Binary Antimicrobial Resistance Prediction from PBP4 Gene Sequences: Comparing Encoder-Only Transformer Models

Suvinava Basak[1], Terril Joel Nazareth[1]

Abstract

Antimicrobial resistance (AMR) is a major global health threat, underscoring the need for rapid, accurate resistance prediction methods. This study presents a comprehensive comparison of different transformer models for binary AMR prediction using *pbp4* gene sequences from *Staphylococcus aureus* (against *cefoxitin*). We evaluated multiple transformer models, including a custom transformer built from scratch, the pre-trained DNABERT-6 model under three settings (full fine-tuning, frozen backbone with trainable head, and with parameter-efficient Low-Rank Adaptation), and the Nucleotide Transformer. To tackle class imbalance common in AMR data, we employed class weighting alongside a novel max-margin loss combined with focal binary cross-entropy loss.

All models were trained and tested on datasets of 150 *S. aureus* sequences (40 non-resistant, 110 resistant). Our custom transformer included genomic-specific tokenization and attention mechanisms similar to DNABERT. Our experiments highlighted the efficiency of LoRA fine-tuning, achieving competitive results with significantly fewer trainable parameters. The best-performing model, DNABERT-6 (in full fine-tune mode), achieved an accuracy of 100%, precision 100%, recall 100%, F1-score 100%, and ROC-AUC 100% on held-out test data. Parameter-efficient approaches, especially LoRA-adapted DNABERT-6, showed promise for scalable, resource-conscious AMR prediction by achieving comparable performance while utilizing only 0.66% of the trainable parameters. The accuracy, precision, recall, F1-score, and ROC-AUC achieved by LoRA-adapted DNABERT-6 is

[1] TU Braunschweig, Master's program in Data Science, Germany

93.33%, 100%, 91.67%, 95.65%, and 100% respectively.

Our results demonstrate the potential of transformer-based models, particularly when combined with parameter-efficient fine-tuning like LoRA, for genomic AMR prediction. This work advances computational methods for rapid resistance detection, by providing a comprehensive comparison of modern transformer architectures, potentially improving clinical decision-making and antibiotic stewardship.

Keywords: Antimicrobial resistance, Staphylococcus aureus, pbp4 gene, encoder-only transformers, DNABERT-6, Nucleotide Transformer

1. Introduction

Antimicrobial resistance (AMR) has emerged as a global health threat in the 21st century, ranked by the World Health Organization (WHO) among the top ten dangers to humanity [1, 2]. The rise and spread of resistant bacteria have undermined antibiotic effectiveness, turning once-treatable infections into life-threatening illnesses and increasing mortality [3]. In the U.S. alone, AMR infections affect over 2.8 million people yearly, causing more than 35,000 deaths and raising mortality rates in various patient groups [4]. Globally, if unchecked, AMR could cause 10 million deaths annually by 2050 and incur over $100 trillion in economic costs [5].

Traditional antimicrobial susceptibility detection practices relies mainly on phenotypic methods, taking 24–72 hours to produce results [6]. This delay often leads to inappropriate empirical antibiotic use, risking treatment failure and promoting resistance [7]. This highlights the urgency for a rapid, accurate resistance prediction method, enabling research into molecular methods using bacterial genetics to predict phenotypic resistance within hours [8].

Next-generation sequencing has revolutionized bacterial genome analysis by making sequencing more accessible and affordable, enabling resistance prediction from genomic features [9]. This leverages the genotype-phenotype relationship, where antimicrobial resistance often results from specific genetic mutations detectable via sequence analysis [10]. For β-lactam antibiotics like *cefoxitin* and *aztreonam* (important therapeutic agents), resistance typically involves alterations to penicillin-binding proteins (PBPs) [11]. The *pbp4* gene is crucial for bacterial cell wall synthesis and a key β-lactam target [12, 13]. Mutations or structural changes in *pbp4* can reduce antibiotic binding affinity, leading to resistance through decreased target susceptibility [14, 15].

Staphylococcus aureus is a major human pathogen causing infections ranging from minor skin issues to severe conditions like pneumonia, bacteremia, and endocarditis [16]. These two clinically significant pathogens differ in both resistance mechanisms and epidemiology. In *S. aureus*, *cefoxitin* resistance is often used as a surrogate marker for methicillin resistance, commonly driven by mutations in the *pbp4* gene that lower antibiotic binding affinity [17]. Genomic analysis of these species-specific mechanisms enables the development of more targeted resistance prediction models.

The advent of deep learning has transformed computational biology, with transformer architectures excelling at modeling complex sequence dependencies [18]. Originally successful in natural language processing, models like Google's BERT (Bidirectional Encoder Representations from Transformers) demonstrated strong performance in capturing contextual relationships in sequences [19]. In genomics, recent breakthrough models like DNABERT-6 and Nucleotide Transformer showing how pre-trained transformers can uncover intricate sequence patterns often missed by traditional methods [20, 21]. Still, applying deep learning to AMR prediction brings specific challenges for good performance: genomic datasets are typically high-dimensional, limited in size, and often suffer from class imbalance [22]. This imbalance, where one resistance class dominates, can result in models that perform well overall but fail on critical cases, having severe consequences of false negatives [23].

