DNA Sequence Classification: An Advanced Machine Learning Framework For Accurate Splice Junction Detection

1st Kazi shaharair Sharif
Department of Computer Science
Oklahoma State University
Stillwater, OK, USA
kazi.sharif@okstate.edu

2nd Ifrat Ikhtear Uddin
Department of Computer Science
University of South Dakota
Vermillion, SD, USA
ifratikhtear.uddin@coyotes.usd.edu

3th Md Abubakkar
Department of Computer Science
Midwestern State University
Dallas, TX, USA
mabubakkar@ieee.org

4th Md Munsur Khan
College of Graduate Studies
Trine University
Angola, IN, USA
mkhan24@my.trine.edu

5th Imran Ahmad
Department of Business Analytics
Wichita State University
Wichita, KS, USA
ixahmad1@shockers.wichita.edu

6th Mohammed Majbah Uddin
School of Business & Technology
Emporia State University
Emporia, Kansas, USA
muddin@g.emporia.edu

***Abstract*—In the context of genomic data analysis, DNA splice junction classification is a critical task for understanding gene expression, as these junctions are sites where introns are removed and exons are joined. Accurate identification of splice junctions is essential for deciphering gene functionality. Traditional methods, such as sequence alignment, are often slow and computationally intensive, especially when processing large-scale DNA datasets. To address this, we developed and evaluated multiple machine learning (ML) and deep learning (DL) models for the accurate classification of splice junctions. Our goal was to enhance classification accuracy, reduce computational costs, and provide a comparative analysis of different modeling approaches to advance research in genomic data analysis. We employed a methodological framework that included traditional ML algorithms, such as Random Forest, Gradient Boosting, Decision Tree, Support Vector Machine (SVM), and XGBoost, as well as contemporary DL architectures like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs). The data preprocessing pipeline incorporated one-hot encoding for optimal feature representation. Empirical results demonstrated the superior performance of ensemble learning methods, with Gradient Boosting and XGBoost achieving exceptional classification accuracies of 97.34% and 97.02%, respectively. Among DL models, CNNs outperformed RNNs, achieving 94.51% accuracy compared to 93.89% for RNNs. The results underscore the exceptional performance of tree-based ensemble methods for splice junction classification, highlighting their superior discriminative power and effectiveness in genomic sequence analysis.**

***Index Terms*—Machine Learning, Splice Site Prediction, DNA Sequence Classification, Machine Learning, Bioinformatics.**

I. Introduction

Splicing errors in gene expression are responsible for approximately 15-30% of all genetic disorders, including cancer, neurodegenerative diseases, and metabolic syndromes [1]. These errors often result in incorrect exon-intron recognition, leading to dysfunctional protein synthesis and, consequently, various genetic pathologies. It is estimated that around 10% of pathogenic mutations arise due to aberrant splicing, making splice site identification a critical task in genomic research [2]. Accurate classification of splice junctions, such as exon-intron (EI), intron-exon (IE), and non-splice (N) sites has profound implications for alternative splicing analysis, genetic disease diagnostics, and therapeutic target identification. However, despite the advancements in computational biology, precisely distinguishing splice sites remains a major challenge due to the complex dependencies and context-sensitive nature of DNA sequences. [3] [4]. Traditional computational methods for splice site detection, including motif-based models and sequence alignment tools, often suffer from low predictive accuracy, lack of scalability, and poor generalizability across large genomic datasets [5]. DNA sequences are inherently high-dimensional and exhibit complex interdependencies, requiring robust feature extraction and advanced pattern recognition techniques for effective classification. Moreover, class imbalances in genomic datasets often lead to high false-positive rates. Given these challenges, there is a growing demand for automated, high-precision predictive models that can effectively learn the intricate patterns within genomic sequences while ensuring generalization across different datasets [6] [7]. This study presents an integrated machine learning (ML) and deep learning (DL) framework designed to enhance splice site classification by leveraging the strengths of both approaches. While ML models, such as Random Forest, Gradient Boosting, and XGBoost, offer interpretability and computational efficiency, DL models like CNNs and RNNs excel in capturing complex sequence patterns. By combining these paradigms, this research aims to develop a robust and scalable benchmark framework that improves predictive accuracy while addressing challenges in interpretability, generalization, and scalability. Key contributions include optimizing feature engineering techniques such as one-hot encoding, sequence embedding, and hyperparameter tuning to enhance classification performance. This research enhances bioinformatics, computational genomics, and precision medicine by delivering a high-accuracy model for genomic sequence classification. Its findings pave the way for future applications in genetic mutation analysis, disease prediction, and bioinformatics-driven clinical decision-making, ultimately supporting advancements in personalized medicine and genomic research.

