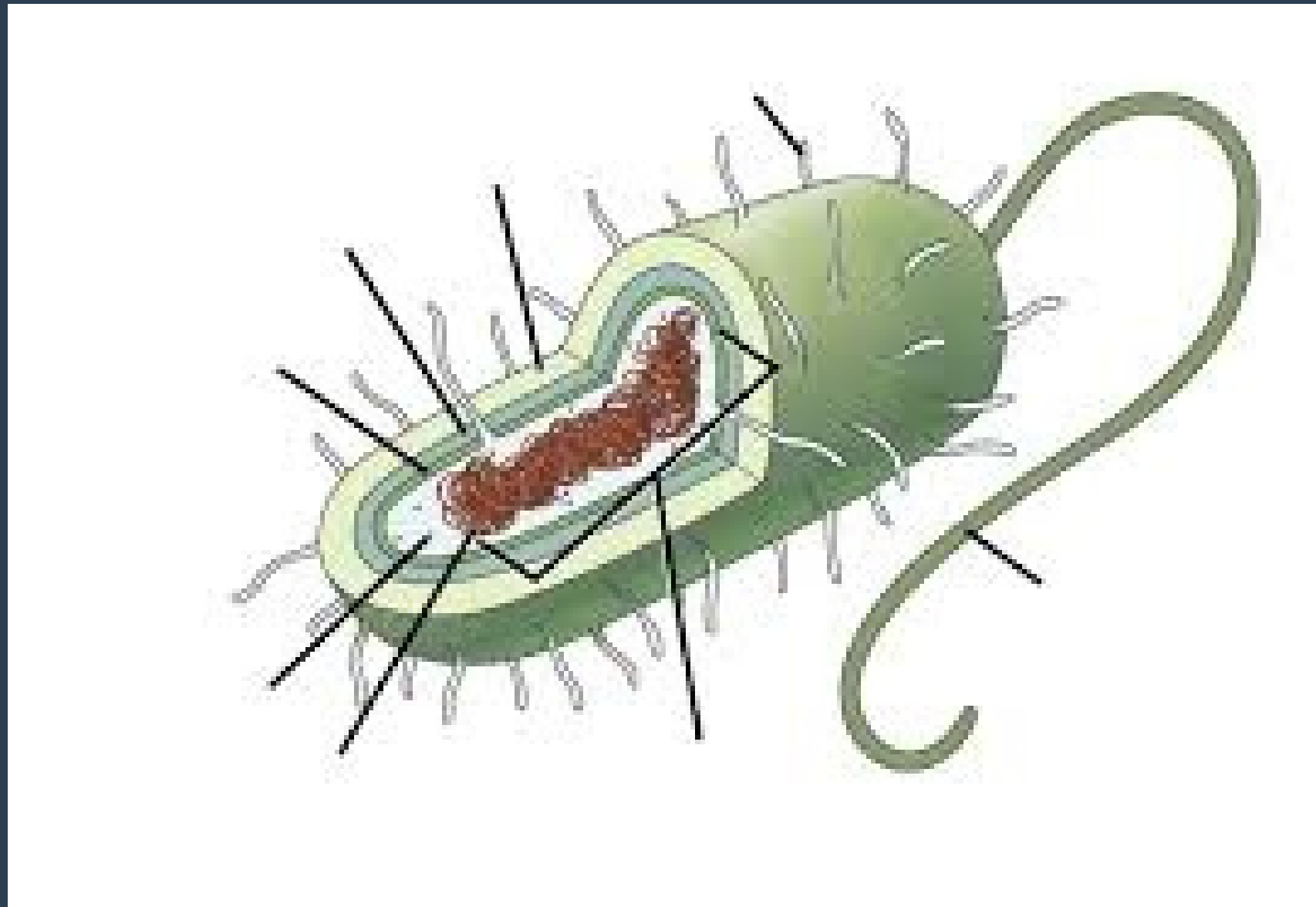# Promoter Recognition in Prokaryotes Using Machine Learning Models

Prepared by

Archil zhghenti

Nika Macharadze

Konstantine Bakhutashvili
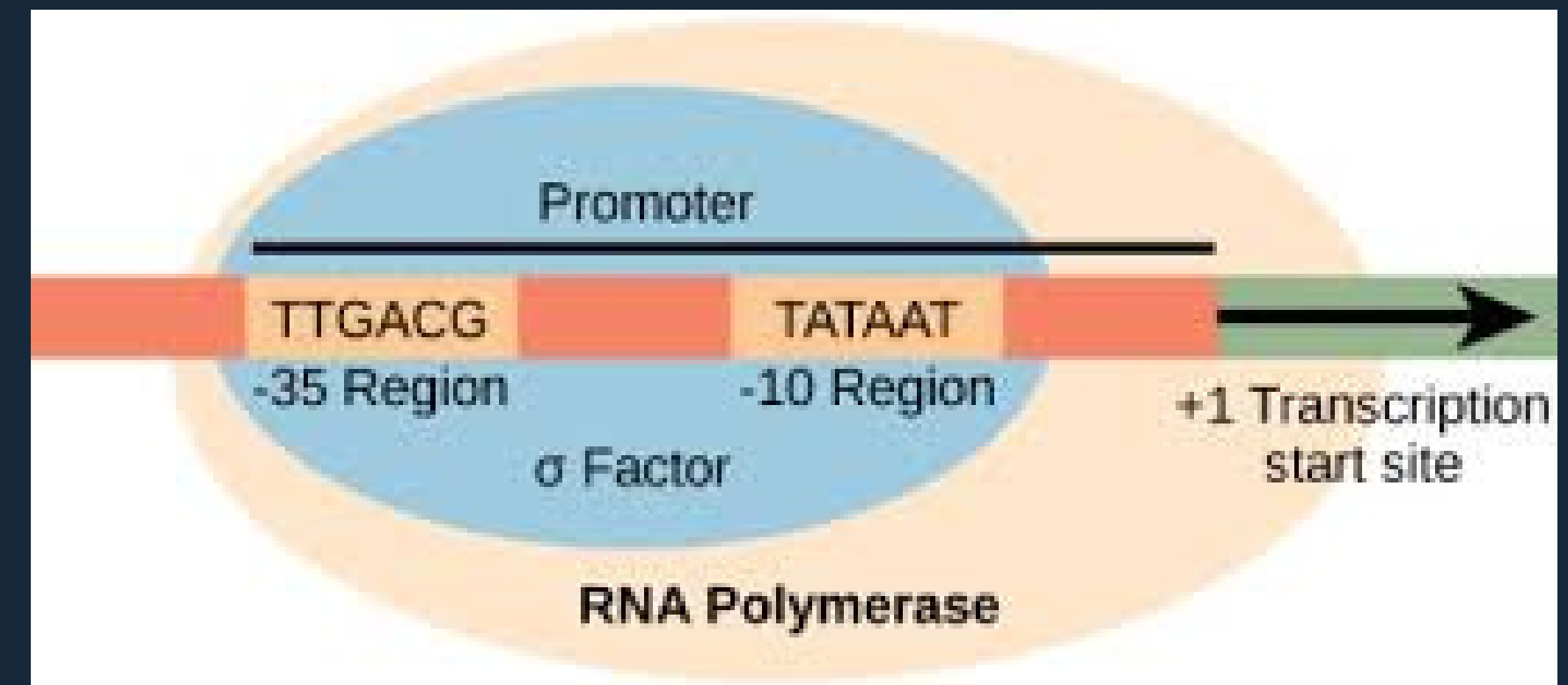
Recognition of prokaryotic and eukaryotic promoters using convolutional deep learning neural networks

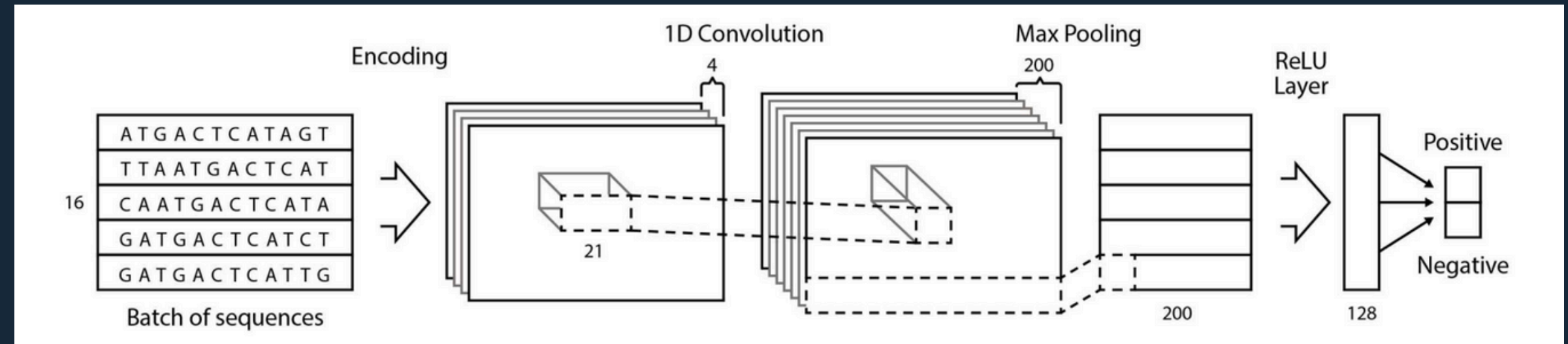https://pmc.ncbi.nlm.nih.gov/articles/PMC5291440/pdf/pone.0171410.pdf

Gene expression starts at promoters. If we can't find promoters accurately, we can't fully understand how genes are regulated. Despite many genomes being sequenced, promoter identification is still a difficult computational problem.