Transfer learning has emerged as a powerful solution to data scarcity, allowing models pre-trained on large datasets to be adapted for tasks with limited labeled data [24]. In computational biology, it has proven effective, enabling models to learn broad biological patterns and be fine-tuned for downstream applications like resistance prediction [25]. However, the best transfer learning strategy for genomic tasks is still an area of study, with methods ranging from full fine-tuning to parameter-efficient techniques that freeze most weights and adapt only specific components [26].

Parameter-efficient fine-tuning methods, especially Low-Rank Adaptation (LoRA), have become increasingly important for delivering strong performance with minimal computational overhead while preserving the stability of pre-trained models [27]. These methods are particularly valuable in clinical settings, where limited resources and the need for efficient model deployment make full-scale fine-tuning impractical [28].

This study addresses the challenge of predicting antimicrobial resistance from *pbp4* gene sequences by systematically comparing several cutting-edge deep learning architectures. We evaluate a custom transformer model tailored for genomic classification alongside pre-trained models like DNABERT-6 and Nucleotide Transformer, using transfer learning strategies such as full fine-tuning, frozen feature extraction, and parameter-efficient methods. To address class imbalance, we incorporate advanced loss functions that combine max-margin objectives with focal loss. Our rigorous experiments highlight the potential of these models to enable rapid and accurate AMR prediction, with implications for precision medicine and contributions to global efforts against antimicrobial resistance [29].

2. Related Work

2.1. Early Machine Learning Approaches for AMR Prediction

Over the past two decades, machine learning for antimicrobial resistance prediction has come a long way. Early work by Huynen et al. [30] laid the groundwork for genotype-to-phenotype prediction using rule-based systems based on known resistance genes, but these methods depended heavily on known resistance determinants. The field then shifted to machine learning when traditional algorithms like support vector machines, random forests, and logistic regression applied to curated genetic features [31, 32]. For example, Moradigaravand et al. used ensemble methods and k-mer features to predict resistance in *Salmonella* species with over 90% accuracy [31], while Drouin et al. highlighted interpretable models with the Set Covering Machine algorithm [32]. Yet, these approaches still relied on manual feature selection and struggled to capture complex patterns in genomic data. The advent of deep learning marked a paradigm shift, enabling automatic feature learning directly from raw sequence data [33].

2.2. Deep Learning Architectures for Genomic Sequence Analysis

Deep neural networks was first introduced for genomic sequence analysis through convolutional neural networks (CNNs), which excelled at identifying motifs and local patterns [34, 35]. Alipanahi et al. pioneered CNN use to predict DNA- and RNA-binding protein specificities [34], followed by Kelley et al.'s Basset model that scaled CNNs to genome-wide chromatin accessibility prediction [35]. In antimicrobial resistance, Khaledi et al. was the first to apply CNNs with DeepARG to predict resistance genes by DNA sequences as 1D signals, outperforming traditional homology methods [36]. Arango-Argoty et al. improved on this with DeepARG-LS, which used longer sequences to better generalize to novel resistance genes [37]. Despite their strengths, CNNs struggled to capture long-range genetic dependencies. Recurrent neural networks (RNNs), especially bi-directional LSTMs with attention, addressed this by modeling sequence dependencies more effectively. Yang et al. demonstrated their power in predicting resistance in *Mycobacterium tuberculosis*, achieving state-of-the-art performance on benchmark datasets [38]. Including attention mechanisms helped focus on key sequence regions while maintaining global context [39].

2.3. Transformer Architectures in Computational Biology

The transformer architecture, introduced by Vaswani et al. [18], revolutionized sequence modeling with its self-attention and parallel processing. Success of transformer models in natural language processing, like BERT [19] and GPT [40], motivated its adaptation to various sequence analysis tasks in computational biology. Thanks to self-attention, transformers can capture long-range dependencies effectively, making them well-suited for genomic tasks where distant genetic elements interact.

2.3.1. DNABERT and Genomic Language Models

The first major adaptation of BERT for genomics was DNABERT by Ji et al. [20], which used k-mer tokenization to treat overlapping nucleotide sequences as discrete tokens like words. Pre-trained on the human genome with masked language modeling, DNABERT excelled across many genomic prediction tasks. Building on this, Zhou et al. developed DNABERT-2 with better tokenization and more efficient attention, boosting performance further across tasks [41]. Nguyen et al. introduced HyenaDNA, using subquadratic attention to efficiently handle longer genomic sequences while maintaining computational efficiency [42].

2.3.2. Nucleotide Transformer and Multi-Species Models

Dalla-Torre et al. introduced Nucleotide Transformer, extending transformers to multi-species genomic analysis by pre-training on diverse genomes from multiple organisms [21]. This approach overcomes species-specific model limits by learning universal patterns across taxanomic boundaries, boosting generalization and versatility in genomic tasks. Building on this, more sophisticated models like GenSLMs [43] and GENA-LM [44] incorporate new architectures and training techniques to further improve genomic predictions.

2.4. Transfer Learning Paradigms in Genomic Applications

Transfer learning gained popularity in NLP and computational biology due to the scarcity in labeled data [24]. Avsec et al. showed that pre-training on large genomic datasets allows models to be fine-tuned effectively for various downstream tasks, even outperforming training from scratch [25]. Many studies have since explored diverse transfer learning strategies for genomics.

2.4.1. Full Fine-Tuning Strategies

Full fine-tuning, which involves updating all model parameters for specific tasks, is common in genomics but challenging for small datasets, often causing overfitting and poor generalization [45]. Chen et al. showed that careful learning rate control and regularization can reduce overfitting and help adapt genomic transformers effectively [46]. Still, with very small datasets, performance may remain limited.