II. LITERATURE REVIEW

Accurate classification of DNA splice junctions is fundamental to understanding gene expression and regulation. Splice junctions define the precise boundaries where introns are removed and exons are joined, ensuring the integrity of genetic transcription. While traditional sequence alignment and homology-based approaches have been widely used for splice junction detection, these methods often face limitations in scalability and computational efficiency, especially when applied to large-scale genomic datasets [8]. With the advent of machine learning (ML) and deep learning (DL), computational approaches to genomic sequence classification have undergone significant transformation. Unlike conventional techniques, ML and DL models automate feature extraction, capturing intricate sequence patterns with higher accuracy and efficiency. A notable example is SpliceAI, introduced by Jaganathan et al. [9], which leverages a 32-layer residual neural network to predict splice junctions and splicing alterations due to genetic mutations with an impressive 95% accuracy.Building on these advancements, Chao et al. [10] developed Splam, a deep residual convolutional neural network designed specifically for splice site prediction. Unlike SpliceAI, which analyzes an extended genomic context of 10,000 nucleotides, Splam focuses on a 400-base-pair window, providing improved accuracy in detecting non-canonical splice junctions with a 96% classification accuracy. Further extending deep learning applications in this domain, Dutta et al. [11] proposed SpliceViNCI, an RNN-based model capable of identifying both canonical and non-canonical splice sites across multiple species. Their study underscored the necessity of designing models with robust generalization capabilities, ensuring adaptability to diverse genetic datasets. In an alternative approach, Chan et al. [12] introduced the Discrete Compositional Energy Network (DCEN), which models splice junction energy values to predict RNA alternative splicing events. This hierarchical model demonstrated enhanced predictive performance compared to conventional deep learning architectures. Similarly, Abd-Alhalem et al. [13] conducted a broad survey of CNNs and RNNs for DNA sequence classification.

Generative Adversarial Networks (GANs) have recently been explored for genomic sequence analysis, particularly for branchpoint site prediction. BP-GAN, introduced by [14], applies an attentive GAN framework to predict human branchpoints (BP) with high accuracy. Unlike conventional supervised learning models, BP-GAN employs adversarial learning to refine feature representations, improving both predictive performance and interpretability. The study reported a classification accuracy of 95.6%, outperforming traditional deep learning-based approaches. The introduction of attention mechanisms within the GAN framework enables BP-GAN to focus on biologically relevant sequence regions, leading to enhanced model explainability and robustness

The comparative performance of CNN and RNN models in splice junction classification was further explored by Aparajita et al. [15], who found that RNNs generally outperform CNNs when trained across multi-species datasets, particularly for detecting non-canonical splice sites. By benchmarking our models against established approaches such as SpliceAI, Splam, and SpliceViNCI, our study offers valuable insights into the strengths and limitations of different ML/DL methodologies. Our results suggest that hybrid approaches combining tree-based ensemble learning with deep learning architectures could provide the most effective solution for large-scale genomic sequence analysis.

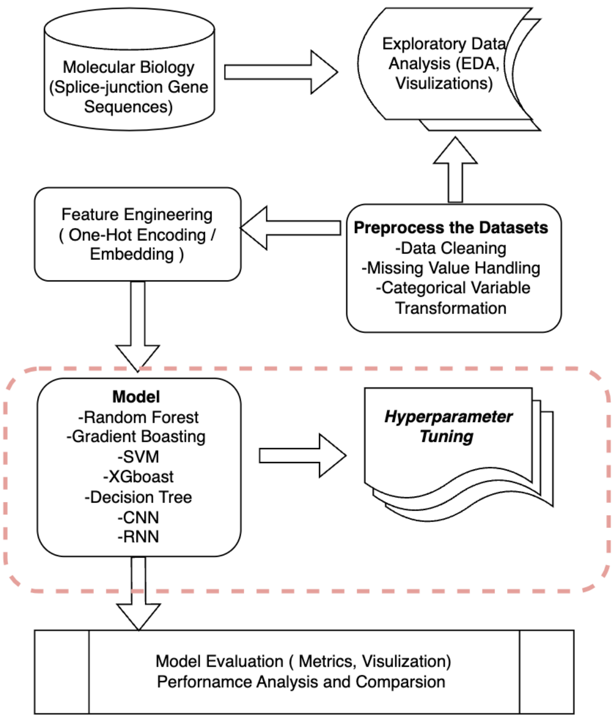
Molecular Biology (Splice-junction Gene Sequences)
Exploratory Data Analysis (EDA, Visulizations)
Feature Engineering (One-Hot Encoding / Embedding)
Preprocess the Datasets
-Data Cleaning
-Missing Value Handling
-Categorical Variable Transformation
Model
-Random Forest
-Gradient Boasting
-SVM
-XGboast
-Decision Tree
-CNN
-RNN
Hyperparameter Tuning
Model Evaluation (Metrics, Visulization)
Performamce Analysis and Comparsion

