10%, and specificity from 87.76% to 93.88%, with AUROC and AUPRC consistently above 0.96.

Without cross-validation, the model was trained on the full set of 1,130 sequences by reserving 300 samples for validation, achieving an accuracy of 85.71%, an AUROC of 0.9558, and an AUPRC of 0.9513. On the independent test set of 91 sequences, the model provided an unbiased performance estimate, as shown in Fig. 4 and Table III. It also outperformed TriplexFPP [13] on the additional 10 benchmark lncRNAs, confirming strong generalizability, as detailed in Table II.

.

C. Performance Comparison

The proposed method, evaluated on a balanced dataset of 531 triplex and 700 non-triplex samples, achieves an accuracy of 90.11%, sensitivity of 85.71%, specificity of 93.88%, AUROC of 0.9689, AUPRC of 0.9683, and F1 score of 0.8889 as shown in Table I.

While these metrics are slightly lower than those reported by Zhang et al. [13], who achieved accuracies above 95%

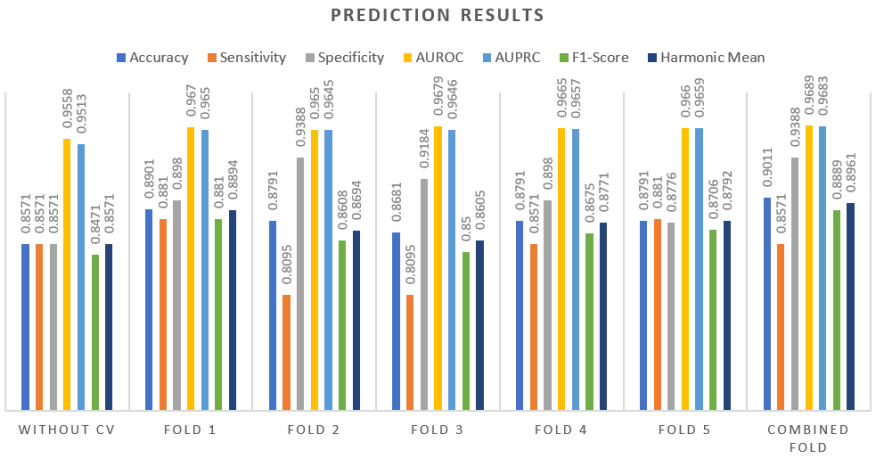

Fig. 4: Performance of the Proposed Model with and without Cross Validation on the Benchmark Dataset

TABLE I: Performance Comparison of Proposed Method with Existing Methods

Author	Dataset	# Samples		Method	Performance						
		Triplex	Non-Triplex		Acc	Sn	Sp	AUROC	AUPRC	F1	Hm
Zhang et al, 2020 [13]	Custom	531	36022	2-layer CNN	98.35	93.40	-	0.9926	0.9999	0.992	0.969
Zhang et al, 2020 [13]	Custom	384	28012	2-layer CNN, CD-HIT 0.9	95.28	-	-	0.9649	0.9996	0.976	0.904
Zhang et al, 2020 [13]	Custom	286	22681	2-layer CNN, CD-HIT 0.8	-	-	-	0.9637	0.9993	-	-
Proposed Method	Triplex lncRNA Dataset	531	700	1D CNN-LSTM	90.11	85.71	93.88	0.9689	0.9683	0.8889	0.8961

TABLE II: Performance Comparison of the Proposed Model with TriplexFPP [13]

Isoform	Len	True Labels	TriplexFPP		Proposed	
			Conf.	Pred.	Conf.	Pred.
MIR100HG-255	972	Triplex	0.87	Triplex	0.85	Triplex
SNHG7-204	2157	Triplex	0.21	Non-Triplex	**0.49**	Non-Triplex
MIR100HG-208	3334	Triplex	1.06	Triplex	0.98	Triplex
LINC00963-215	444	Triplex	0.78	Triplex	**1.00**	Triplex
MIR22HG-201	1383	Triplex	0.67	Triplex	**0.86**	Triplex
PVT1-215	1619	Triplex	0.72	Triplex	**1.00**	Triplex
MIR100HG-217	1820	Triplex	0.78	Triplex	0.73	Triplex
NEAT1-209	3301	Triplex	0.00	Non-Triplex	**0.24**	Non-Triplex
MEG3-232	515	Triplex	0.97	Triplex	**0.99**	Triplex
MEG3-205	1694	Triplex	1.02	Triplex	0.99	Triplex

TABLE III: Model Performance (Precision, Recall, F1-Score) on Triplex and Non-Triplex Class with and without Cross Validation