2.4.2. Feature Extraction and Selective Fine-Tuning Approaches

Alternative methods use pre-trained models as fixed feature extractors, feeding learned representations into task-specific classifiers. This works well with limited data, as shown by Koo and Ploenzke in regulatory genomics [47]. Eraslan et al. showed that layer-wise fine-tuning carefully selected trainable layers helps maintaining trade-off between adaptation capability and overfitting risks [48].

2.5. Parameter-Efficient Fine-Tuning in Genomics

The high computational and storage demands of fine-tuning large transformer models, often with millions or even billions of parameters, have led to the development of more parameter-efficient alternatives.

2.5.1. LoRA Methodology and Applications

Low-Rank Adaptation (LoRA), proposed by Hu et al. [27], offers a particularly efficient solution by adapting pre-trained models with minimal parameter updates. It works by decomposing weight updates into low-rank matrices, reducing the number of trainable parameters significantly while maintaining competitive performance. Li et al. applied LoRA to protein sequence analysis and achieving comparable performance to full fine-tuning, using less than 1% of the parameters [49].

Low-Rank Adaptation (LoRA) is a technique for parameter-efficient fine-tuning of large pre-trained models by injecting trainable low-rank matrices into existing weight layers. Given a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$, LoRA approximates the update ΔW as a product of two low-rank matrices:

$$\Delta W = AB, \quad A \in \mathbb{R}^{d\times r}, \quad B \in \mathbb{R}^{r\times k}, \quad r \ll \min(d,k)$$

Instead of updating W_0 directly, only A and B are learned during fine-tuning, significantly reducing the number of trainable parameters. The adapted weight is:

$$W = W_0 + \Delta W = W_0 + AB$$

where r is the rank hyperparameter controlling the trade-off between model capacity and parameter efficiency.

Other parameter-efficient methods have also proven useful in genomics. Adapter layers by Houlsby et al. [50] insert small trainable modules between transformer layers to adapt to new tasks while retaining pre-trained knowledge. Prefix tuning by Li and Liang [51] and prompt tuning by Lester [52] adjust small sets of task-specific inputs to guide model behavior without modifying most of the model.

2.6. Addressing Class Imbalance in Genomic Machine Learning

Class imbalance represents a major challenge in genomic dataset for machine learning applications. Traditional approaches to addressing class imbalance, including resampling techniques and cost-sensitive learning, have been adapted for genomic applications with varying degrees of success. [53]

2.6.1. Advanced Loss Functions for Imbalanced Data

Recent efforts have developed advanced loss functions to handle class imbalance. Lin et al. proposed focal loss, which down-weights easy examples during training to address class imbalance [54]. Wang et al. adapted this approach for genomic data, improving classification in cancer genomics [55]. Furthermore, Cui et al. introduced class-balanced loss, using effective sample numbers to better handle long-tail distributions in genomic datasets [56]. These methods have proven helpful, especially in highly imbalanced datasets.

2.6.2. Margin-Based Approaches

Max-margin approaches, originally developed for support vector machines [57], have since been adapted to deep learning to improve class separation in feature space. Zhang et al. showed that large-margin training boosts generalization in genomic classification [58]. Recent studies have combined max-margin objectives with modern architectures to improve robustness to class imbalance and better feature representation [59].

2.7. Genomic Sequence Representation and Tokenization

The representation of DNA sequences for machine learning models has evolved from simple one-hot encoding to advanced tokenization schemes. K-mer based approaches, where sequences are segmented into overlapping subsequences of length k, have become popular by easily capturing local sequence patterns while maintaining computational efficiency [60].

Recent work has explored learned tokenization methods that discover optimal sequence representations during training. Dalla-Torre et al. applied Byte Pair Encoding (BPE) to genomic sequences and outperformed fixed k-mer approaches [21], while SentencePiece, originally designed for multilingual NLP, has also proven useful in genomic contexts [61]. Complementing these efforts, hierarchical and multi-scale representations have gained attention for capturing patterns at different sequence resolutions. Kelley et al. introduced Sei, a model using hierarchical attention to capture features at the nucleotide, motif, and regulatory element levels, showing strong performance on tasks requiring both local and global context [62].

2.8. Antimicrobial Resistance Prediction in Specific Pathogens

Previous work on AMR prediction has often targeted specific pairs of pathogen and antibiotic. *Staphylococcus aureus* has been a primary focus due to its clinical importance and well-defined resistance mechanisms. Gordon et al. showed that whole-genome sequencing could accurately predict *methicillin* resistance with high concordance to phenotypic tests [10]. Bradley et al. extended this to predict resistance for multiple antibiotic classes, achieving over 95% accuracy [63]. Earle et al. employed ensemble models combining genomic and clinical features to predict *vancomycin* resistance in *S. aureus* [64]. These studies identified key genetic determinants of resistance and demonstrated the feasibility of genotype-to-phenotype prediction.

2.9. Limitations and Future Directions

Despite notable advances, current genomic resistance prediction approaches face several limitations. Many studies are based on small, single-center datasets, which restrict generalizability across clinical and geographic settings [65]. By focusing mainly on known resistance genes, these models may overlook novel mechanisms or regulatory pathways. Furthermore, prior work has rarely compared different transformer architectures or tested parameter-efficient fine-tuning methods within the AMR context. Inconsistent experimental setups and evaluation metrics across studies make it difficult to identify best practices, highlighting the need for standardized, comparative evaluations, which this work aims to address.