Fig. 1. Workflow Diagram

III. METHODOLOGY

This section outlines the systematic approach employed in developing a predictive framework. Figure 1 illustrates the entire workflow process used in the study, showcasing data preprocessing, model implementation, and optimization steps.

A. Data Exploration

The Molecular Biology (Splice Junction) Dataset from the UCI Machine Learning Repository consists of approximately 3,190 labeled DNA sequences, each 60 nucleotides long. It is designed for classification tasks, identifying Exon-Intron (EI) and Intron-Exon (IE) boundaries, as well as Non-Splice (N) regions. The dataset helps in understanding gene splicing by distinguishing transition points between exons and introns [16]. Figure 2 shows the class distribution of the target variable (EI, IE, and N), highlighting the imbalance in the dataset. This supports any discussion on data preprocessing, such as handling class imbalance using techniques like resampling or weighting.

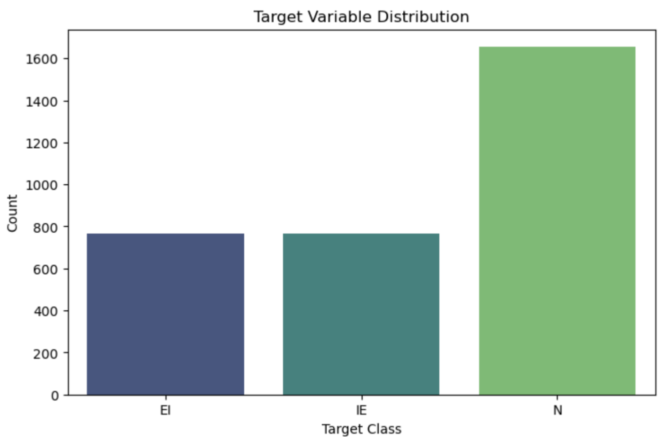

Fig. 2. Target Variable Distribution

B. Data Cleaning and Preprocessing

A series of preprocessing implementation were carried out to refine the data and enhance model performance. The first step involved handling missing or redundant data. To check that, the dataset contain any missing values, which eliminated the need for imputation. However, to prevent overfitting and ensure that the models learned from diverse data points rather than redundant patterns, duplicate sequences were identified and removed. Figure 3 visualizes the proportion of each nucleotide (A, T, G, C), which helps justify the selection of feature encoding techniques (one-hot encoding vs. embedding) for model input.Since DNA sequences are inherently categorical and sequential, they cannot be directly processed by ML and DL models in their raw form. To transform them into a format suitable for computational analysis, two primary encoding strategies were implemented. The first approach, One-Hot Encoding, involved representing each nucleotide (A, C, G, T), along with ambiguous bases such as R, S, and N, as binary vectors. This method allowed for an explicit representation of nucleotide variations while maintaining the positional integrity of the sequences. The second approach, Integer Encoding, was specifically used for deep learning models. In this method, each nucleotide was assigned a unique integer value, which was then used as input for an embedding layer in deep learning architectures such as Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs). Integer encoding was beneficial for capturing long-range dependencies in sequences, making it particularly suitable for models that leverage sequence learning capabilities. This step was crucial in maintaining the integrity of the dataset and improving model generalization. Figures 4 and 5 analyze the length variation of DNA sequences, helping to decide whether padding, truncation, or other preprocessing steps are required.