What Is a Promoter?

Promoter = DNA region where transcription starts
Located upstream of genes
Controls when, where, and how strongly a gene is expressed

CNN-based Promoter Recognition

Organism	Sn	Sp	CC	CNN architecture
Escherichia coli s70	0.90	0.96	0.84	100,7, 0 / 150, 21, 12
Bacillus subtilis	0.91	0.95	0.86	100,15, 2 / 250, 17, 2

- Uses Convolutional Neural Networks (CNNs)
- Input: raw DNA sequence only
- No hand-crafted biological features
- Trained on:
- Escherichia coli σ70 promoters
- Bacillus subtilis promoters

$$Sn = TP/(TP + FN)$$

$$Sp = TN/(TN + FP)$$

$$AC = (TP + TN)/(TN + TP + FN + FP)$$

$$CC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Transformer Model

DNA as a language

DNA is a sequence over a 4-letter alphabet: A, C, G, T

Can be modeled similarly to natural language

Enables use of NLP-based deep learning models

From nucleotides to "words"

DNA has no spaces or punctuation

Unlike text, there are no natural word boundaries
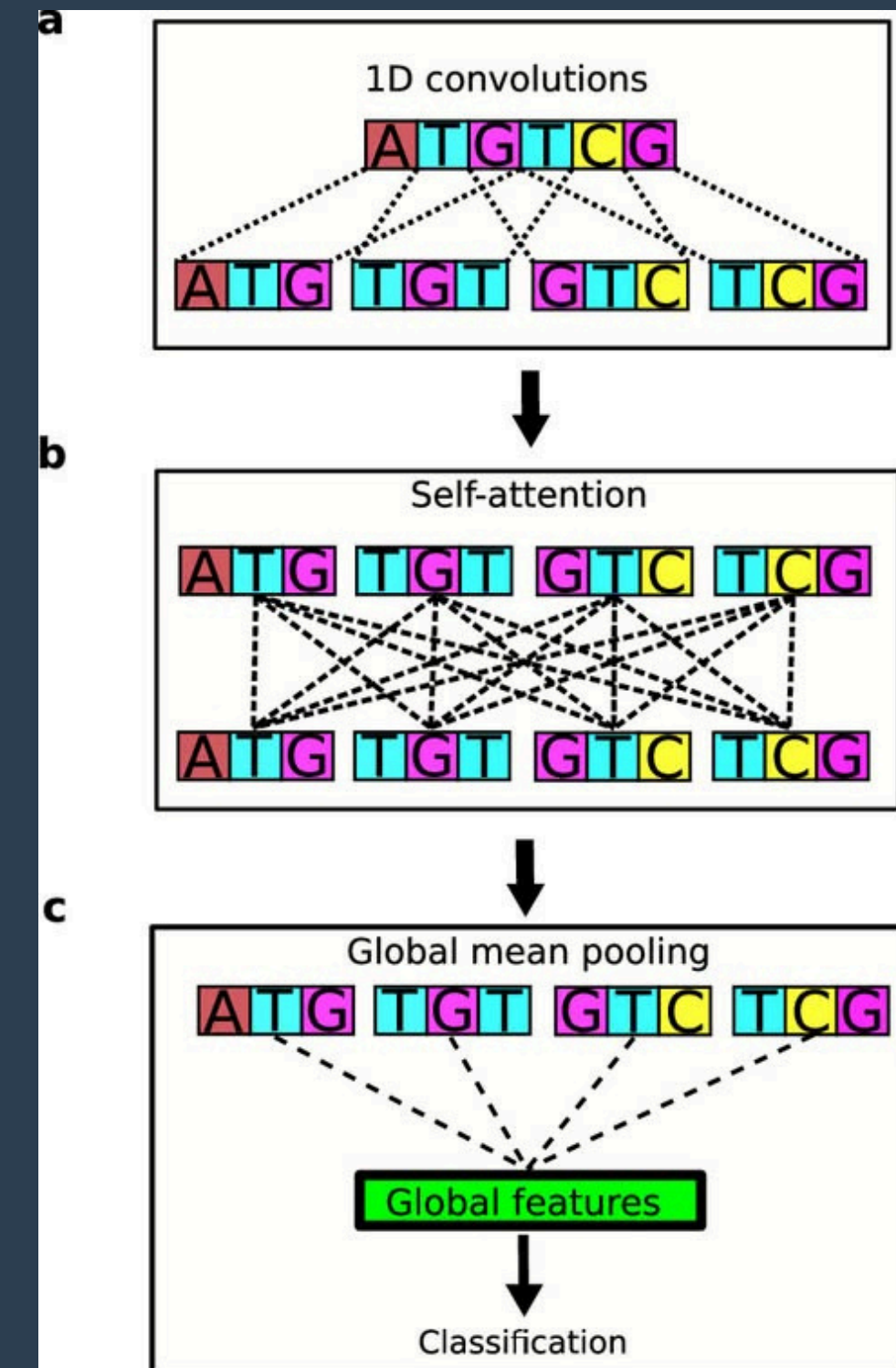
We create "words" using k-mers

Model approach

Nucleotide embedding layer

1D Convolution to extract k-mers

Transformer processes learned sequence representations

Enables motif discovery without biological priors
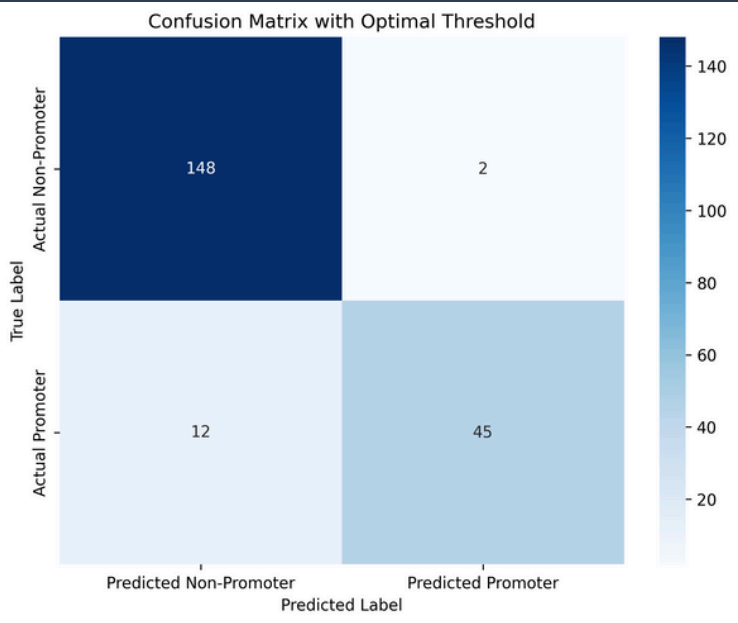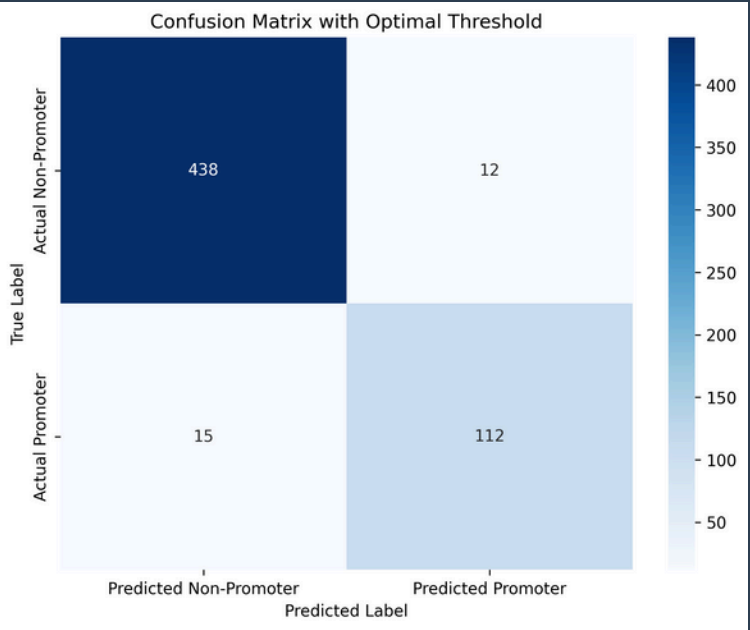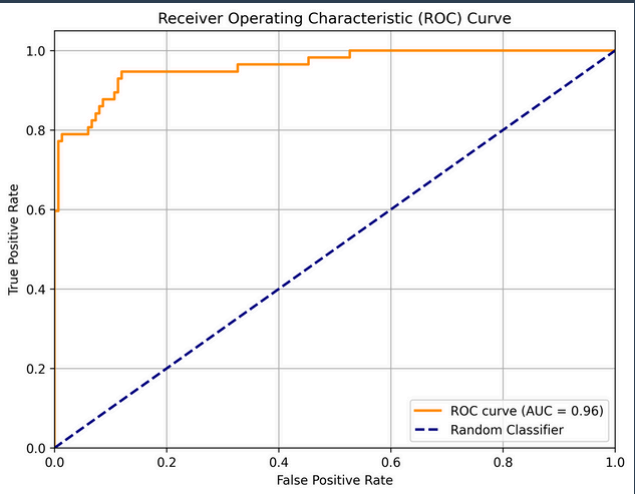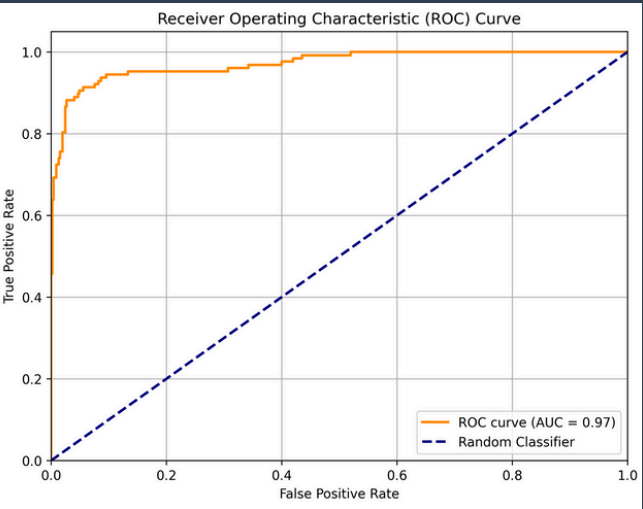
Performance Evaluation