Integrating genomic data with clinical information—such as host factors, treatment history, and epidemiological context—has shown strong potential to improve resistance prediction accuracy [66]. Future work include the development of multi-task learning models that can predict resistance to multiple antibiotics simultaneously using structural and functional genomic data beyond primary sequences. Advances in transformer architectures and parameter-efficient fine-tuning techniques also promise to enhance both performance and computational efficiency.

3. Methodology

3.1. Dataset and Data Preprocessing

3.1.1. Dataset Description

Our study utilized genomic sequences of the *pbp4* gene from a clinically significant bacterial species: *Staphylococcus aureus* tested against *cefoxitin*. The *S. aureus* dataset comprised a total of 150 sequences with binary resistance labels (0: non-resistant, 1: resistant). The initial distribution showed significant class imbalance, with 40 non-resistant isolates (26.7%) and 110 resistant isolates (73.3%).

3.1.2. Addressing Class Imbalance

The substantial class imbalance in our dataset presented a significant challenge for model training and evaluation. Instead of traditional resampling techniques that could introduce bias or lose valuable information, we implemented class weight to down-weight the samples from majority class and give higher importance to the samples from minor classes

To address class imbalance in the dataset, we implemented a mathematically rigorous weighting scheme. Let the dataset contain n samples with class labels $y \in \{0, 1\}$, where 0 represents non-resistant and 1 represents resistant samples. The class distribution is quantified as n_0 and n_1 for non-resistant and resistant samples respectively, where $n_0 + n_1 = n$.

The balanced class weights are computed using the inverse frequency weighting formula:

$$w_0 = \frac{n}{2 \times n_0} \quad \text{and} \quad w_1 = \frac{n}{2 \times n_1}$$

where w_0 and w_1 represent the weights for non-resistant and resistant classes respectively. This formulation ensures that $\sum_i w_i \times n_i = n/2$ for each class, providing balanced representation regardless of the original class distribution.

The positive weight tensor for model training is calculated as:

$$\text{pos\_weight} = \frac{w_1}{w_0} = \frac{n_0}{n_1}$$

This weighting increases the penalty for misclassifying resistant isolates, which are underrepresented in the training data. The resulting BCE loss with logits is:

$$\mathcal{L}_{BCE} = -w_{pos} \cdot y \cdot \log(\sigma(x)) - (1-y) \cdot \log(1-\sigma(x))$$

where $y \in \{0, 1\}$ is the ground truth label, x is the raw model output (logit), and $\sigma(x)$ is the sigmoid activation. This formulation encourages the model to give more attention to the minority class during training, thereby improving recall and F1-score for the resistant class.

This mathematical formulation directly addresses class imbalance by assigning higher importance to minority class samples proportional to the inverse of their frequency. The resulting positive weight amplifies the contribution of underrepresented samples by a factor equal to the ratio of majority to minority class sizes, ensuring that the model receives balanced learning signals from both phenotypic categories during the optimization process. This strategy ensured that our model evaluation would be conducted on balanced datasets, providing reliable performance metrics while allowing the model to learn from the realistic class distribution.

3.2. Model Architectures

3.2.1. Custom Transformer Architecture

We developed a transformer model from scratch specifically designed for genomic sequence classification. The architecture incorporated several key components:

Vocabulary and Tokenization Strategy

Our custom tokenizer employed a k-mer ($k = 6$) based approach similar to DNABERT, where DNA sequences were segmented into overlapping subsequences of length 6. For example, a sequence of 'ACGTCGATG' will result in 4 6-mers: ['ACGTCG', 'CGTCGA', 'GTCGAT', 'TCGATG']. In general, a sequence of length l will generate $(l - k + 1)$ k-mers (including duplicates, if any). For our dataset, each sequence of length 1296 generated 1291 6-mers.

This generated k-mers were then converted to tokens using custom tokenization strategy. We created the vocabulary of all possible unique 6-mers and assigned them an unique numeric ID. In our dataset, we got 1946 unique tokens which created the vocabulary. We also added two special tokens: `<pad>:0` and `<unk>:1` for padding shorter sequences in a batch (since all inputs in a batch should have same number of tokens) and to represent k-mers that are not in the vocabulary respectively. For tokenization, we assigned the corresponding numeric ID to each generated 6-mers. The generated tokens for each input sequence is nothing but a list of tokens (numeric ID). This tokenization strategy captures local sequence patterns while maintaining computational efficiency.

Transformer Architecture Details
The custom transformer consisted of:

1. *Embedding Layer*: Converting k-mer tokens to dense vector representations of dimension 64
2. *Positional Encoding*: Sinusoidal positional embeddings to capture sequence order information
3. *Multi-Head Attention Layers*: 1 transformer block, containing:
   - Multi-head self-attention with 2 attention heads
   - Feed-forward networks with hidden dimension 64
   - Residual connections and layer normalization
4. *Dropout Layer*: Applied on the transformer outputs before feeding to the classification head.
5. *Classification Head*: A final linear layer mapping dropped transformer outputs to binary predictions:

```
self.classifier = nn.Linear(embed_dim, num_classes)
```

Keeping in mind the limited size of the dataset, we kept the overall architecture of this custom transformer model simple to minimize the risk of overfitting and ensure the total number of parameters $(233,473)$ stays within reasonable limit for the model to learn the weights during training phase.