C. Feature Engineering and Dimensionality Reduction

Feature engineering was essential for transforming DNA sequences into a format suitable for machine learning (ML) and deep learning (DL) models. For ML models like Random

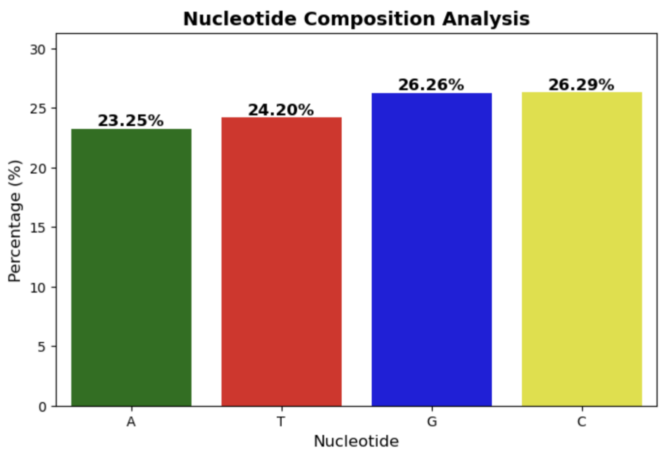

Fig. 3. Nucleotide Composition Analysis

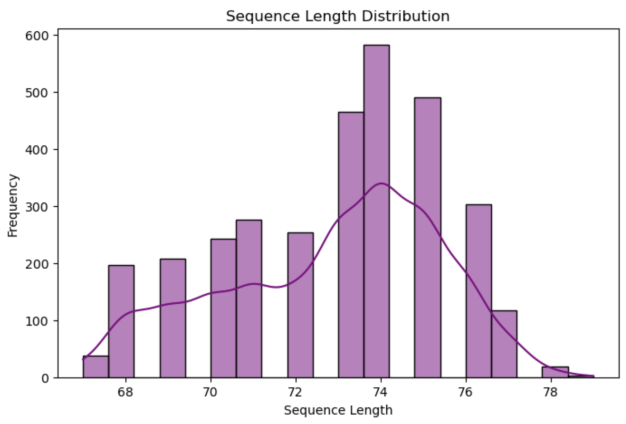

Fig. 4. Sequence Length Distribution

Forest and XGBoost, we used One-Hot Encoding, converting nucleotide bases (A, T, G, C, and ambiguous bases) into binary vectors, preserving positional information but increasing memory usage. For DL models like CNNs and RNNs, we applied sequence embedding using TensorFlow's Embedding Layer, capturing sequential dependencies for improved splice site prediction.

To ensure consistent scaling, we applied z-score normalization to ML features, aiding models like SVM, while DL models learned optimal feature distributions dynamically. Additionally, we explored dimensionality reduction with PCA and t-SNE to visualize feature space structure. PCA (Figure 6) was tested for dimensionality reduction but led to information loss, so it wasn't applied in training. Instead, t-SNE (Figure 7) provided insights into class separability by mapping high-dimensional features into a lower-dimensional space. These techniques helped ML models handle structured data while enabling DL models to learn hierarchical sequence patterns, improving classification accuracy and generalization.

D. Model Implementation

Our study implemented and optimized multiple machine learning (ML) and deep learning (DL) models for splice site classification. Each model was carefully fine-tuned, with