Ecoli

Positi	k-mer	Impor
47	GCAA	0.23
51	ATTG	0.228
49	AAATT	0.226
50	AATT	0.223
46	GGCA	0.215
53	TGAT	0.211
43	ACAG	0.204
63	CTTTT	0.199
52	TTGAT	0.198
65	TTTAA	0.19

Bacillus

Position	k-mer	Importance
54	TAAAGGG	0.998
50	ACAATAA	0.995
52	AATAAAG	0.994
47	ACTACAA	0.994
48	CTACAAT	0.992
51	CAATAAA	0.96
49	TACAATA	0.833
46	GACTACA	0.681
53	ATAAAGG	0.288
45	TGACTAC	0.17

Metric	Value
Accuracy	**95.32%**
Precision	90.32%
Recall (Sensitivity)	88.19%
Specificity	97.33%
F1-score	89.24%
Matthews Correlation	0.863

Metric	Value
Accuracy	**93.24%**
Precision	95.74%
Recall (Sensitivity)	78.95%
Specificity	98.67%
F1-score	86.54%
Matthews Correlation	0.828

Class Distribution in Escherichia coli s70 Dataset

Class Distribution in Bacillus subtilis Dataset

Receiver Operating Characteristic (ROC) Curve — ROC curve (AUC = 0.97)

Receiver Operating Characteristic (ROC) Curve — ROC curve (AUC = 0.96)

Confusion Matrix with Optimal Threshold

Confusion Matrix with Optimal Threshold
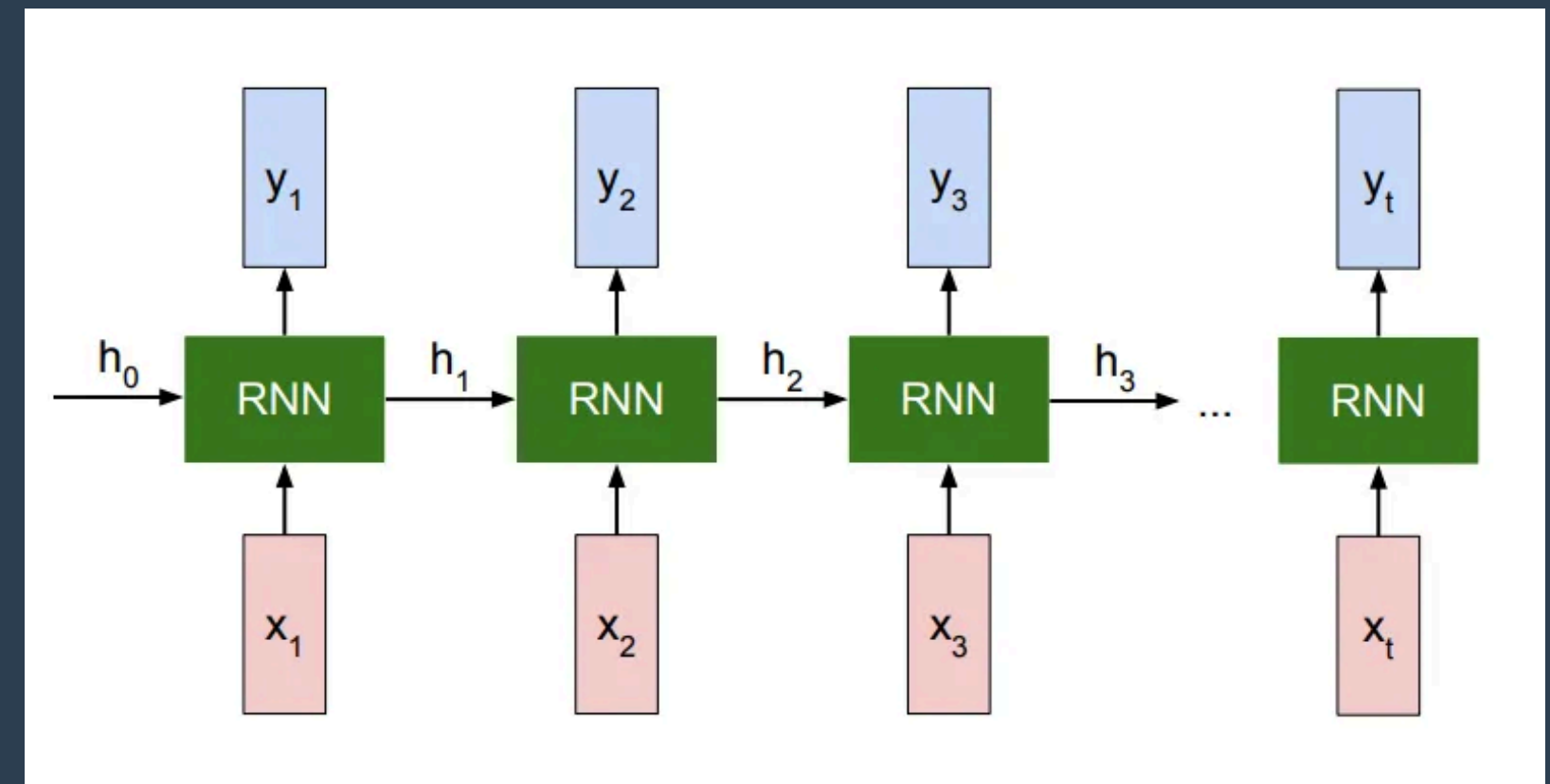
RNN vs LSTM for DNA Promoter Classification

Initial Approach: Vanilla RNN

Why RNN Failed:

- Vanishing Gradient Problem: DNA sequences, even at 81 base pairs, contain long-range dependencies. RNNs struggle with sequences of this length because gradients diminish exponentially as they backpropagate through time, making it difficult to learn relationships between distant nucleotides.
- Limited Memory Capacity: Vanilla RNNs have a simple hidden state that gets overwritten at each time step. This makes it challenging to retain important information from earlier positions in the sequence - crucial for identifying promoter motifs that may be separated by several base pairs.
- DNA-Specific Challenges: Promoter regions often contain multiple regulatory elements (TATA box, CAAT box, GC-rich regions) that need to be recognized in relation to each other. The RNN's architecture lacks the gating mechanisms necessary to selectively preserve or forget information about these critical patterns.
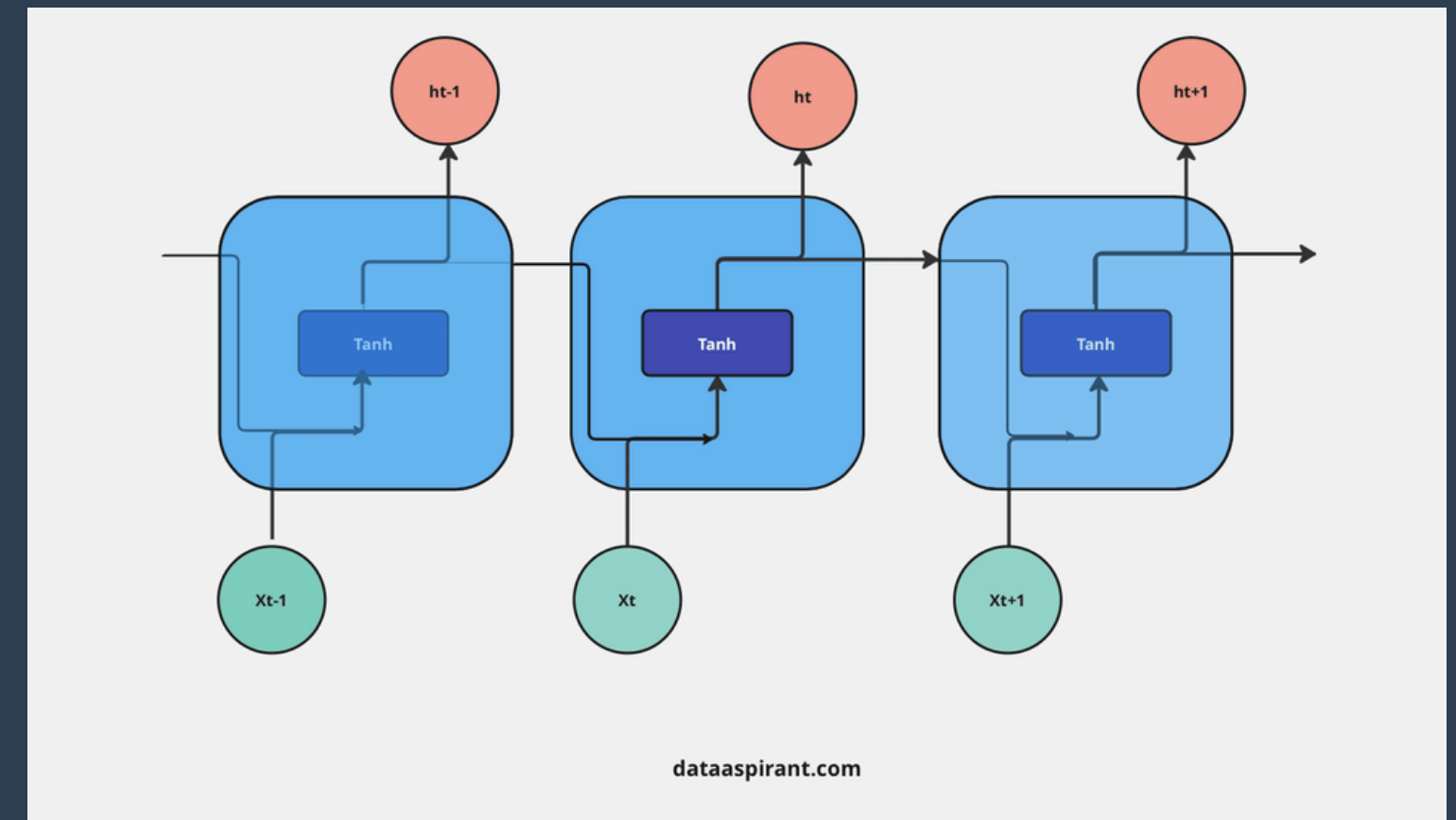
RNN vs LSTM for DNA Promoter Classification

Transition to LSTM

Why LSTM Works for DNA Sequence Analysis

- Gating Mechanisms: LSTMs employ three specialized gates:
  - Forget Gate: Decides what information to discard from the cell state (e.g., non-informative nucleotides)
  - Input Gate: Controls what new information to add to the cell state (e.g., newly encountered motifs)
  - Output Gate: Determines what information to output from the cell state
  - These gates enable the model to selectively retain important features (like TATA boxes at position -30) while ignoring noise.
- Cell State Memory: Unlike RNNs, LSTMs maintain a separate cell state that flows through the entire sequence with minimal modifications. This "memory highway" allows long-range dependencies between distant promoter elements to be preserved effectively.
- Gradient Flow: The cell state provides a path for gradients to flow backward without vanishing, enabling the model to learn patterns across the entire DNA sequence length.

dataaspirant.com

RNN vs LSTM for DNA Promoter Classification

LSTM Architecture
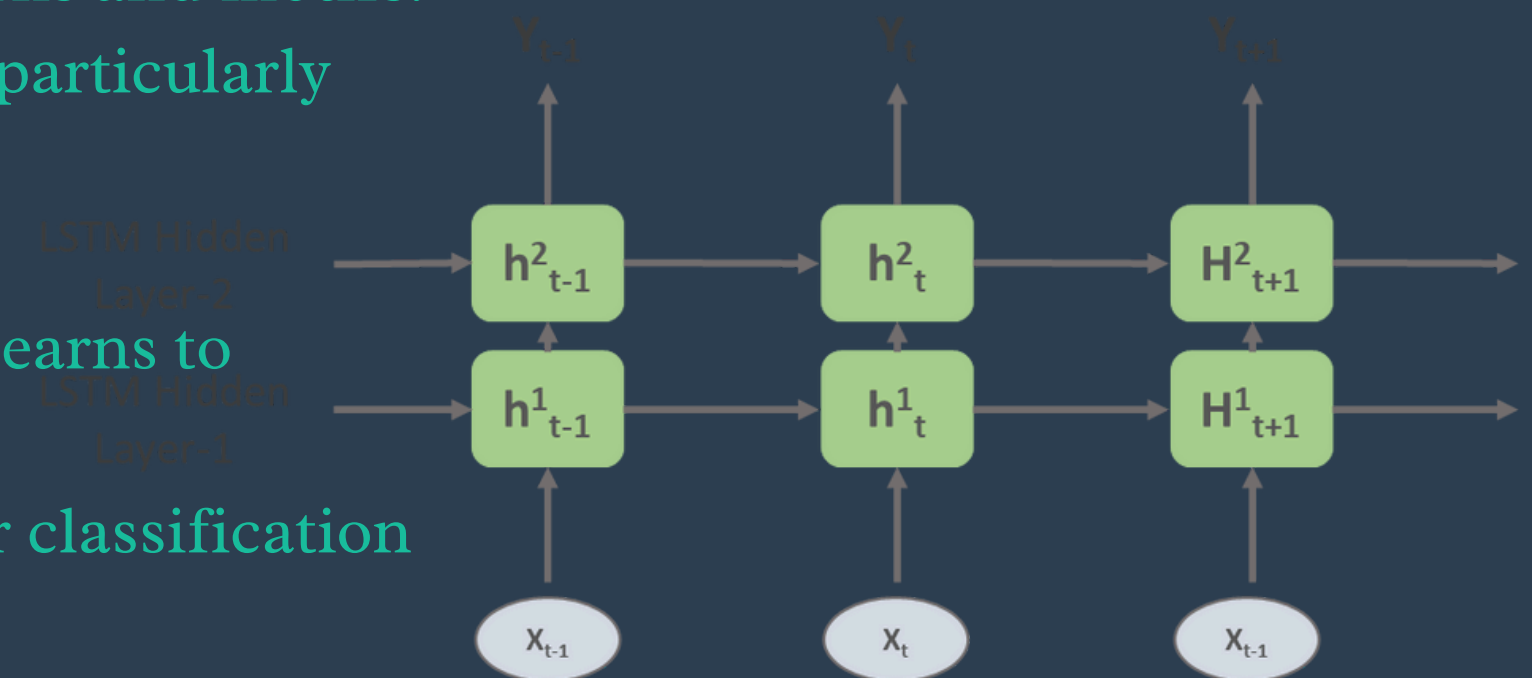
Core LSTM Configuration
- 64 hidden dimentions - provides sufficient capacity to learn complex patterns in 81bp sequences without overfitting
- Two stacked LSTM layers create a hierarchical representation - the first layer learns basic nucleotide patterns, while the second layer captures higher-level motif combinations
- bidirectional - processes sequences in both forward and reverse directions

DNA-Specific Enhancements
- A convolutional layer precedes the LSTM to detect local k-mer patterns and motifs.
- Layer normalization stabilizes training and accelerates convergence, particularly important for deep bidirectional architectures

Advanced Features
- Instead of using just the final hidden state, an attention mechanism learns to weight the importance of each position in the sequence.
- A fully-connected layer with 64 units after the LSTM adds non-linear classification capacity
- 30% dropout prevents overfitting by randomly dropping connections during training

