Journal of Intelligent Systems and Internet of Things
Vol. 18, No. 02, PP. 315-326, 2026
DOI: https://doi.org/10.54216/JISIoT.180222

DNA Sequence Identification via Biologically Guided Feature Engineering and Hybrid ML–LSTM Networks

Marwa Mawfaq Mohamedsheet Al-Hatab[1,*], Maysaloon Abed Qasim[2], Sinan S. Mohammed Sheet[1]
[1]Technical Engineering College, Northern Technical University, Mosul, Iraq
[2]Technical Engineering College for Computer and Artificial Intelligence, Northern Technical University, Mosul, Iraq
Emails: marwa.alhatab@ntu.edu.iq; maysloonalhashim@ntu.edu.iq; sinan_sm76@ntu.edu.iq

Abstract

The promoter is the part of DNA, which is responsible of initiating RNA polymerase transcription of a gene. The location of this part of DNA is upstream the transcription start site. According to researches, the genetic promotors contribute majorly in many human diseases such as cancer, diabetes and Huntington's disease. Therefore, promotor detection corresponds as a very crucial task. In this study, a hypered detection system, which integrates biologically developed feature extraction with traditional machine learning (ML) algorithms in addition to use Long Short-Term Memory (LSTM) network as a deep learning approach, has been proposed. The dataset used includes 106 nucleotide sequences. Results obtained from the study show that the perfect performance across all metrics (accuracy, sensitivity, specificity, precision, and F1-score) has been achieved when Naive Bayes was used as a classifier, which reached 100% and AUC=1. The confusion matrix analyses and ROC curve confirm that LSTM model achieved 100% training accuracy and 84.38% test accuracy. The architecture and performance of the proposed model make it applicable in IoT-based intelligent genomic and healthcare systems, which enabling real-time and remote promoter detection.

Received: March 14, 2025 Revised: June 02, 2025 Accepted: July 10, 2025

Keywords: Promoter detection; Machine learning; LSTM

1. Introduction

A promoter is a specific region of DNA where RNA polymerase begins the transcription of a gene. These sequences are typically located just before, or upstream of, the transcription start site. Both RNA polymerase and essential transcription factors attach to the promoter and the transcription initiation site. The promoter also determines which strand of DNA (often called the sense strand) will serve as the template and sets the direction in which transcription proceeds. [1].

Promoters are stretches of DNA that interact with RNA polymerase to regulate the position and frequency of transcription initiation. Comparative studies of a limited number of promoters identified two conserved regions located approximately 35 and 10 base pairs upstream of the transcription start site [2]. Broader analyses of promoters in Escherichia coli, as well as in its associated phages and plasmids, reinforced the model of a "consensus" promoter sequence: a TTGACA motif near the -35 region and a TATAAT motif around the -10 region typically separated by about 17 base pairs. Transcription generally begins at a purine nucleotide about 7 base pairs downstream from the 3′ end of the -10 region [3]. Additional nucleotides flanking the -35 and -10 sites, along with those near the transcription start point, occur more frequently than random and can sometimes influence promoter efficiency. However, the -35 and -10 motifs show the highest level of sequence conservation and are the primary sites where mutations affect transcriptional strength. Moreover, variations in the spacing between these two conserved regions can affect how effectively the promoter functions [4].

2. Related Work

The identification of promotor regions is crucial for understanding the gene transcription regulation. In a study by Oubounyt et al. (2019), DeePromoter, a deep learning framework that integrates Long Short-Term Memory (LSTM) networks with Convolutional Neural Networks (CNN) have been introduced. This model generates a negative dataset from the sequences of the promotor instead of select non-promoter regions randomly, and by this method the prediction, accuracy has been enhanced and rate of false positives has been reduced by. In order to improve accessibility, an online web server for the prediction system has been developed by authors [5].

Menon et al. (2020) proposed a hybrid method called IPMD used in predicting promoter regions in both prokaryotic and eukaryotic genomes. This method integrates a modified Mahalanobis Discriminant with a position correlation scoring function. Because of the variety of the structure of the promoter region and content of different functional motifs, accurately detecting them remains challenging. The authors mentioned that many current prediction systems do not meet performance expectations. They also suggested that the prediction tools could be improved by distinguishing between strong and weak promoters [6].

Bhandari et al. (2021) used different machine learning and deep learning algorithms to predict promotor regions cross three types of eukaryotic species: saccharomyces cerevisiae, Arabidopsis thaliana, and Homo sapiens. Two strategies have been used in their study as a preprocessing for inputting DNA sequences into a one-dimensional convolutional neural network (CNN), the first one is frequency-based tokenization (FBT) and the second is one-hot encoding. Their achieved results prove that FBT reduced input dimensionality, thereby speeding up training without any effeteness on the sensitivity or specificity of the model. The study showed that that CNN has a big effectiveness in binary and multiclass classification of promoter sequences [7].