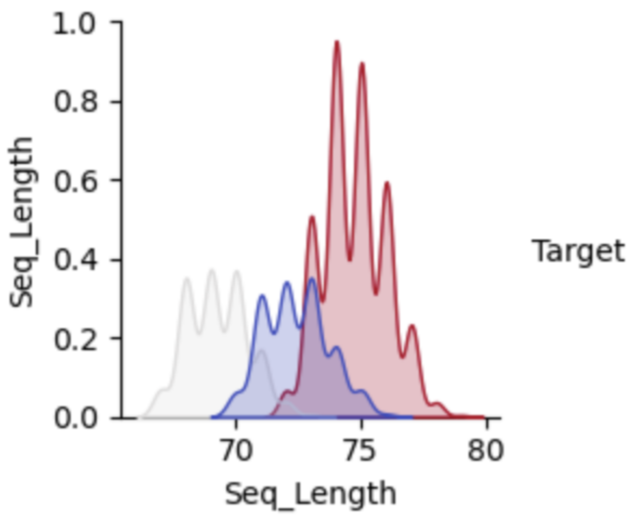

Fig. 5. Density Plot

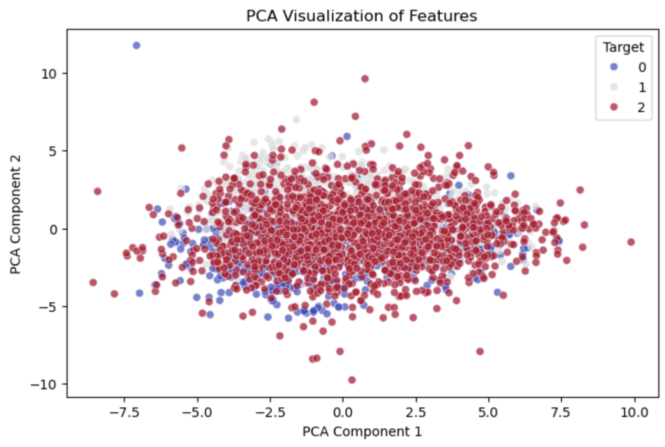

Fig. 6. PCA Visualization of Features

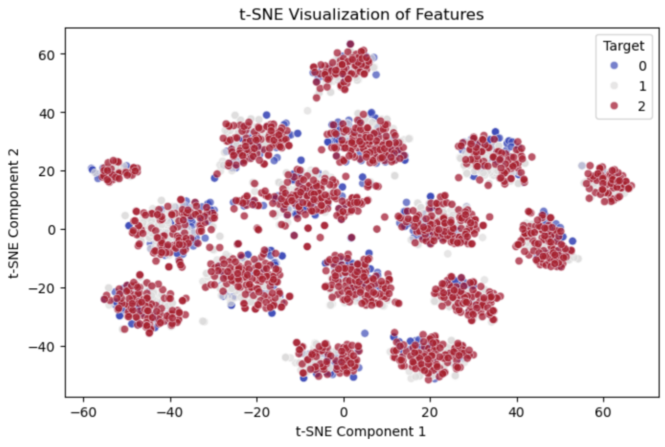

Fig. 7. t-SNE Visualization of Features

hyperparameters optimized through Grid Search and Random Search, while model performance was validated using cross-validation and separate test sets.

*1) **Random Forest**:* We employed Random Forest for its ability to handle complex, non-linear sequence relationships using an ensemble of decision trees. Hyperparameters such as the number of trees (100), max depth (None), and minimum samples per split were fine-tuned. Its robustness against overfitting and ability to capture important sequence-based predictors made it a strong candidate for classification.

*2) **Gradient Boosting**:* Gradient Boosting was implemented to improve prediction performance by iteratively refining weak learners. Key hyperparameters such as learning rate (0.1), number of estimators (100), and maximum depth were optimized. GB's ability to correct errors at each stage contributed to its strong generalization on unseen splice site data.

*3) **Decision Tree**:* A Decision Tree was used as a baseline classifier for comparison. By adjusting tree depth and minimum sample splits, we ensured the model was neither too simplistic nor prone to overfitting. While it provided clear interpretability, its performance was generally lower than ensemble models.

*4) **SVM**:* To handle high-dimensional sequence representations, we implemented an SVM with an RBF kernel. Key parameters such as C, gamma, and kernel type were optimized using Grid Search. Additionally, StandardScaler was applied to normalize input data, ensuring effective separation of splice site classes.

*5) **XGBoost**:* We selected XGBoost for its efficiency and performance improvements over standard gradient boosting. Tree depth (max depth = 6), learning rate (0.1), and the number of trees (100) were optimized. The model's regularization techniques (L1 & L2 penalties) and parallelized execution improved training speed and classification accuracy. XGBoost's feature importance scores also provided insights into the most informative sequence features.

*6) **Convolutional Neural Network (CNN)**:* Neural networks were designed to identify spatial patterns within DNA sequences using a structured approach. The model included Conv1D layers with 64 filters and a kernel size of 3 to extract meaningful local features, followed by MaxPooling1D, which helped reduce dimensionality and improve computational efficiency. The extracted features were then flattened and passed through fully connected dense layers, leading to the final classification using a softmax activation function. To ensure stable training, the model was optimized using Adam (learning rate = 0.001) and Sparse Categorical Crossentropy as the loss function.

*7) **Recurrent Neural Network (RNN) with LSTM**:* To capture long-range dependencies in DNA sequences, we built an LSTM-based recurrent neural network (RNN) designed to learn sequential patterns in splice sites. The model began with an embedding layer, which transformed nucleotide sequences into dense vector representations, making them more suitable for learning. This was followed by LSTM layers with 64 units, allowing the model to recognize important sequential dependencies. Finally, fully connected dense layers with a softmax activation function were used for classification. The model was trained using the Adam optimizer (learning rate = 0.001) for 10–20 epochs with a batch size of 32, ensuring stable learning and efficient processing of DNA sequences.