S.No	Fold	Class	Precision	Recall	F1 Score
1	Without CV	Non-Triplex	0.88	0.86	0.87
		Triplex	0.84	0.86	0.85
2	Fold 1	Non-Triplex	0.90	0.90	0.90
		Triplex	0.88	0.88	0.88
3	Fold 2	Non-Triplex	0.85	0.94	0.89
		Triplex	0.92	0.81	0.86
4	Fold 3	Non-Triplex	0.85	0.92	0.88
		Triplex	0.89	0.81	0.85
5	Fold 4	Non-Triplex	0.88	0.90	0.89
		Triplex	0.88	0.86	0.87
6	Fold 5	Non-Triplex	0.90	0.88	0.89
		Triplex	0.86	0.88	0.87
7	Combined Fold	Non-Triplex	0.88	0.94	0.91
		Triplex	0.92	0.86	0.89

and AUPRC values exceeding 0.99 using a two-layer CNN on larger and more imbalanced datasets, the proposed model demonstrates robust performance on a rigorously curated and balanced dataset.

This suggests that the proposed 1D CNN-LSTM model offers robust and reliable performance with improved generalizability despite lower absolute scores, reflecting a trade-off between dataset composition and model complexity.

D. Webserver

A Gradio interface is employed to run a local server that displays prediction results from a CSV file. Users can select an lncRNA through synchronized dropdowns using Transcript ID, Gene ID, or Alias. The interface outputs the metadata, predicted label as Triplex or Non-Triplex, and the corresponding confidence score indicating its potential for triplex formation, as illustrated in Figure 5.

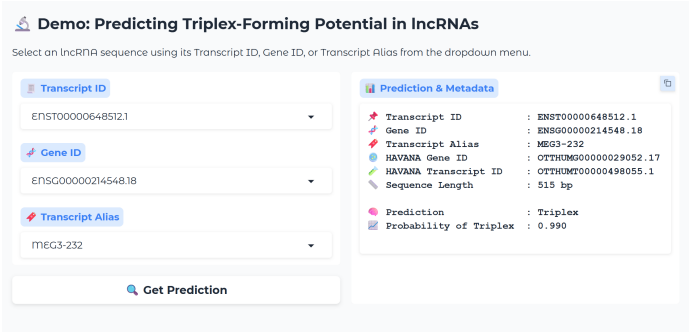

Fig. 5: Webserver Interface

V. Conclusion and Future Scope

This work presents a deep learning framework based on a 1D CNN–LSTM hybrid model that leverages sequence data to predict the triplex-forming potential of lncRNAs by capturing both local patterns and long-range dependencies. The proposed cascaded, multi-scale architecture integrates parallel convolutional paths for multi-resolution feature extraction with stacked LSTMs. Trained without synthetic resampling, the model achieved 90.11% accuracy, 85.71% sensitivity, 93.88% specificity, an AUROC of 0.9689, an AUPRC of 0.9683, an F1-score of 0.8889, and a harmonic mean of 0.8961 using stratified five-fold cross-validation. Dropout and extended training improved convergence and reduced overfitting, while confidence calibration minimized overconfident false negatives. These results demonstrate the model's robustness and utility for accurate and reliable lncRNA screening.

Future work includes using focal loss to improve learning from borderline cases and applying sequence-aware augmentation like SMOTE to balance the dataset. These steps aim to enhance model robustness and support large-scale lncRNA prescreening.

References

[1] Duca, M., Vekhoff, P., Oussedik, K., Halby, L., Arimondo, P. B. (2008). The triple helix: 50 years later, the outcome. Nucleic acids research, 36(16), 5123-5138.

[2] Felsenfeld, G., Davies, D. R., Rich, A. (1957). Formation of a three-stranded polynucleotide molecule. Journal of the American Chemical Society, 79(8), 2023-2024.

[3] Rajagopal, P., Feigon, J. (1989). Triple-strand formation in the homopurine: homopyrimidine DNA oligonucleotides d (GA) 4 and d (TC) 4. Nature, 339(6226), 637-640.

[4] Leisegang, M. S., Bains, J. K., Seredinski, S., Oo, J. A., Krause, N. M., Kuo, C. C., ... Brandes, R. P. (2022). HIF1-AS1 is a DNA: DNA: RNA triplex-forming lncRNA interacting with the HUSH complex. Nature communications, 13(1), 6563.

[5] Mondal, T., Subhash, S., Vaid, R., Enroth, S., Uday, S., Reinius, B., ... Kanduri, C. (2015). MEG3 long noncoding RNA regulates the TGF- pathway genes through formation of RNA–DNA triplex structures. Nature communications, 6(1), 7743.

[6] Sentürk Cetin, N., Kuo, C. C., Ribarska, T., Li, R., Costa, I. G., Grummt, I. (2019). Isolation and genome-wide characterization of cellular DNA: RNA triplex structures. Nucleic acids research, 47(5), 2306-2321.

[7] Chu, C., Quinn, J., Chang, H. Y. (2012). Chromatin isolation by RNA purification (ChIRP). Journal of visualized experiments: JoVE, (61), 3912.