Habib et al. (2022), introduced another study and used available promoter and splice site datasets to evaluate different machine learning approaches used in the classification of DNA sequences. There results highlighted that most of the approaches achieved accuracy more than 90% on test data, while accuracy of just two models was less than 90% in training data. These results confirmed that using machine learning techniques in DNA sequence classification add robustness and high performance to these systems [8].

Nikumbh and Lenhard (2023), produced a chunking-based strategy to classify promoter sequence patterns using non-negative matrix factorization (NMF). By implemented the tool seqArchR, their method groups promoter sequences based on motifs lied at specific distances from transcription start sites (TSS). In this study, the classification of promoters has been enhanced into functional categories and the system succeeded in identifying both known TSS-associated motifs, such as TATA and DPE, in addition to new lineage-specific sequence motifs. The study emphasized that the core promoters is very important in facilitating the binding of transcription initiation complexes [9].

Paul et al. (2024) developed a new method called MLDSPP (Machine Learning and Duplex Stability Promoter Prediction in Prokaryotes) in order to classify bacterial promoter regions through 12 different genomes. The system combined DNA structural attributes like duplex stability with advanced machine learning algorithms such as XGBoost and by this combination it superiors the performance of known tools such as Sigma70pred and iPromoter2L and can archive F1-scores more than 95% and also improved both the accuracy and interpretability of predictions [10].

This study also examines the impact of carefully engineered sequence features and advanced classification methods on distinguishing promoters from non-promoters. While previous research has utilized traditional DNA descriptors and standard machine learning techniques for promoter detection, they often overlook the potential of novel feature designs that more fully capture the intricate patterns within DNA sequences. Moreover, many existing approaches fail to exploit sophisticated sequential models specifically suited for genomic data.

To fill these gaps, we propose a new collection of biologically informed and computationally derived features that effectively represent key sequence properties. To assess their value, we tested different kinds of traditional machine learning classifiers, Support Vector Machines (SVM), Logistic Regression (LR), k-Nearest Neighbors (KNN), Decision Trees (DT), and Naive Bayes (NB) and compare their performance against a deep learning model based on Long Short-Term Memory (LSTM) networks. Our LSTM architecture takes 57 extracted features per sequence as input, processes them through an LSTM layer with 100 hidden units, and then passes the output through fully connected and SoftMax layers for promoter versus non-promoter classification [11].

Our results reveal that combining rich biological and computational feature sets with advanced classifiers substantially improves prediction accuracy, providing a powerful and versatile framework for promoter identification.

3. Methodology

Figure 1 illustrates step-by- step methodology adopted in this study.

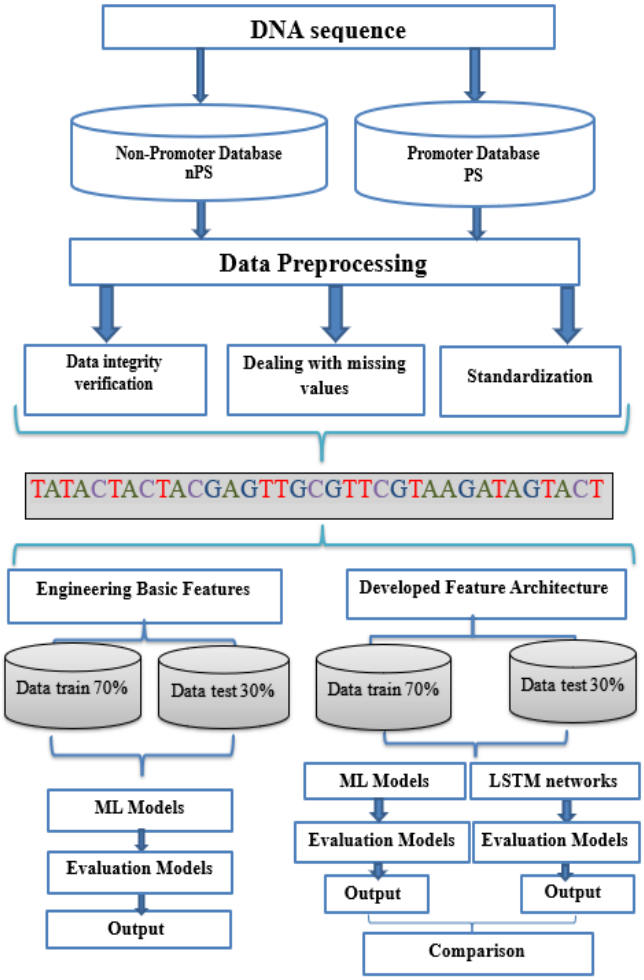


Figure 1. Flowchart of the methodology

3.1 Data Set and Preprocessing

The dataset used for this study contains 106 DNA sequence instances, with two classes equally distributed: 53 promoter sequences (positive cases) and 53 non-promoter sequences (negative cases). Each sequence contains 57 base-pair positions, spanning from –50 to +7 relative to the transcription start site [12]. This dataset is publicly available.