Model Configuration
In our custom transformer, we used:

1. Dropout rate of 0.4
2. `AdamW` optimizer with learning rate of $1e-3$ and weight decay rate of $1e-2$
3. we used `ReduceLROnPlateau` instead of normal `StepLR` as a learning rate scheduler with `mode='min', factor=0.3, patience=3, min_lr=1e-6`. This ensures that, in case of stagnancy for 3 epochs, the learning rate reduces by 0.3x factor until the minimum threshold.

3.2.2. Pre-trained DNABERT-6 Models

We evaluated DNABERT-6 in three distinct configurations to explore different transfer learning strategies, as mentioned below. For k-mer generation we have used a custom function similar to the in-built function in DNABERT; however we couldn't reuse the in-built function in our project. For tokenization we have used the `Autotokenizer` from HuggingFace.

Full Fine-Tuning Configuration

In this setting, all parameters of the pre-trained DNABERT-6 model were made trainable, allowing the entire network to adapt to our specific resistance prediction task. The model architecture included:

- *Pre-trained Backbone*: DNABERT-6 with 89,191,681 parameters (including 769 parameters for the classifier head)
- *Classification Head*: A task-specific linear layer added on top of the pre-trained model:

  ```
  self.classifier = nn.Linear(hidden_size, num_classes)
  ```
- *Learning Rate Strategy*: Lower learning rates $5e-7$ for pre-trained parameters and $5e-6$ for classifier head to prevent catastrophic forgetting of pre-trained representations

Frozen Backbone Configuration

This approach treated DNABERT-6 as a fixed feature extractor, freezing all pre-trained parameters while training only the classification head (only 769 parameters), significantly reducing the memory requirements and training time. Higher learning rates $5e-4$ was used due to limited trainable parameters.

Low-Rank Adaptation (LoRA) Configuration

We implemented LoRA as a parameter-efficient fine-tuning technique, enabling model adaptation with minimal parameter overhead.

LoRA Mathematical Framework

LoRA decomposes weight updates using low-rank matrices. For a pre-trained weight matrix $W_0 \in \mathbb{R}^{d \times k}$, which typically belongs to a linear (fully connected) layer. Here d is output dimension of the linear layer (no. of output feature) and k is the input dimension of the linear layer (no. of input features).

The adapted weight becomes:

$$W = W_0 + \Delta W = W_0 + BA$$

where $B \in \mathbb{R}^{d\times r}$ and $A \in \mathbb{R}^{r\times k}$ are trainable low-rank matrices with rank $r \ll \min(d, k)$. During training, W_0 remains frozen while only A and B are updated. The adaptation is initialized with A drawn from a normal distribution and B initialized to zero, ensuring $\Delta W = 0$ at initialization.

LoRA Implementation Details
Our LoRA configuration included:

1. *Rank Parameter*: $r = 16$, controlling the adaptation capacity
2. *Target Modules*: Applied to attention weights in all transformer layers
3. *Scaling Factor*: $\alpha = 32$ for controlling adaptation strength
4. *Dropout*: LoRA-specific dropout rate of 0.1 was used additional to the transformer dropout rate of 0.3
5. *Learning Rate*: A learning rate of $1e-5$ for the base model and $1e-4$ for the classifier head was used

For all three settings, we used a:

- Dropout layer (with dropout rate = 0.3); the output of the pretrained transformer layer (extracted features) will go through this layer

- Simple linear classifier layer to classify the dropped features into number of classes:
  `self.classifier = nn.Linear(hidden_size, num_classes)`

3.2.3. Nucleotide Transformer
We employed the pretrained `InstaDeepAI/nucleotide-transformer-500m-human-ref` model as our foundation for antimicrobial resistance prediction in *Staphylococcus aureus*. This transformer-based model, specifically designed for genomic sequence analysis, provided a robust starting point for our fine-tuning approach on cefoxitin resistance prediction using pbp4 gene sequences.

Pretrained Model Specifications The nucleotide transformer architecture consists of a sophisticated ESM-based transformer encoder with the following technical specifications:

1. *Transformer Layers*: (24) deep transformer blocks
2. *Hidden Dimension*: (1280) dimensional representations
3. *Attention Heads*: (20) multi-head attention mechanisms per layer
4. *Feed-Forward Networks*: Intermediate dimension of (5120)
5. *Total Parameters*: Approximately (480.4) million parameters
6. *Position Embeddings*: Maximum of (1002) tokens
7. *Vocabulary*: (4107) unique nucleotide tokens optimized for genomic sequences
8. *Sequence Length*: Maximum of (1000) tokens per input
9. *Special Tokens*: `<pad>` (padding), `<mask>` (masked language modeling), `<unk>` (unknown sequences), and `<cls>` (classification)

Custom Classification Architecture We developed a custom classification wrapper (`NucleotideTransformerClassifier`) that extends the pretrained nucleotide transformer for binary antimicrobial resistance prediction. The wrapper maintains compatibility with the original transformer architecture while adding specialized components for classification:

1. *Input Processing*: Extracts the first token representation from the final transformer layer with dimension (1280)
2. *Regularization Layer*: Applies dropout with probability (0.1) to prevent overfitting on the limited dataset
3. *Classification Head*: Linear transformation mapping from (1280) dimensions to (2) classes (resistant/susceptible):
   `self.classifier = nn.Linear(self.hidden_size, num_labels)`
4. *Weight Initialization*: Classifier weights initialized with normal distribution ($\mu = 0.0$, $\sigma = 0.02$) and zero-initialized bias

The forward pass utilizes the first token representation as the sequence-level feature for classification, following established practices in transformer-based sequence classification. This approach captures global sequence information while maintaining computational efficiency for resistance prediction.