RNN vs LSTM for DNA Promoter Classification

LSTM Architecture

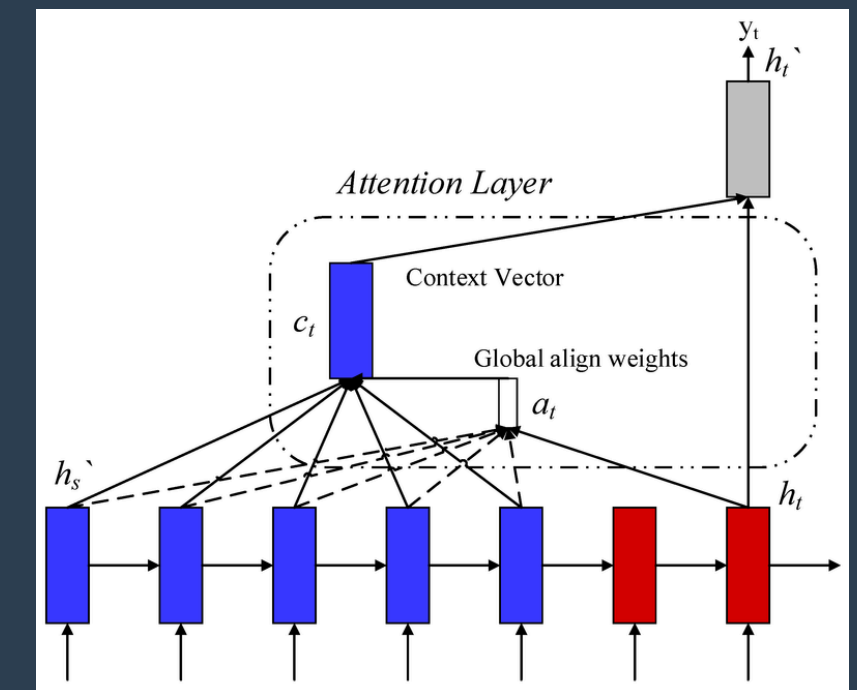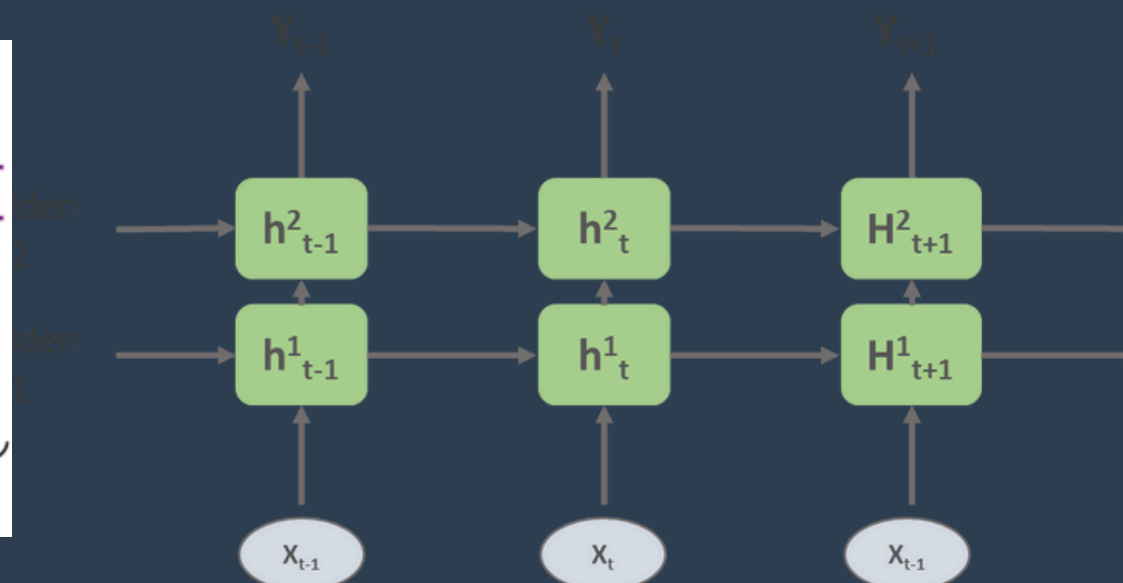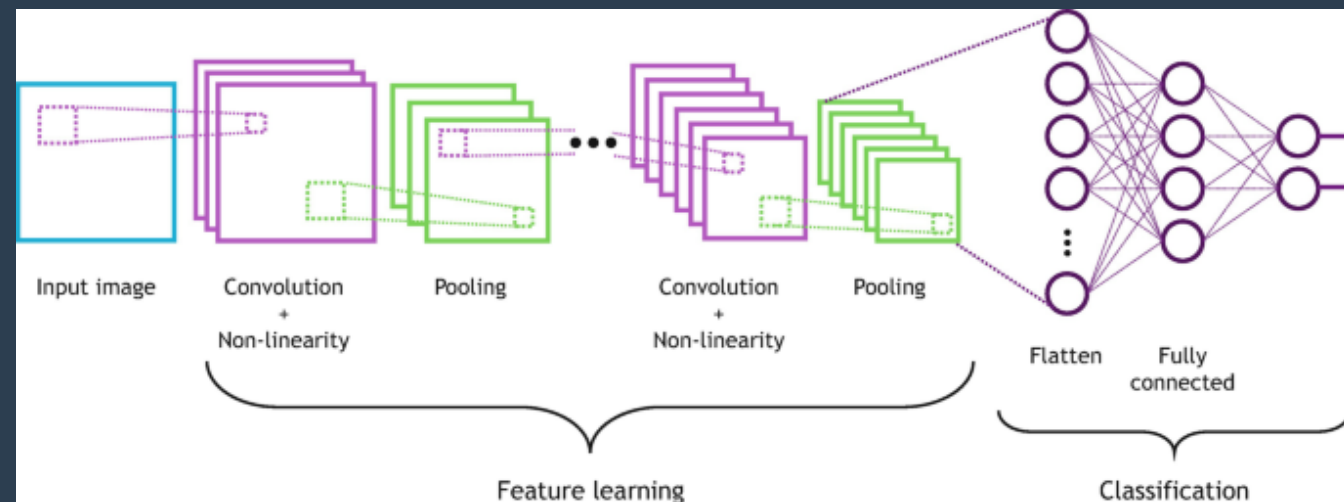Hierarchical Feature Learning: CNN → LSTM → Attention
- creates a pipeline that moves from local motifs to sequential dependencies to position-specific importance

Selective Attention:
- The attention mechanism automatically identifies the most discriminative positions, mimicking how biologists focus on core promoter elements when analyzing sequences

Robust Regularization
- The combination of dropout and layer normalization prevents overfitting while maintaining model expressiveness, critical when working with limited biological datasets