3.1.1 Split the data into training and testing datasets

The dataset has been partitioned randomly into two subsets in order to enable model training and independent evaluation. 80% of the data, which represent (84 instances), has been specified for training while the remaining 20%, which represents (22 instances), has been allocated for testing.

3.1.2 Data integrity verification

Before processing, the dataset has been checked in to verify that all instances have the expected length of 57 base-pair positions and the class labels are correctly specified. To maintain data integrity any inconsistencies have been determined and investigated

3.1.3 Dealing with missing values

In this part of dataset, no missing values were observed and each instance has been populated with valid base-pair data. However, in the case of presence of missing data, a suitable imputation strategy can be applied to prevent losing of any information and keep feature space consistently.

3.1.4 Standardization

Feature standardization has been applied to the numerically encoded base-pair representations in order to improve the model convergence and ensure that all features contributed equally during training. Each feature was rescaled to zero mean and unit variance according to the training set statistics. Then the same transformation parameters have been applied to the testing set to avoid the leakage of the data and ensure the faith of evaluation.

3.2. Feature Engineering

3.2.1 Basic Feature Engineering

The basic feature engineering method treated with DNA sequences as groups of individual nucleotide symbols, adenine (A), thymine (T), cytosine (C), and guanine (G) [13]. In this approach, each DNA strand has been broken into separate nucleotide components, each nucleotide considered an individual feature.

3.2.2 Advanced Feature Engineering

The main core of advanced feature engineering method is deriving characteristics from DNA sequence in order to enhance classification performance. This method analyzes the nucleotide composition, evaluate G and C content, assess k-mer frequency and analyze sequence complexity. Table 1 and 2 illustrate details of these newly developed features.

Table 1: Developed feature values for PS and n_PS of DNA Sequences

Feature	Description	Promoter Sequences (PS) Regions	Promoter Sequences (PS) Regions	Biological Significance
Nucleotide Count (A, T, C, G)	Numbers of each type of nucleotide within the DNA sequence.	- A: 15-18 per 57 nucleotides - T: 16-19 per 57 nucleotides - C: 11-13 per 57 nucleotides - G: 10-12 per 57 nucleotides	- A: 13-15 per 57 nucleotides - T: 14-16 per 57 nucleotides - C: 13-14 per 57 nucleotides - G: 14-15 per 57 nucleotides	Promoter (PS) regions normally contain a higher concentration of adenine (A) and thymine (T), and this make the DNA unwinding and transcription factor binding easy. Conversely, non-promoter (n_PS) regions always have more distribution of nucleotides, which lead to structural stable in the DNA.
G_C Content	Proportion of Gand C in the DNA sequence.	40-45%	48-52%	Low level of G and C nucleotides in promoter (PS) regions increases the flexibility of DNA helping in initiate the transcription, while high level of them in non-promoter (n_PS) regions gives greater structural stability.
K_mer Analysis	Frequency of subsequences of length k (e.g., 2-mers, 3-mers).	- Frequent motifs: TATA, CGG, GCG - Significant k-mers: High frequency of regulatory motifs	- Frequent motifs: Random or less structured patterns - Significant k-mers: No consistent motifs	K-mer analysis discovers organized sequence patterns in promoter (PS) regions, which are essential in the regulation of the gene; in the other hand non-promoter (n_PS) regions have patterns that are more random.
Sequence Complexity	Variability in nucleotide distribution across the sequence.	High complexity: Presence of varied motifs and elements	Low complexity: Simple, repetitive, or random patterns	The high complex sequence in PS indicates having regulatory elements, but lower complexity in n_PS shows lower regulatory function.

Table 2: Average nucleotide levels for PS and n_PS

Nucleotide	Average Count PS	Average Count n_PS	Biological Significance
A	15.79	14.02	A is more common in PS type, which leads to initiate unwinding and transcription
T	17.19	15.11	High level of T in PS can help in making DNA more flexible and make the unwinding easier for transcription.
C	12.62	13.51	The decrease of C in nucleotide in PS leads to make DNA less stable
G	11.40	14.45	G nucleotide usually low in PS and this impress the stability of DNA and the accessibility for transcription machinery.

3.2.3. Nucleotide Frequency Analysis:

In order to provide a foundational understanding of the sequence composition the first step is counting the occurrences of each nucleotide, adenine (A), thymine (T), cytosine (C), and guanine (G) across all DNA sequences [14].