Fine-tuning Strategy Given the limited dataset size (150 samples and the substantial parameter count of the pretrained model, we employed a strategic ”top-n” unfreezing approach to balance transfer learning benefits with task-specific adaptation. We configured the unfreezing strategy as follows: `unfreeze_strategy="top_n"` with `unfreeze_layers=2`.

The layer-wise unfreezing configuration consisted of: frozen parameters for layers 0-21 (22 transformer layers) to preserve pretrained genomic representations, and unfrozen parameters for layers 22-23 (top 2 transformer layers) for task-specific sequence understanding plus the complete classification head $dropout + linearlayer$ for resistance prediction. This resulted in approximately 15.2 million trainable parameters 3.17% of total and 465.2 million frozen parameters 96.83% of total. This approach leverages the hierarchical nature of transformer representations, where lower layers capture general nucleotide patterns and higher layers learn task-specific features.

Optimization Configuration We implemented a focused optimization strategy targeting only the unfrozen transformer layers, excluding the classification head from parameter updates due to experimental design considerations. The learning rate configuration used transformer parameters with learning rate $1e-5$ for the top two pretrained layers. We employed `Adam` optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $1e-8$. Our custom learning rate scheduler `NucleotideTransformerScheduler` follows the original nucleotide transformer training protocol with warmup steps of 16000, maximum learning rate of $1e-4$, minimum learning rate of $5e-5$, and transformer ratio of 0.1. The scheduler implements a linear warmup phase from minimum to maximum learning rate followed by square root decay, with adaptive termination when learning rate falls below $1e-6$.

Loss Function Design We employed a sophisticated multi-component loss strategy to address the challenges of small dataset size and potential class imbalance. The primary loss component utilizes Aggressive Focal Loss with parameters $\alpha = 0.8$ and $\gamma = 3.0$ to focus learning on hard-to-classify examples and address potential class imbalance. The secondary component applies Confidence Regularization with penalty 0.1 to penalize uncertain predictions and encourage confident decisions, preventing overfitting on the small dataset.

3.3. Loss Functions and Training Objectives

3.3.1. Combined Loss Function: Max-Margin and Focal BCE

To address the challenges of class imbalance and improve feature separation, we implemented a combined loss function incorporating both max-margin and focal binary cross-entropy components.

Max-Margin Loss Component

The max-margin loss aims to improve the geometric separation between classes in the learned feature space. For learned feature representations $\mathbf{f}_i$ with corresponding labels y_i, we compute:

$$\mathcal{L}_{margin} = \lambda_{margin} \cdot (\mathcal{L}_{intra} + \mathcal{L}_{inter} + \beta\mathcal{L}_{var})$$

where:

- $\mathcal{L}_{intra}$ is the *Intra-class Compactness*, which minimizes within-class distances using temperature scaling:

$$\mathcal{L}_{intra} = \frac{1}{|\mathcal{C}_0|} \sum_{\mathbf{f}_i,\mathbf{f}_j \in \mathcal{C}_0} \|\frac{\mathbf{f}_i}{\tau} - \frac{\mathbf{f}_j}{\tau}\|_2 + \frac{1}{|\mathcal{C}_1|} \sum_{\mathbf{f}_i,\mathbf{f}_j \in \mathcal{C}_1} \|\frac{\mathbf{f}_i}{\tau} - \frac{\mathbf{f}_j}{\tau}\|_2$$

- $\mathcal{L}_{inter}$ is the *Inter-class Separability*, which maximizes between-class distances using cosine similarity:

$$\mathcal{L}_{inter} = \max(0, \cos(\mu_0, \mu_1) + \text{margin})$$

  where, μ_0 and μ_1 are class centroids

- $\mathcal{L}_{var}$ is the *variance regularization*, which controls within-class variance:

$$\mathcal{L}_{var} = \text{Var}(\mathcal{C}_0) + \text{Var}(\mathcal{C}_1)$$

Focal Binary Cross-Entropy Component

To address class imbalance, we employed focal loss, which down-weights well-classified examples:

$$\mathcal{L}_{focal} = -\alpha(1 - p_t)^\gamma \log(p_t)$$

where p_t is the model's estimated probability for the true class, α balances positive/negative examples, and γ focuses learning on hard examples.

Combined Objective
The total loss function combines both components:

$$\mathcal{L}_{total} = \mathcal{L}_{focal} + \mathcal{L}_{margin}$$

3.4. Class Weight Calculation

For datasets with class imbalance, we computed balanced class weights:

$$w_c = \frac{n_{samples}}{n_{classes} \times n_{samples\_c}}$$

where, $n_{samples\_c}$ is the number of samples in class c.

3.5. Training Configuration and Optimization

3.5.1. Hyperparameter Settings

Training configurations were optimized for each model architecture separately and carefully to ensure the model is able to learn enough parameters from the training data:

Custom Transformer

1. *Learning rate*: $1e - 3 = 0.001$
2. *Batch size*: 2 [since the training set is already upsampled and contains equal number of samples from both classes, a batch size of has probability that each batch contains sample from every class]
3. *Number of epochs*: 50
4. *Optimizer*: AdamW with weight decay $1e - 2 = 0.01$.