LSTM Performance Evaluation

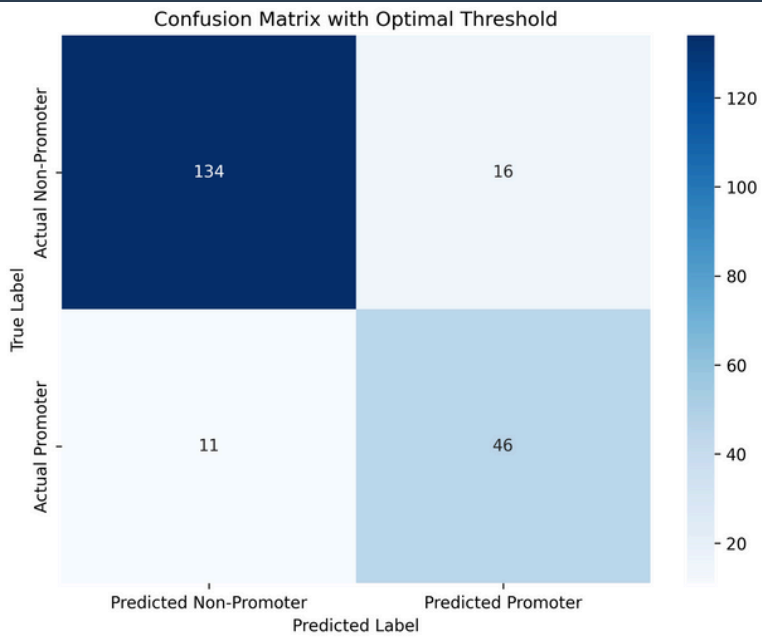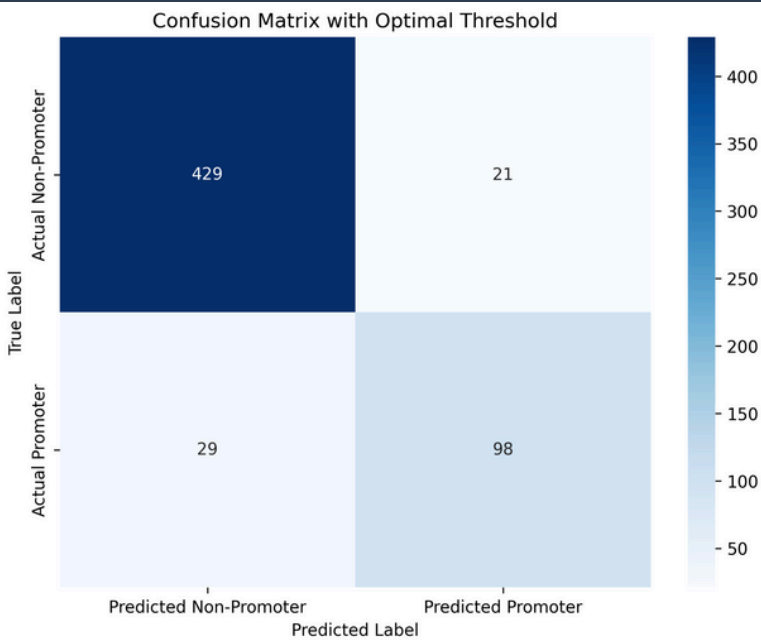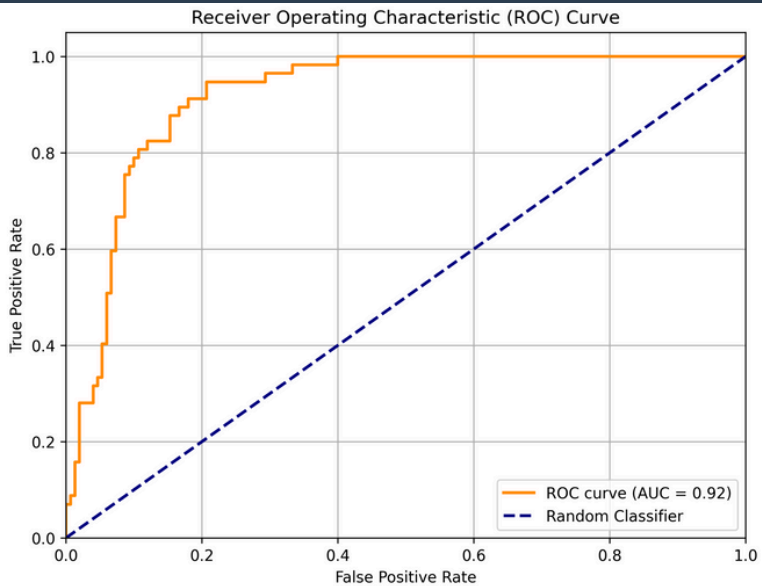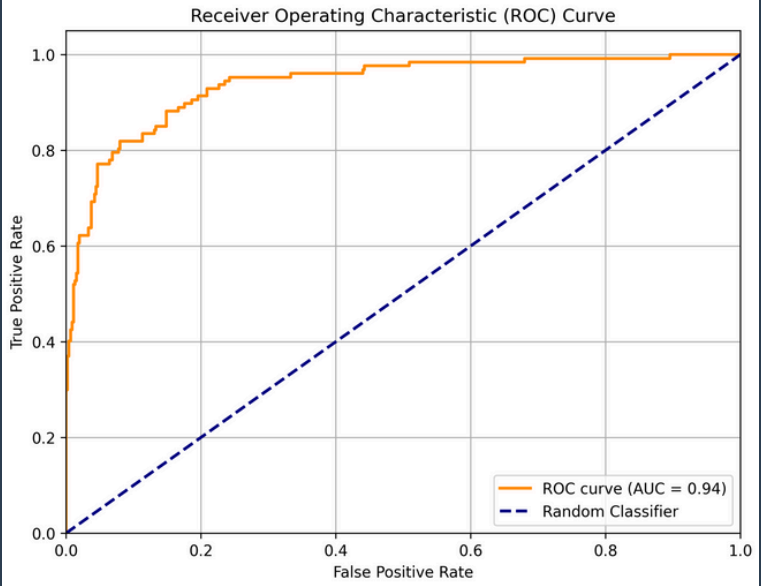## Ecoli

Position	k-mer	Importance
49	AAATTGA	0.147
50	AATTGAT	0.146
47	GCAAATT	0.141
39	ACGGACA	0.132
8	AAAATAT	0.131
52	TTGATGA	0.13
64	TTTTAAA	0.13
63	CTTTTAA	0.128
6	GAAAAAT	0.128
65	TTTAAAC	0.126

Bacillus

Position	k-mer	Importance
67	GAAAATT	0.517
56	AAGGGGA	0.415
69	AAATTTC	0.404
68	AAAATTT	0.366
41	ATAGTGA	0.358
66	CGAAAAT	0.286
26	TGTACAA	0.258
52	AATAAAG	0.253
58	GGGGATA	0.25
32	AATTACA	0.247

Ecoli

Class Distribution in Escherichia coli s70 Dataset

Receiver Operating Characteristic (ROC) Curve — ROC curve (AUC = 0.94), Random Classifier

Confusion Matrix with Optimal Threshold

Bacillus

Class Distribution in Bacillus subtilis Dataset

Receiver Operating Characteristic (ROC) Curve — ROC curve (AUC = 0.92), Random Classifier

Confusion Matrix with Optimal Threshold

Metric	Value
Accuracy	**91.30%**
Precision	82.35%
Recall (Sensitivity)	77.16%
Specificity	95.33%
F1-score	79.67%
Matthews Correlation Coefficient (MCC)	0.742

Metric	Value
Accuracy	**86.90%**
Precision	74.19%
Recall (Sensitivity)	80.70%
Specificity	89.33%
F1-score	77.31%
Matthews Correlation	0.682

LSTM Performance Evaluation

Combined

Position	k-mer	Importance
29	ACAAATT	0.17
14	TATTTTT	0.166
21	TTGTCTG	0.125
11	TTCTATT	0.119
12	TCTATTT	0.106
13	CTATTTT	0.095
16	TTTTTTT	0.089
27	GTACAAA	0.066
47	ACTACAA	0.065
54	TAAAGGG	0.064

Metric	Value
Accuracy	**85.56%**
Precision	68.81%
Recall (Sensitivity)	69.94%
Specificity	90.33%
F1-score	69.37%
Matthews Correlation Coefficient (MCC)	0.599

Combined

Class Distribution in Bacillus subtilis Dataset

Receiver Operating Characteristic (ROC) Curve

Confusion Matrix with Optimal Threshold

XGBoost Model

Strong baseline for tabular features
Robust and interpretable compared to deep models

Feature Engineering (DNA → XGBoost Input)

Flattened One-Hot Encoding (used):
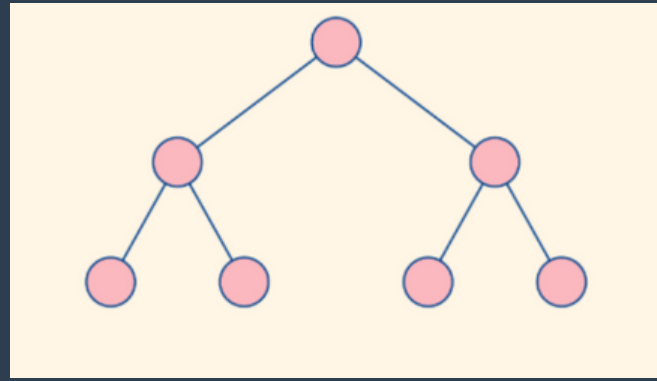Each nucleotide (A, C, G, T) → 4-dimensional vector
Sequence length LLL → $L×4L$ \times $4L×4$ → flattened to a
4L-dimensional vector
Preserves positional information, critical for promoter motifs
Output size: 1004 features per sequence