E. Model Training

The dataset was split into 80% training and 20% testing, allowing the models to learn patterns while retaining a separate portion for evaluation. For deep learning models (CNN, RNN-LSTM), we used a batch size of 32, ensuring efficient training while maintaining memory optimization. The models were trained for 10–20 epochs, striking a balance between learning convergence and avoiding overfitting. [17]. For machine learning models (Random Forest, XGBoost, SVM, etc.), cross-validation was applied to fine-tune hyperparameters. A k-fold cross-validation approach (typically k=5) was used, ensuring that each data subset contributed to both training and validation. This strategy helped improve generalization and reduced model variance across different runs [18].

F. Model Evaluation

The evaluation of the proposed model was conducted using comprehensive metrics to ensure diagnostic reliability.

Accuracy measures the proportion of correctly classified instances (true positives and true negatives) among all samples. Precision indicates the fraction of true positive predictions out of all positive predictions. Recall measures the model's ability to correctly identify all true positive cases. F1 Score provides a harmonic mean of precision and recall [19]. These metrics collectively demonstrate the robustness and clinical applicability of the model.

G. Hyperparameter Tuning

To ensure optimal model performance, we implemented systematic hyperparameter tuning techniques for both machine learning (ML) and deep learning (DL) models. For ML models such as Random Forest, XGBoost, Gradient Boosting, and SVM, we applied Random Search and Grid Search to explore the best combination of parameters. Random Search helped quickly identify promising hyperparameter ranges, while Grid Search fine-tuned parameters such as the number of estimators (trees), learning rate, max depth, and minimum samples per split for tree-based models. In SVM, we tuned kernel types (linear, RBF), regularization parameter (C), and gamma values [20].

For deep learning models (CNN, RNN-LSTM), we optimized hyperparameters through controlled experiments. We adjusted the learning rate dynamically using the Adam optimizer, selecting values from 0.0001 to 0.01 to balance training speed and model convergence. To prevent overfitting, Dropout Regularization was applied at rates between 0.2 and 0.5 in fully connected layers. Additionally, batch sizes (32, 64, 128) and the number of epochs (10–50) were tuned to achieve the best model generalization. These techniques significantly improved classification accuracy and model stability across different splice site categories.

IV. Results and Discussion

Table I highlights the performance of different machine learning (ML) and deep learning (DL) models based on accuracy, precision, recall, F1-score, and ROC AUC scores. The results show that Gradient Boosting and XGBoost were the top performers, achieving the highest accuracy and F1-scores. Random Forest also delivered strong results, while SVM and Decision Tree had slightly lower performance, likely due to challenges in handling complex sequence variations. Among ML models, Gradient Boosting had the highest accuracy at 97%, closely followed by XGBoost (97%) and Random Forest (96%). These models demonstrated strong precision and recall, making them highly reliable for splice site classification.Decision Tree (94%) and SVM (92%) had lower accuracy. SVM, in particular, faced challenges in clearly separating the different classes. For deep learning models, the CNN model achieved a high ROC AUC score of 0.9907, indicating its strong predictive power. The RNN model had a slightly lower AUC score of 0.9888, possibly due to its reliance on long-sequence dependencies and the need for further optimization.

TABLE I
Performance Metrics of Models on DNA Sequence Classification

Model	Precision	Recall	F1-Score	Accuracy
Random Forest	0.96	0.96	0.96	0.9639
Gradient Boosting	0.97	0.98	0.97	0.9734
Decision Tree	0.93	0.93	0.93	0.9373
SVM	0.91	0.92	0.92	0.9248
XGBoost	0.96	0.97	0.97	0.9702
CNN	0.94	0.94	0.94	0.9451
RNN	0.93	0.94	0.94	0.9389