3.2.4. Measurement of G_C Content:

This feature calculates the frequency of guanine and cytosine in a DNA sequence. G-C pairs contribute to increase DNA stability because they form three hydrogen bonds which is opposed to A-T pairs which form two hydrogen bonds [15].

3.2.5. K-mer Frequency Analysis:

The purpose of K-mer analysis is to examine the subsequences with k length in DNA strand in order to detect the recurring motifs. This analysis is essential to identify common elements associated with promoter function, such as the TATA box or C_G-rich areas [16].

3.2.6. Sequence Complexity Evaluation:

This metric assesses the ariablety or repetitivety of a DNA sequence.

3.2.7. Combined Feature Strategy:

By integrating all of these features, nucleotide counts, GC content, k-mer patterns, and sequence complexity, the model gains a comprehensive view of the differences between promoter and non-promoter regions.

Unlike traditional feature engineering methods in which DNA sequences are treated as individual nucleotide symbols (A, T, C, G) without taking in consider their biological or structural context, this study developed features that combine biologically meaningful patterns and structural insights. In order to capture regulatory signals specific to promoter regions, features such as nucleotide distribution profiles, GC content, k-mer motif analysis, and sequence complexity have been designed. The developed features enhance model interpretability and predictive performance in addition to reflect the physical and functional differences between promoter and non-promoter regions

3.3. Classifier Initialization and Model Selection

3.3.1 SVM

SVM is a supervised learning algorithm for binary classification that seeks to identify the optimal hyperplane which maximizes the margin between the two classes, in our study promoter and non-promoter DNA sequences. The decision function for the linear SVM can be expressed as in equation (1)[17].

$$f_{linear}(x) = \sum_{i=1}^{n} \alpha_i \, y_i (x . x_i) + b \quad (1)$$

where: α_(i) is the Lagrange multiplier, y_i class labels, and x_i support vectors.

3.3.2 KNN

KNN is a non-parametric, instance-based classifier that classifies a DNA sequence based on the most common class label among its k nearest neighbors. as in equation (2)[17]:

$$U = \arg \, max_U \sum_{i=1}^{k} I(U_i = U) \quad (2)$$

where: I(U_i=U) is represent indicator function, if (U_i=U) the value is 1 and otherwise 0. k is several nearest neighbors.

3.3.3. LR

LR provides a probabilistic model that maps DNA sequence feature vectors to promoter probability estimates. The algorithm fits a linear decision boundary in the feature space and applies the sigmoid activation to obtain class membership probabilities as in equation (3)[18].

$$(y = 1\backslash X) = \frac{1}{1+e^{-(w.X+b)}} \quad (3)$$

where: is X feature vector, w represents the weight vector, and b is the bias term.

3.3.4 NB

NB is a probabilistic classifier that applies Bayes' theorem under the simplifying assumption that all features are conditionally independent given the class label. Given the continuous nature of the feature vectors, the Gaussian Naive Bayes variant was adopted. The rule is represented by equation (4)[19]:

$$P(y\backslash X) = \frac{P(y)\prod_{i=1}^{n} P(x_i\backslash y)}{P(X)} \quad (4)$$

Where: The prior probability of the class is represented by P(y), P(x_i\y) is the probability of feature and x_i is the given class.

3.3.5 DT

Decision Trees classify sequences by recursively partitioning the feature space into class-specific regions. This yields an intuitive set of hierarchical decision rules that can highlight the most informative features, as in equation (5) [20]:

$$Gini(t) = 1 - \sum_{i=1}^{c} p(i\backslash t)^2 \quad (5)$$

where p(i\t) is the proportion of class i at node t.

3.4. Applications of using IoT for Promoter Detection in Genomics

The proposed system of promotor detection can be applied using Internet of Things (IoT) frameworks, which use connected devices like sensors or portable diagnostic tools to collect and analyze genomic data in real time. Although that this study does not integrate IoT technology directly, but the system ability to use advanced neural networks (e.g., LSTM) in sequential DNA data processing make it suitable for using smart healthcare and remote genomic monitoring systems in the future.

3.4.1 Long Short-Term Memory (LSTM) Network Architecture

The nucleotide promoter classification system was implemented using (LSTM) neural network. This architecture was specifically chosen for its ability to model sequential data and capture complex dependencies between nucleotide features. The design of the network includes a well-defined sequence of layers, parameter settings, and optimized training configurations.

3.4.2 Network Layers

The proposed LSTM network processes nucleotide sequences through multiple layers that transform raw input features into class predictions [21]. Table 3 presents the complete layer configuration.

Table 3: Detailed Network Layers and Descriptions

Layer	Description	Parameters
Sequence Input	Accepts input sequences with 57 parallel features	Input size: 57
LSTM Layer	Captures temporal dependencies and internal feature correlations	Hidden units: 100
Fully Connected	Maps LSTM outputs to class scores	Output size: Number of classes
SoftMax	Converts raw class scores to probabilities	-
Classification	Computes categorical cross-entropy loss for optimization	-

3.4.3 Network Parameters

The network parameters were carefully selected to balance classification accuracy, computational efficiency, and model generalization. Table 4 shows detailed parameter configuration.