[8] Bonetti, A., Agostini, F., Suzuki, A. M., Hashimoto, K., Pascarella, G., Gimenez, J., ... Carninci, P. (2020). RADICL-seq identifies general and cell type–specific principles of genome-wide RNA-chromatin interactions. Nature Communications, 11(1), 1018.

[9] Gavrilov, A. A., Zharikova, A. A., Galitsyna, A. A., Luzhin, A. V., Rubanova, N. M., Golov, A. K., ... Razin, S. V. (2020). Studying RNA–DNA interactome by Red-C identifies noncoding RNAs associated with various chromatin types and reveals transcription dynamics. Nucleic acids research, 48(12), 6699-6714.

[10] Buske, F. A., et al. (2012). Triplexator: detecting triple helices in genomic data. *Genome Research*, 22(7), 1372–1381.

[11] He, S., et al. (2015). LongTarget: a tool to predict lncRNA DNA-binding via Hoogsteen base-pairing. *Bioinformatics*, 31(2), 178–186.

[12] Kuo, C. C., et al. (2019). Detection of RNA–DNA binding sites in lncRNAs. *Nucleic Acids Research*, 47(6), e32.

[13] Zhang, Y., et al. (2020). Deep learning-based DNA:RNA triplex prediction. *BMC Bioinformatics*, 21, 1–13.

[14] Wen, Y., et al. (2022). Fasim-LongTarget: fast and accurate genome-wide lncRNA/DNA binding prediction. *Computational and Structural Biotechnology Journal*, 20, 3347–3350.

[15] Warwick, T., Seredinski, S., Krause, N. M., Bains, J. K., Althaus, L., Oo, J. A., ... Brandes, R. P. (2022). A universal model of RNA. DNA: DNA triplex formation accurately predicts genome-wide RNA–DNA interactions. Briefings in bioinformatics, 23(6), bbac445.

[16] Cicconetti, C., et al. (2023). 3plex enables deep computational investigation of triplex forming lncRNAs. *Computational and Structural Biotechnology Journal*, 21, 3091–3102.

[17] Hincapié-López, M., et al. (2024). Comparison of classical ML algorithms to predict TFOs. *Computational and Structural Biotechnology Reports*, 1, 100013.

[18] Xian, L., et al. (2025). Easy triplex: tool for predicting DNA–RNA triple helices. *Computational and Structural Biotechnology Journal*, 27, 1550–1558.

[19] Zhang, Z., Schwartz, S., Wagner, L., Miller, W. (2000). A greedy algorithm for aligning DNA sequences. Journal of Computational biology, 7(1-2), 203-214.

[20] Myers, G. (1998, July). A fast bit-vector algorithm for approximate string matching based on dynamic programming. In Annual Symposium on Combinatorial Pattern Matching (pp. 1-13). Berlin, Heidelberg: Springer Berlin Heidelberg.

[21] Benjamini, Y., Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal statistical society: series B (Methodological), 57(1), 289-300.

[22] Farrar, M. (2007). Striped Smith–Waterman speeds database searches six times over other SIMD implementations. Bioinformatics, 23(2), 156-161.

[23] Huang, X., Miller, W. (1991). A time-efficient, linear-space local similarity algorithm. Advances in applied mathematics, 12(3), 337-357.

[24] Karlin, S., Altschul, S. F. (1990). Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. Proceedings of the National Academy of Sciences, 87(6), 2264-2268.

[25] Varenyk, Y., Spicher, T., Hofacker, I. L., Lorenz, R. (2023). Modified RNAs and predictions with the ViennaRNA Package. Bioinformatics, 39(11), btad696.

[26] Kaufmann, B., Willinger, O., Kikuchi, N., Navon, N., Kermas, L., Goldberg, S., Amit, R. (2021). An Oligo-Library-Based Approach for Mapping DNA–DNA Triplex Interactions In Vitro. ACS Synthetic Biology, 10(8), 1808-1820.

[27] Zhang, Y., Jia, C., Fullwood, M. J., Kwoh, C. K. (2021). DeepCPP: a deep neural network based on nucleotide bias information and minimum distribution similarity feature selection for RNA coding potential prediction. Briefings in bioinformatics, 22(2), 2073-2084.

[28] Zhang, Y., Jia, C., Kwoh, C. K. (2021). Predicting the interaction biomolecule types for lncRNA: an ensemble deep learning approach. Briefings in Bioinformatics, 22(4), bbaa228.

[29] Chang, J., Chapman, B., Friedberg, I., Hamelryck, T., de Hoon, M., Cock, P., ... Wilczynski, B. (2010). Biopython tutorial and cookbook. Update, 15-19.

[30] Ige, Ayokunle Olalekan, and Malusi Sibiya. "State-of-the-art in 1d convolutional neural networks: A survey." IEEE Access (2024).

[31] Van Houdt, Greg, Carlos Mosquera, and Gonzalo Nápoles. "A review on the long short-term memory model." Artificial intelligence review 53.8 (2020): 5929-5955.