Dataset is class-imbalanced

Computed scale_pos_weight = (# negative samples) / (# positive samples)
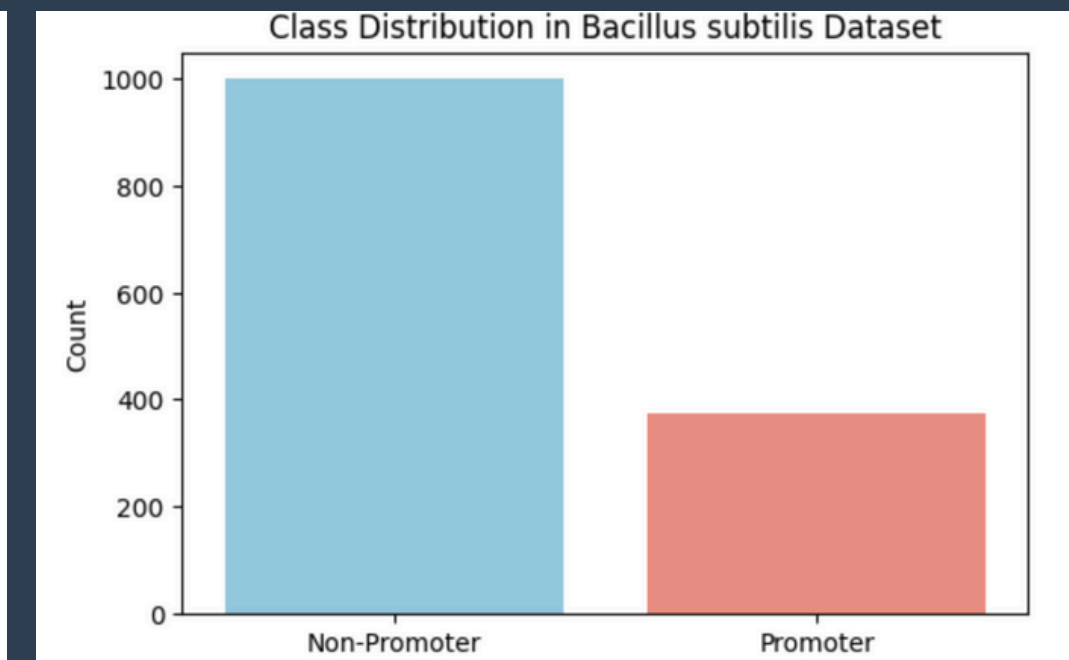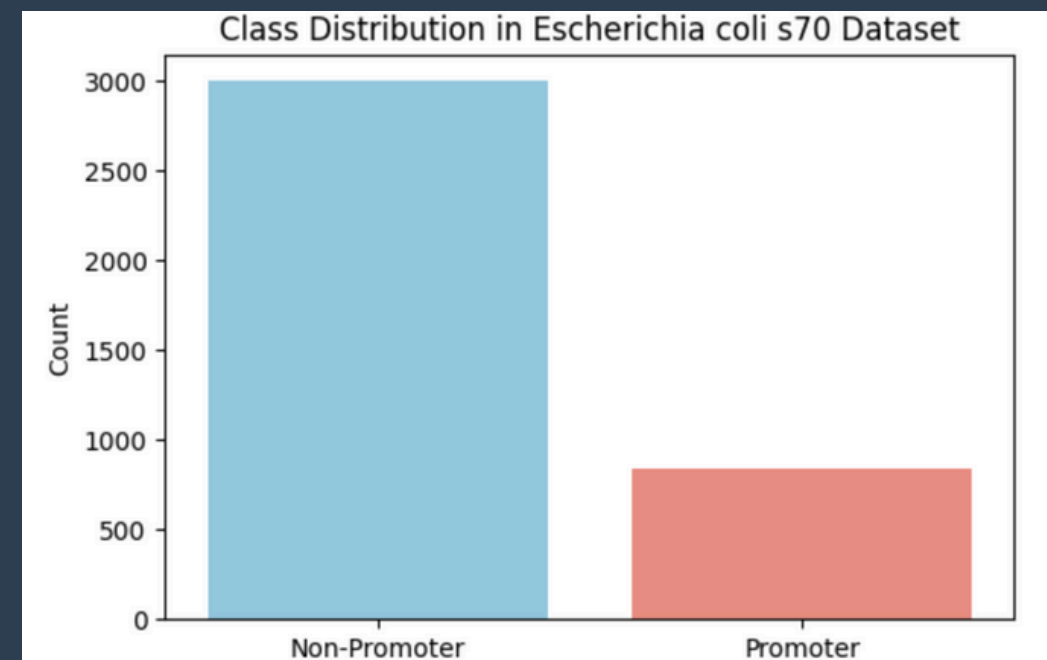
FASTA (promoter / non-promoter)
|
v
One-hot encoding + flatten
(DNA sequence → numeric vector)
|
v
Train XGBoost classifier
(use scale_pos_weight for imbalance)
|
v
Predictions + metrics (Sn, Sp, MCC)

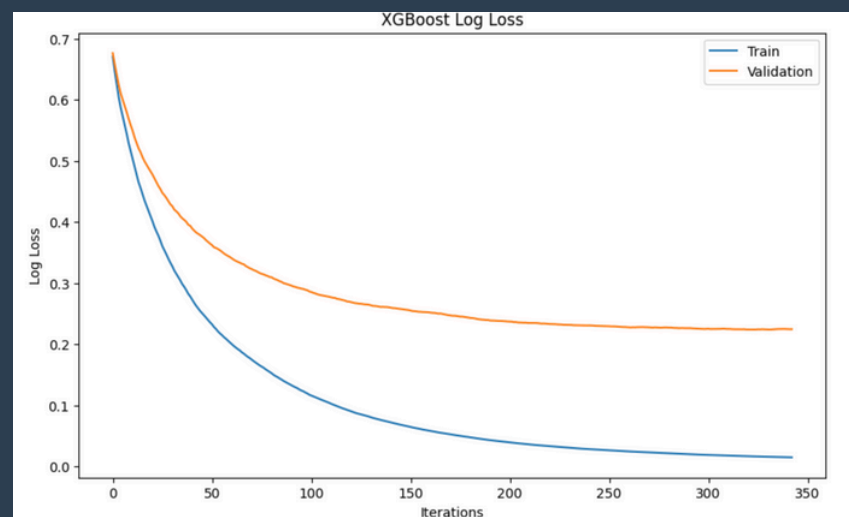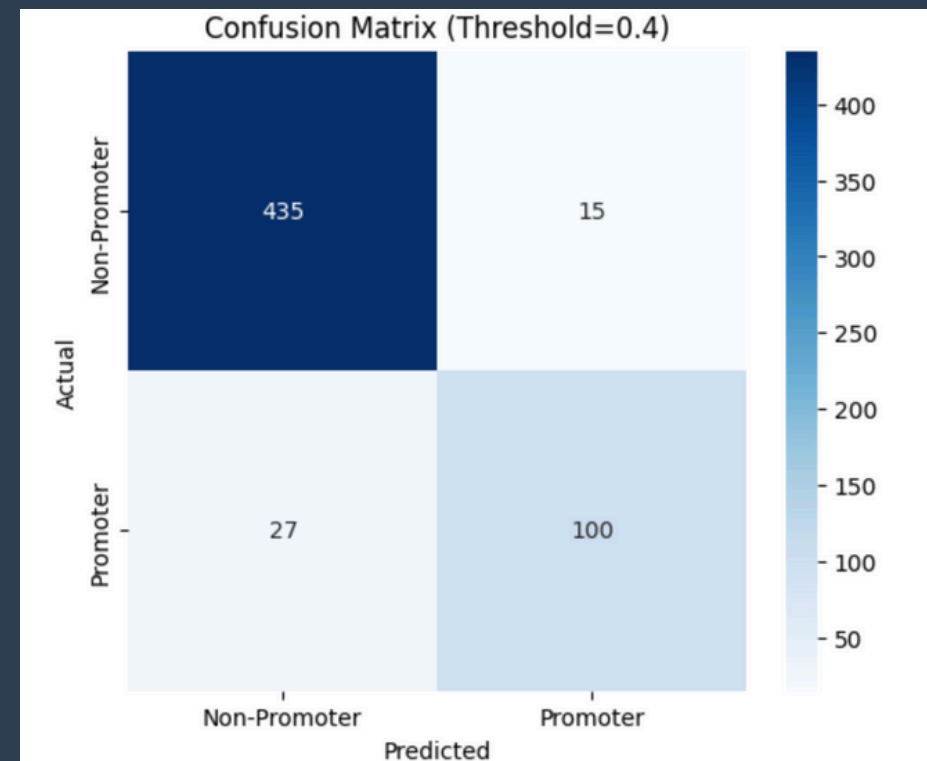# Performance Evaluation
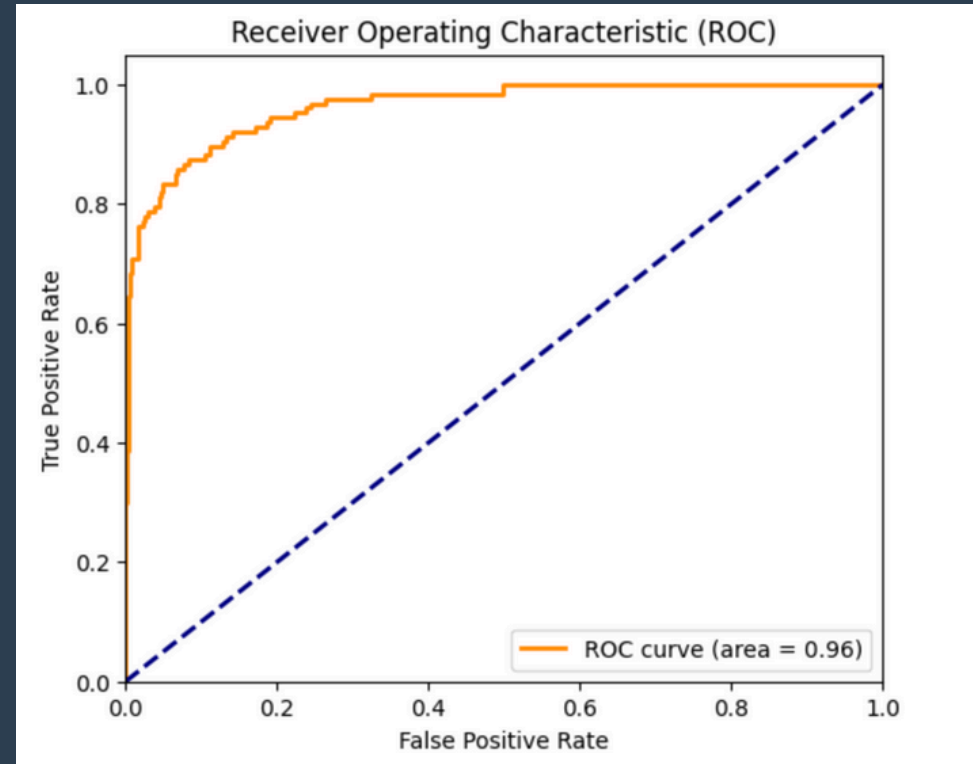
Ecoli