Table 4: Network parameters configuration

Parameter	Value	Description
Input Feature Size	57	Number of nucleotide features per sequence
Sequence Length	1	Each sample treated as one sequence
LSTM Hidden Units	100	Number of memory cells in LSTM layer
Output Classes	Automatically inferred from data	Number of unique classes in target variable
Output Mode	'last'	Use output at final time step for classification
Activation Function	SoftMax (final layer)	Converts class scores into probabilities
Loss Function	Cross-Entropy	Minimizes classification error during training

3.4.4 Training Options

Training of the LSTM network was performed using the adaptive Adam optimizer [22]. The training configuration is summarized in Table 5.

Table 5: Training configuration settings

Training Parameter	Value	Description
Optimization Algorithm	Adam	Adaptive learning rate optimization
Maximum Epochs	100	Maximum number of full passes over training data
Mini-Batch Size	64	Number of samples processed per training step
Shuffle	Every epoch	Randomly shuffles data at each epoch
Validation Data	30% of dataset	Reserved data for performance monitoring
Validation Frequency	Every 30 iterations	Frequency of validation during training
Learning Rate Schedule	Default (constant)	No learning rate decay applied
Training Monitoring	Enabled (real-time plots)	Tracks accuracy and loss progression visually
Verbose Output	Disabled	Suppresses detailed logs for cleaner output

3.5 Performance Evaluation:

In this study, the predictions for each ML model are evaluated using multiple metrics to assess classification accuracy and robustness [23].

3.5.1 Accuracy:

Accuracy can be calculated using equation (6)[24][25].

$$A = \frac{True\ Positive\ (TP) + True\ Negative(TN)}{True\ Positive\ (TP) + True\ Negative(TN) + False\ Positive(FP) + False\ Negative(FN)} \quad (6)$$

3.5.2 Precision:

Equation (7) illustrate the mathematical formula to determine precision.

$$Precision = \frac{TP}{TP+FP} \quad (7)$$

3.5.3 Recall (Sensitivity):

Equation (8) represents the Recall (sensitivity) and indicates actual positives.

$$Recall = \frac{TP}{TP+FN} \quad (8)$$

3.5.4 F1-Score:

By using equation (9) F1-Score can be determinate.

$$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (9)$$

4. Results and Discussion

4.1 performance a ML Classifier Models

Tables 6 and 7 summarize the output classification metrics for all classifiers

Table 6: Performance metrics of classifiers for engineering basic features

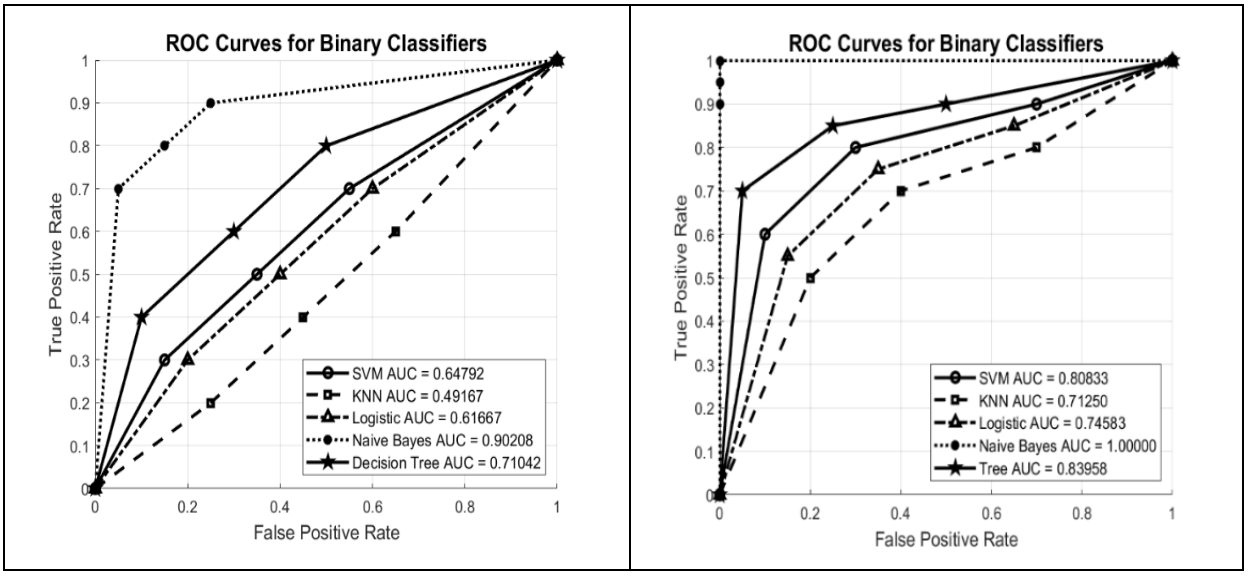
	Accuracy (%)	Precision	F1-Score	Sensitivity	Specificity
SVM	0.65	0.61	0.67	0.73	0.56
KNN	0.48	0.48	0.58	0.73	0.25
LR	0.61	0.58	0.65	0.73	0.5
DT	0.71	0.69	0.71	0.73	0.69
NB	0.90	0.93	0.90	0.870	0.94