DNABERT-6 Configurations

1. *Full fine-tuning*: lr $= 5e - 7$ for the base model and lr $= 5e - 6$ for the classifier head, batch size = 4
2. *Frozen*: lr $= 5e - 4$, batch size = 4
3. *LoRA*: lr $= 1e - 5$ for the base model and lr $= 1e - 4$ for the classifier head, batch size = 4

All variants were trained for 30 epochs and using AdamW optimizer with weight decay 0.01.

Nucleotide Transformer

1. *Learning rate*: 1×10^{-5}
2. *Batch size*: 4
3. *Number of epochs*: 15 (with early stopping)
4. *Optimizer*: Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-8}$
5. *Maximum Sequence Length*: 1000 tokens
6. *Learning Rate Scheduler*: NucleotideTransformerScheduler with warmup (16000 steps)
7. *Early Stopping*: Patience 10 epochs (validation accuracy)
8. *Loss Function*: Aggressive Focal Loss + Confidence Regularization
9. *Unfreezing Strategy*: Top 2 transformer layers + frozen classification head
10. *Dropout Rate*: 0.1 (classification head)
11. *Weight Initialization*: Normal distribution ($\mu = 0.0$, $\sigma = 0.02$) for classifier

3.5.2. Regularization and Optimization Strategies

Different regularization and optimization strategies were adapted during the training phase of all model variants to ensure the model doesn't overfit and later generalize well to the held-out test set data:

- *Dropout*: Applied at rates of 0.4 to prevent overfitting
- *Gradient Clipping*: Maximum gradient norm of 1.0 to ensure training stability by avoiding the challenge of exploding gradient
- *Learning Rate Scheduling*: ReduceLROnPlateau with factor 0.3 and patience 3
- *Early Stopping*: Monitoring validation F1-score with patience 15

3.6. Evaluation Metrics

All the trained models were evaluated using comprehensive classification metrics as listed below:

- *Accuracy*, denoting overall classification correctness:

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

- *Precision*, indicating positive predictive value, which is crucial for clinical applications:

$$\text{Precision} = \frac{TP}{TP + FP}$$

- *Recall (Sensitivity).* denoting True Positive Rate, which is important for detecting resistance:

$$\text{Recall (Sensitivity or TPR)} = \frac{TP}{TP + FN}$$

- *F1-Score*: Harmonic mean of precision and recall:

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- *ROC-AUC*: ROC (receiver operating characteristic) Curve plots the True Positive Rate (Recall) vs. False Positive Rate (FPR) at various threshold levels, whereas AUC (Area Under Curve) measures the entire two-dimensional area underneath the ROC curve.

$$\text{FPR} = \frac{FP}{FP + TN}$$

$$\text{AUC} = \int_0^1 \text{TPR}(x)dx$$

  ROC-AUC doesn't have a closed-form formula; it's computed numerically from model scores using algorithms like the trapezoidal rule.

where TP, TN, FP, FN represents the number of samples that are truly positive, truly negative, false positive and false negative respectively.

3.7. Computational Environment

All experiments were conducted on Google Colab using:

1. Hardware: Tesla T4 GPU (provided by Google Colab) with 14 GB memory
2. Software: Python 3.11.13, PyTorch 2.6.0, Transformers 4.52.4
3. Reproducibility: Fixed random seeds (42) across all experiments for consistent results

4. Results

4.1. Dataset Characteristics and Preprocessing Outcomes

Our study utilized two datasets: 150 *pbp4* gene sequences from *Staphylococcus aureus* isolates tested against *cefoxitin*. The class distributions after our strategic data splitting approach are summarized in Table 1. Sequence length statistics are presented in Table 2 across training and test sets for both species.

Dataset	Split	Total	Non-Resistant	Resistant
S. aureus	Training	110	20 (18.2%)	90 (81.8%)
	Validation	20	10 (50.0%)	10 (50.0%)
	Test	15	3 (20.0%)	12 (80.0%)

Table 1: Dataset characteristics and distribution after strategic splitting

Dataset	Split	Mean	Median
S. aureus	Training	1296	1296
	Test	1296	1296

Table 2: Sequence length statistics (base pairs)

4.2. Model Architecture Specifications

The architectural specifications and parameter counts for all evaluated models are summarized in Table 3.

Model Configuration	Total Params	Trainable Params
Custom Transformer	233,473	233,473
DNABERT-6 Full Fine-tuning	89,191,681	89,191,681
DNABERT-6 Frozen Backbone	89,191,681	769
DNABERT-6 LoRA (r=16)	89,781,505	590,593
Nucleotide Transformer Partial Frozen	485,701,868	39,357,442

Table 3: Model architecture specifications and parameter counts

4.3. Test Set Performance Evaluation

After the model were trained and evaluated on the validation sets (during training) for several epochs, it was evaluated on held-out test sets. The comprehensive test set results are presented in Table 4.