--- Threshold: 0.3 ---
Accuracy: 0.9116
Sensitivity (Sn): 0.8346
Specificity (Sp): 0.9333
CC (MCC): 0.7497

--- Threshold: 0.4 ---
Accuracy: 0.9272
Sensitivity (Sn): 0.7874
Specificity (Sp): 0.9667
CC (MCC): 0.7821

--- Threshold: 0.5 ---
Accuracy: 0.9341
Sensitivity (Sn): 0.7559
Specificity (Sp): 0.9844
CC (MCC): 0.8010
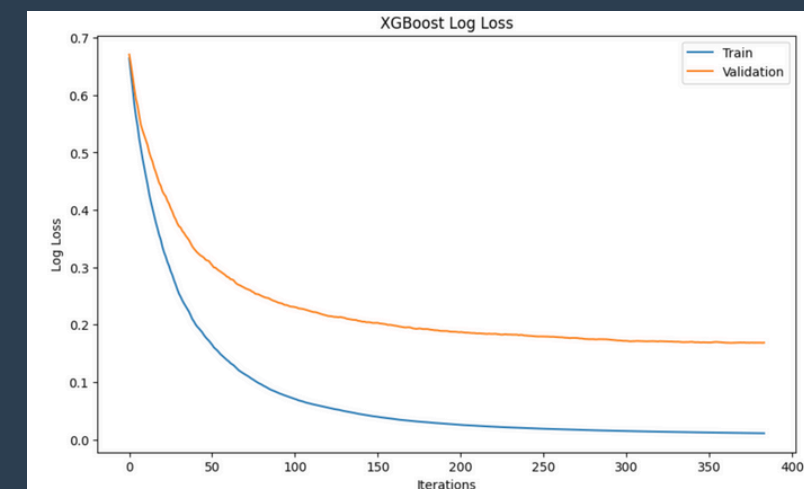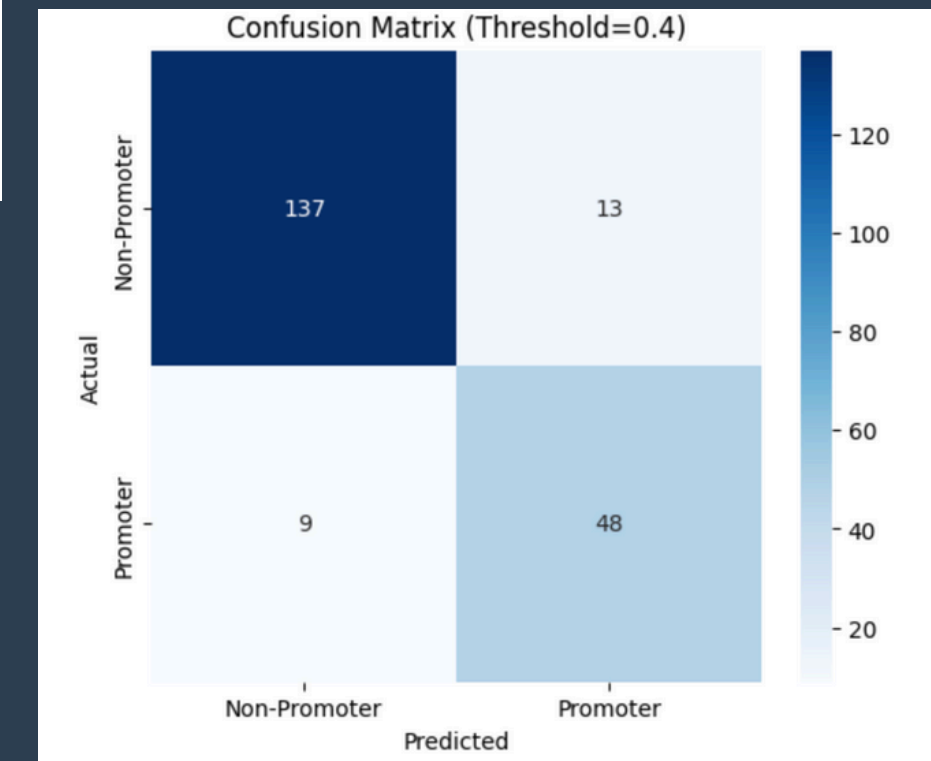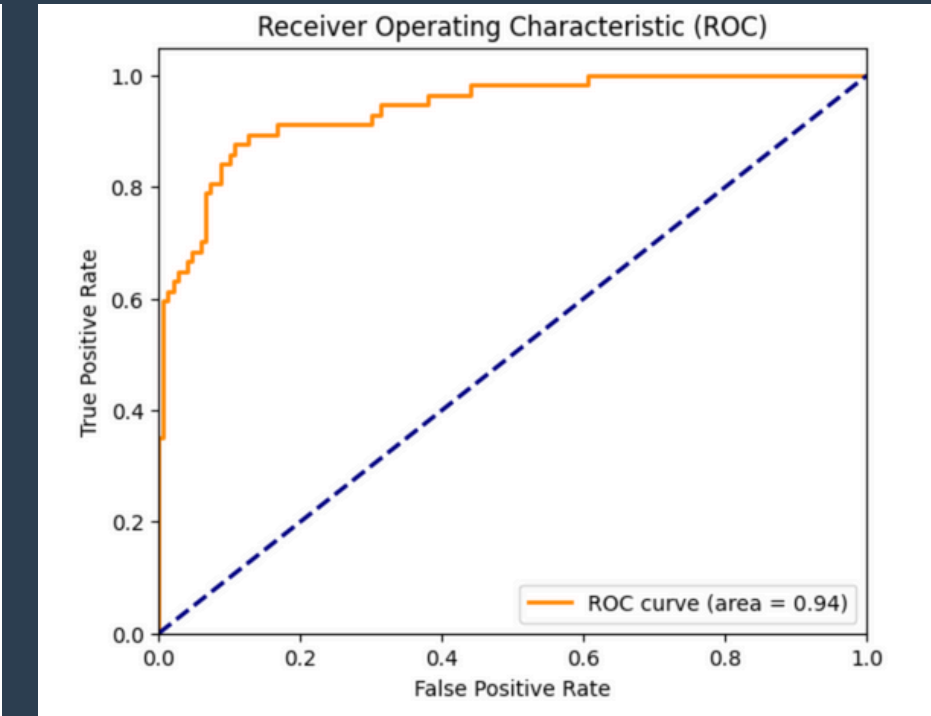
Bacillus

--- Threshold: 0.3 ---
Accuracy: 0.8889
Sensitivity (Sn): 0.8772
Specificity (Sp): 0.8933
CC (MCC): 0.7386

--- Threshold: 0.4 ---
Accuracy: 0.8937
Sensitivity (Sn): 0.8421
Specificity (Sp): 0.9133
CC (MCC): 0.7402

--- Threshold: 0.5 ---
Accuracy: 0.8841
Sensitivity (Sn): 0.8070
Specificity (Sp): 0.9133
CC (MCC): 0.7128

Models Comparasion

Organism	Model	Accuracy	Sensitivity (Sn)	Specificity (Sp)	MCC / CC
E. coli	Real Model	–	0.9	0.96	0.84
E. coli	Transformer	0.953	0.882	0.973	0.863
E. coli	LSTM	0.913	0.772	0.953	0.742
E. coli	XGBoost	0.927	0.787	0.967	0.782
B. subtilis	Real Model	–	0.91	0.95	0.86
B. subtilis	Transformer	0.932	0.789	0.987	0.828
B. subtilis	LSTM	0.87	0.807	0.893	0.683
B. subtilis	XGBoost	0.894	0.842	0.913	0.74

Thank You for Your Attention