Table 7: Performance metrics of classifiers for enhanced feature architecture

	Accuracy (%)	Precision	F1-Score	Sensitivity	specificity
SVM	0.81	0.76	0.81	0.87	0.75
KNN	0.71	0.67	0.73	0.8	0.63
LR	0.74	0.68	0.76	0.87	0.63
DT	0.84	0.81	0.84	0.87	0.81
NB	1	1	1	1	1

As illustrated in table 6 and 7 traditional classifiers trained on the basic feature set demonstrated only modest performance while when the classifiers trained using the enhanced feature architecture their performance improved significantly.

The ROC curves shown in Figure 2 compare the performance of classifiers using conventional features (Figure 2-a) versus the enhanced feature set (Figure 2-b). The results clearly demonstrate that improved feature engineering enhance classification performance across most models. Naive Bayes achieves a perfect AUC of 1.0 with the enhanced features, reflecting its exceptional ability to distinguish promoter sequences (PS) from non-promoter sequences (n_PS). Decision Trees also perform strongly, achieving an AUC of 0.8396, indicating robust discriminatory power. Support Vector Machines (SVM) and Logistic Regression (LR) yield moderate performance, with AUC values of 0.8083 and 0.7458, respectively. In contrast, k-Nearest Neighbors (KNN) shows the weakest performance, with a lower AUC of 0.7125, Naive Bayes maintains strong performance, achieving an AUC of 0.9021.



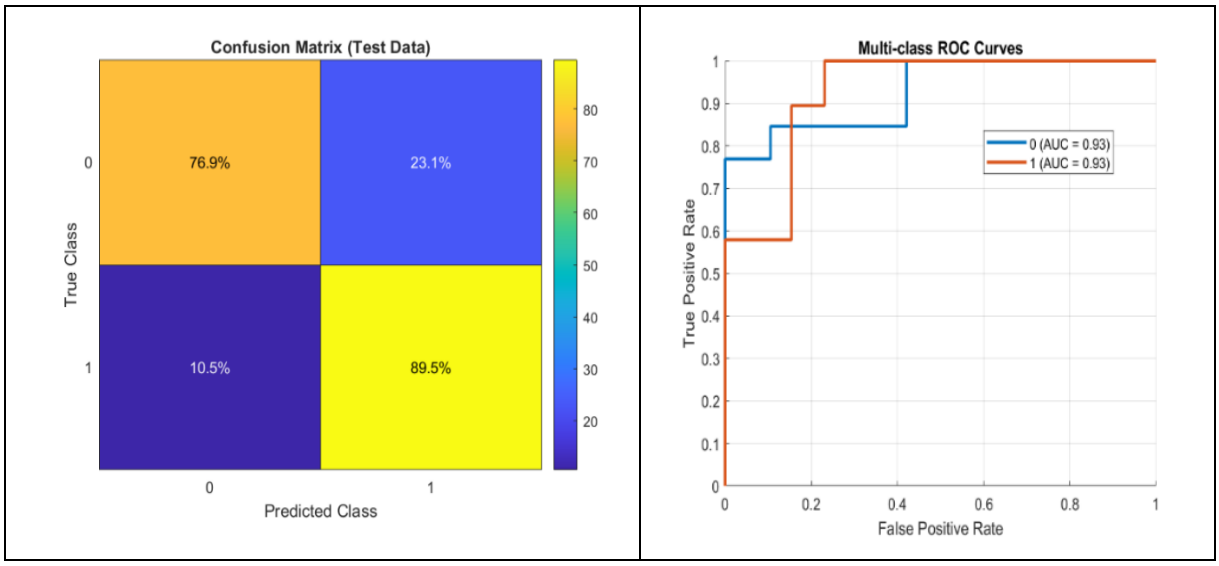
(a) (b)

Figure 2. ROC curves for binary classifiers: a- conventional features, b- Proposed features.

4.2 Analysis of LSTM Model

Figure 3-a which illustrates the confusion matrix shows a low number of false positives and false negatives, while the ROC curves for LSTM Model which is shown in figure 3-b exhibited a steep rise toward the top-left corner, with an area under the curve (AUC) nearing 1.0, underscoring the excellent discriminative power of the model.