Figure 8illustrates critical metrics such as ROC AUC, Precision, Recall, F1 Score, and accuracy were analyzed to comprehensively assess diagnostic reliability. ROC AUC is a key measure of how well a model can distinguish between different splice site classes. A higher score means better classification performance. Gradient Boosting and XGBoost had the best ROC AUC scores (0.9961), showing they are highly effective in distinguishing exon-intron (EI), intron-exon (IE), and non-splice (N) junctions. Random Forest followed closely with 0.9960, confirming its ability to capture complex patterns in DNA sequences. SVM achieved a solid 0.9868 AUC score, proving it can effectively separate splice site classes despite its lower accuracy. Decision Tree had the lowest AUC score at 0.9506, reflecting its struggle with more complex classification tasks. CNN and RNN had AUC scores of 0.9907 and 0.9888, respectively, showing their capability in learning sequence patterns but suggesting the need for further fine-tuning. Overall, Gradient Boosting and XGBoost were the most effective models, delivering high accuracy, F1-scores, and AUC scores, making them the best choice for splice site prediction. Deep learning models like CNN and RNN showed great potential but would benefit from additional training and fine-tuning to reach the performance level of tree-based models. Table II compares splice junction classification accuracy among state-of-the-art models. SpliceAI [9] achieved 95.00%, and Splam [10] improved to 96.00%. Our Gradient Boosting model outperformed both with 97.34%, demonstrating the superior effectiveness of tree-based ensemble learning for genomic sequence classification.

V. Conclusion and Future Works

This study developed a hybrid ML-DL framework for splice site prediction, combining XGBoost, Random Forest,

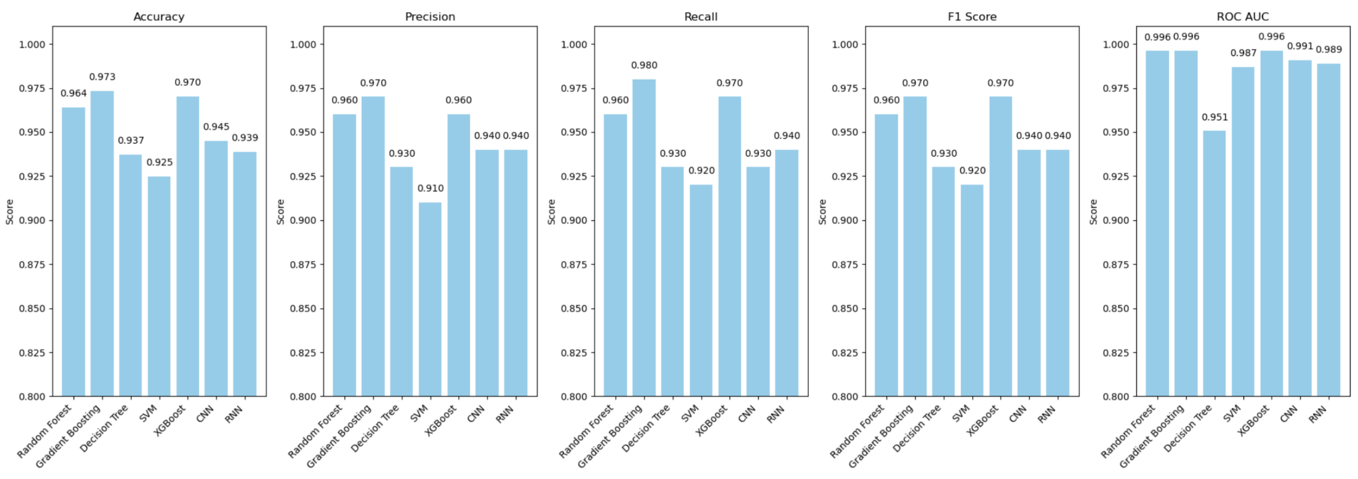

Fig. 8. Accuracy, Precision, Recall, F1 Score, ROC AUC Analysis

TABLE II
COMPARISON OF SPLICE JUNCTION CLASSIFICATION PERFORMANCE

Model	Study	Accuracy
SpliceAI [9]	Jaganathan et al. (2019)	95.00%
Splam [10]	Chao et al. (2024)	96.00%
Gradient Boosting (Our Model)	This Study	**97.34%**

CNN, and LSTM to improve classification accuracy. The approach effectively captured sequence patterns and long-range dependencies, while feature importance analysis provided interpretability. Future work will focus on expanding the dataset for better generalization and exploring SHAP and LIME for model explainability. The goal is to integrate this framework into bioinformatics tools, making splice site prediction more accurate and useful for genetic research and disease diagnostics.

REFERENCES

[1] D. Gao, E. Morini, M. Salani *et al.*, "A deep learning approach to identify gene targets of a therapeutic for human splicing disorders," *Nature Communications*, vol. 12, no. 1, p. 3332, 2021.

[2] T. V. Riepe, M. Khan, S. Roosing, F. P. Cremers, and P. A. 't Hoen, "Benchmarking deep learning splice prediction tools using functional splice assays," *Human Mutation*, 2021.