Table (8) illustrates evaluation comparison between Machine Learning and Deep Learning, which is used.



(a) (b)

Figure 3. (a) Confusion matrix, (b) ROC curve for LSTM Model

Table 8: Comparison of evaluation between Machine Learning and Deep Learning

Model	Accuracy (%)	Strengths	Limitations	Key Reasons	Future Considerations
Naive Bayes	100	- Very effective with well-designed features - Simple and fast	- May not capture complex feature dependencies	- Handcrafted features captured essential nucleotide patterns	- Effective for small datasets if feature engineering is strong
Decision Tree	84	- Easy interpretability - Handles non-linear relations	- Limited with complex sequence dependencies	- Some dependencies captured but limited long-range ability	- Ensemble methods may improve robustness
SVM	81	- Good at handling non-linear boundaries - Strong in high-dimensional spaces	- Sensitive to parameter tuning - Requires well-structured features	- Benefited from discriminative handcrafted features	- Kernel optimization and hybrid models may enhance results
Logistic Regression	74	- Simple, interpretable - Low computational cost	- Limited in modeling complex patterns	- Captured some linear separable features	- Feature expansion or regularization may improve performance
KNN	71	- Simple, instance-based - No training phase	- Sensitive to data distribution and noise - Poor generalization in complex spaces	- Dependent on distance metrics for classification	- May benefit from feature dimensionality reduction
LSTM (RNN)	100 (Training) 84.38 (Test)	- Automatically learns sequence patterns - Captures short- and long-range dependencies - No feature engineering needed	- Requires large datasets - Risk of overfitting - Less interpretable	- Small dataset limited generalization - Strength increases with sequence complexity	- Larger, more diverse datasets - Transfer learning - Hybrid DL-biological feature mod

Our findings indicate that the newly developed feature extraction method significantly enhances the classification of promoter versus non-promoter sequences. The proposed approach yielded a higher proportion of correctly identified promoter sequences compared to traditional feature sets. When integrated with both conventional machine learning algorithms and LSTM classifiers, the enhanced feature architecture led to improvements in predictive performance. Among the models tested, the K-Nearest Neighbors (KNN) classifier achieved the highest accuracy and F1-score, suggesting strong compatibility with the extracted features.

These results underscore the value of incorporating both sequential dependencies and diverse feature representations for more accurate promoter prediction. However, it is important to note that the study was conducted using a relatively small dataset from a single species. Although the results were good, they reflect the limitations of the dataset and may be further and in-depth studies may be needed to confirm its generalizability and robustness, especially regarding its performance on larger, more diverse genomic datasets and its ability to capture complex sequential dependencies inherent in DNA.

5. Conclusion

As a conclusion, a hybrid promoter detection model that integrates biological feature extraction with traditional machine learning and LSTM deep learning has been introduced in this study. The results obtained confirm the strength of the proposed approach; Naive Bayes obtained a perfect performance of (100%), while the test accuracy using LSTM reached 84.38%. The system can be applied in real time in many applications such as early disease detection, including cancer and genetic disorders. Its design also makes it suitable for using in IoT-based genomic tools, enabling real-time and remote diagnostics. However, the small dataset size and lack of testing on broader genomic data make the system limited. Future work can focus on validation through using larger datasets and develop the system to use in lightweight, smart healthcare systems.

References

[1] C. Seila, L. J. Core, J. T. Lis, and P. A. Sharp, "Divergent transcription: a new feature of active promoters," *Cell Cycle*, vol. 8, no. 16, pp. 2557–2564, 2009, doi: 10.4161/cc.8.16.9335.

[2] V. Nain, S. Sahi, and P. A. Kumar, "In silico identification of regulatory elements in promoters," in *Computational Biology and Applied Bioinformatics*. InTech, 2011, pp. 47–66.

[3] R. McWhinnie, "Design of temperature inducible transcription factors and cognate promoters," Ph.D. dissertation, 2016.

[4] J. Blazeck and H. S. Alper, "Promoter engineering: recent advances in controlling transcription at the most fundamental level," *Biotechnol. J.*, vol. 8, no. 1, pp. 46–58, 2013, doi: 10.1002/biot.201200183.

[5] M. Oubounyt, Z. Louadi, H. Tayara, and K. T. Chong, "DeePromoter: robust promoter predictor using deep learning," *Front. Genet*, vol. 10, p. 286, 2019, doi: 10.3389/fgene.2019.00286.

[6] S. Menon, S. Piramanayakam, and G. Agarwal, "Computational identification of promoter regions in prokaryotes and eukaryotes," *EPRA Int. J. Res. Dev. (IJRD)*, vol. 6, no. 1, pp. 1–5, 2020, doi: 10.36713/epra7667.