[3] N. Scalzitti, A. Kress, R. Orhand *et al.*, "Spliceator: multi-species splice site prediction using convolutional neural networks," *BMC Bioinformatics*, vol. 22, no. 1, p. 561, 2021.

[4] T. Thanapattheerakul, W. Engchuan, and J. H. Chan, "Predicting the effect of variants on splicing using convolutional neural networks," *PeerJ*, vol. 8, p. e9470, 2020.

[5] V. Akpokiro, H. M. A. M. Chowdhury, S. Olowofila, R. Nusrat, and O. Oluwadare, "Cnnsplice: Robust models for splice site prediction using convolutional neural networks," *Computational and Structural Biotechnology Journal*, vol. 21, pp. 3210–3223, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S200103702300212X

[6] R. Wang, J. Xu, X. Huang, W. Qi, and Y. Zhang, "Splicescanner: An accurate and interpretable deep learning-based method for splice site prediction," in *Advanced Intelligent Computing Technology and Applications*, D.-S. Huang, P. Premaratne, B. Jin, B. Qu, K.-H. Jo, and A. Hussain, Eds. Singapore: Springer Nature Singapore, 2023, pp. 447–459.

[7] S. Albaradei, A. Magana-Mora, M. Thafar, M. Uludag, V. B. Bajic, T. Gojobori, M. Essack, and B. R. Jankovic, "Splice2deep: An ensemble of deep convolutional neural networks for improved splice site prediction in genomic dna," *Gene*, vol. 763, p. 100035, 2020, articles initially published in Gene: X 5, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2590158320300097

[8] A. L. Bazinet and M. P. Cummings, "A comparative evaluation of sequence classification programs," *BMC Bioinformatics*, vol. 13, no. 92, 2012.

[9] K. Jaganathan *et al.*, "Predicting splicing from primary sequence with deep learning," *Cell*, vol. 176, no. 3, pp. 535–548, 2019.

[10] K. H. Chao *et al.*, "Splam: a deep-learning-based splice site predictor that improves spliced alignments," *Genome Biology*, vol. 25, 2024.

[11] A. Dutta *et al.*, "Deep learning models for identification of splice junctions across species," *bioRxiv preprint*, 2021.

[12] A. Chan *et al.*, "Rna alternative splicing prediction with discrete compositional energy network," in *ACM CHIL*, 2021.

[13] T. Abd-Alhalem *et al.*, "A survey on deep learning methods for dna sequence classification," *Journal of Bioinformatics and Computational Biology*, 2021.

[14] H. Lee, S. Yeom, and S. Kim, "BP-GAN: Interpretable human branch-point prediction using attentive generative adversarial networks," *IEEE Access*, vol. 8, pp. 93 763–93 772, 2020.

[15] A. Aparajita *et al.*, "Deep learning models for identification of splice junctions across species," *bioRxiv preprint*, 2021.

[16] "Molecular Biology (Splice-junction Gene Sequences)," UCI Machine Learning Repository, 1991, DOI: https://doi.org/10.24432/C5M888.

[17] R. Abdullah Al, N. Mohammad Navid, K. S. Sharif, H. Al-Amin, and A. Rudmila, "A comprehensive framework for advanced machine learning and deep learning models in cervical cancer prediction," in *2024 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*, 2024.

[18] B. P. Ghosh, T. Imam, N. Anjum, M. T. Mia, C. U. Siddiqua, K. S. Sharif, M. M. Khan, M. A. I. Mamun, and M. Z. Hossain, "Advancing chronic kidney disease prediction: Comparative analysis of machine learning algorithms and a hybrid model," *Journal of Computer Science and Technology Studies*, vol. 6, no. 3, pp. 15–21, 2024.

[19] K. S. Sharif, M. M. Uddin, and M. Abubakkar, "Neurosignal precision: A hierarchical approach for enhanced insights in parkinson's disease classification," in *2024 International Conference on Intelligent Cybernetics Technology Applications (ICICyTA)*, 2024, pp. 1244–1249.

[20] A. A. Rakin, K. S. Sharif, M. N. Nayyem, M. A. H. Raju, R. Arafin, and M. Z. Hossain, "Advancing cervical cancer risk stratification via ensemble learning models integrated with shap-based interpretability methods," in *2024 IEEE International Conference on Computing (ICOCO)*, 2024, pp. 559–564.