[7] S. S. Bhandari, R. Walambe, and K. Kotecha, "Comparison of machine learning and deep learning techniques in promoter prediction across diverse species," *PeerJ Comput. Sci.*, vol. 7, p. e365, 2021, doi: 10.7717/peerj-cs.365.

[8] M. A. Habib, M. M. H. Manik, and B. Khulna, "Classification of DNA sequence using machine learning techniques," *EasyChair*, vol. 4, 2022, doi: 10.36300/easychair.4.2022.

[9] S. Nikumbh and B. Lenhard, "Identifying promoter sequence architectures via a chunking-based algorithm using non-negative matrix factorisation," *PLoS Comput. Biol.*, vol. 19, no. 11, p. e1011491, 2023, doi: 10.1371/journal.pcbi.1011491.

[10] S. Paul *et al.*, "MLDSPP: bacterial promoter prediction tool using DNA structural properties with machine learning and explainable AI," *J. Chem. Inf. Model.*, vol. 64, no. 7, pp. 2705–2719, 2024, doi: 10.1021/acs.jcim.4c00230.

[11] M. Takaku *et al.*, "ATAC-seq Guided Interpretable Machine Learning Reveals Cancer-Specific Chromatin Features in Cell-free DNA," *Res. Square*, Jan. 2025, doi: 10.21203/rs.3.rs-5485170/v1.

[12] D. Dua and C. Graff, *UCI Machine Learning Repository: Promoter Gene Sequences*. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Promoter+Gene+Sequences

[13] M. Martin-Landrove and B. P. Embaid, "Liapunov exponent distributions and maps for multiple parameter logistic equation. Application to DNA and RNA sequences," *arXiv preprint*, arXiv: 2505.02276, 2025.

[14] N. Kukushkin, *One Hand Clapping: Unraveling the Mystery of the Human Mind*. Rowman & Littlefield, 2025.

[15] F. Peña, L. Univaso, C. Román-Figueroa, and M. Paneque, "In Silico Genomic Analysis of Chloroplast DNA in Vitis Vinifera L.: Identification of Key Regions for DNA Coding," *Genes*, vol. 16, no. 6, p. 686, 2025, doi: 10.3390/genes16060686.

[16] Y. Hrytsenko, N. M. Daniels, and R. S. Schwartz, "Determining population structure from k-mer frequencies," *PeerJ*, vol. 13, p. e18939, 2025, doi: 10.7717/peerj.18939.

[17] M. M. Hussain, J. A. Zubair, A. Hassan, and K. Benahmed, "An improved K-nearest neighbors' classification for disease prediction," *IEEE Access*, vol. 8, pp. 100470–100477, 2020, doi: 10.1109/ACCESS.2020.2995684.

[18] N. Sharma, "Logistic regression in machine learning: A comprehensive study," *Int. J. Recent Technol. Eng. (IJRTE)*, vol. 8, no. 6, pp. 2471–2475, 2020, doi: 10.35940/ijrte.F8455.038620.

[19] K. Jaiswal and V. Srivastava, "An improved naive Bayes algorithm for disease prediction," in *Proc. 2017 Int. Conf. Comput. Commun. Technol. Smart Nation (IC3TSN)*, Gurgaon, India, 2017, pp. 173–176, doi: 10.1109/IC3TSN.2017.8284507.

[20] K. Jhajharia and P. Mathur, "A comprehensive review on machine learning in agriculture domain," *IAES Int. J. Artif. Intell*, vol. 11, no. 2, pp. 753–763, 2022, doi: 10.11591/ijai.v11.i2.pp753-763.

[21] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural Comput.*, vol. 31, no. 7, pp. 1235–1270, 2019, doi: 10.1162/neco_a_01199.

[22] M. Reyad, A. M. Sarhan, and M. Arafa, "A modified Adam algorithm for deep neural network optimization," *Neural Comput. Appl.*, vol. 35, no. 23, pp. 17095–17112, 2023, doi: 10.1007/s00521-023-08795-0.

[23] R. R. O. Al-Nima, M. M. M. Al-Hatab, and M. A. Qasim, "An artificial intelligence approach for verifying persons by employing the deoxyribonucleic acid (DNA) nucleotides," *J. Electr. Comput. Eng.*, vol. 2023, Art. no. 6678837, 2023, doi: 10.1155/2023/6678837.

[24] R. H. M. Ameen, N. M. Basheer, and A. K. Younis, "A survey: Breast cancer classification by using machine learning techniques," *NTU-JET*, vol. 2, no. 1, 2023, doi: 10.56286/ntujet.v2i1.367.

[25] S. Q. Hasan, “Shallow Model and Deep Learning Model for Features Extraction of Images”, *NTU-JET*, vol. 2, no. 3, 2023, doi: 10.56286/ntujet.v2i3.449.