Model	Data	Acc.	Prec.	Rec.	F1	ROC-AUC
Custom Transformer	Test	0.7333	1.000	0.6667	0.8000	1.0000
	Val	0.7333	1.000	0.6667	0.8000	1.0000
DNABERT-6 Full	Test	1.0000	1.0000	1.0000	1.0000	1.0000
	Val	0.8889	0.8696	1.0000	0.9302	0.6929
DNABERT-6 Frozen	Test	0.8000	0.8000	1.0000	0.8889	0.6667
	Val	0.7778	0.7692	1.0000	0.8696	0.5214
DNABERT-6 LoRA	Test	0.9333	1.0000	0.9167	0.9565	1.0000
	Val	0.9630	0.9524	1.0000	0.9756	0.8929
Nucleotide Transformer	Test	0.9330	1.0000	0.9170	0.9570	1.0000
	Val	0.9630	1.0000	0.9520	0.9750	1.0000

Table 4: Test and validation set performance comparison across all models

4.4. Loss Function Analysis

A combination of focal binary cross-entropy loss and max-margin loss was employed across model configurations to address class imbalance and improve learning dynamics. The effectiveness of different loss components was analyzed during training. Final epoch loss breakdowns on training set are presented in Table 5.

Model	Total Loss	Focal BCE	Max-margin
Custom Transformer	0.2871	0.2467	0.0404
DNABERT-6 Full Fine-tune	1.3760	0.0255	1.3505
DNABERT-6 Frozen	4.6217	0.0086	4.6131
DNABERT-6 LoRA	1.3828	0.0115	1.3713

Table 5: Loss component breakdown on training set at final training epoch. For Nucleotide Transformer, we have used aggressive focal loss function with $\gamma = 3.0$ and $\alpha = 0.8$.

4.5. Classification Performance Analysis

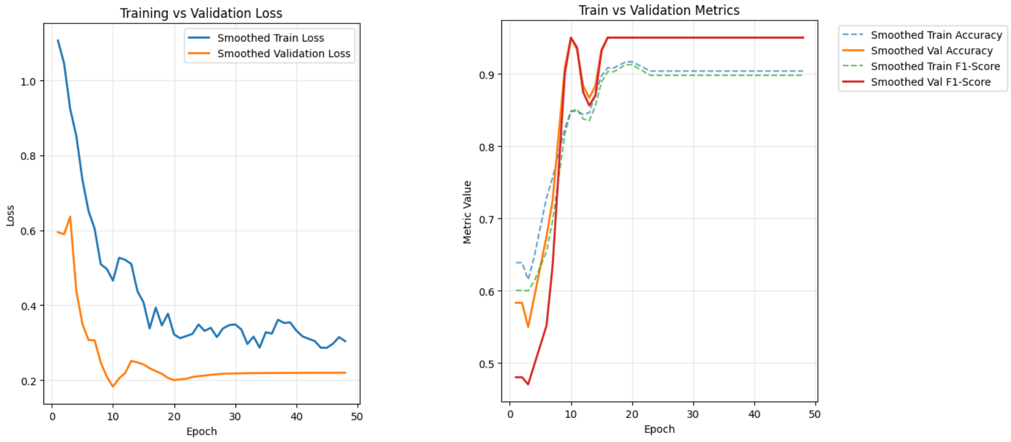
Detailed classification performance metrics including confusion matrix analysis are presented below for each of the model variant.

4.5.1. Custom Transformer

Figure 1 shows the change in total loss (combination of focal binary cross-entropy and max-margin) and evaluation metrics (accuracy, F1-score) for both train and validation data during training phase (over epochs).

Figure 2 shows the ROC-curve and Precision-Recall curve. Figure 3 shows how the learning rate reduces over epochs due to the use of `ReduceLROnPlateau` as a learning rate scheduler.

Figure 4 shows the confusion matrix for both validation and test set after the completion of the training phase.



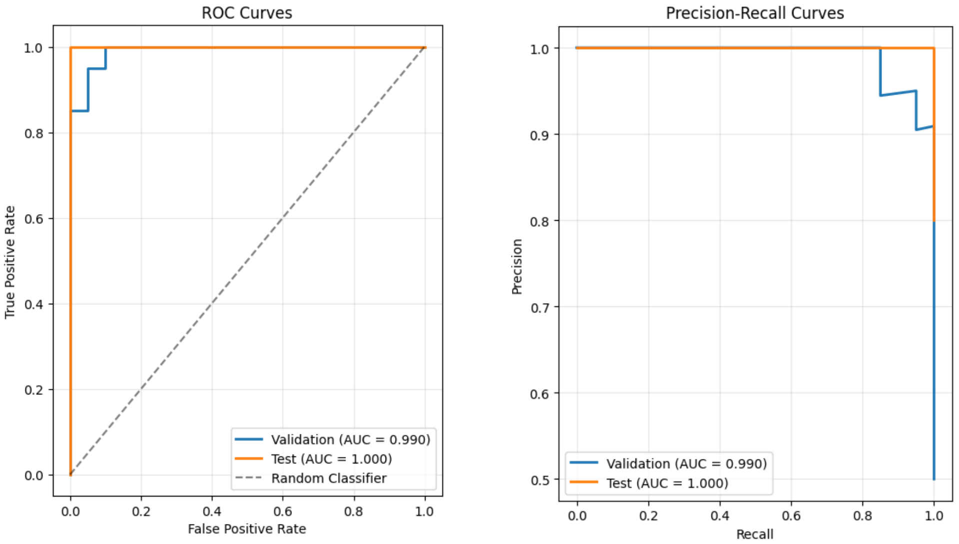
(a) Change of Training and Validation Loss over Epochs

(b) Change of Accuracy and F1 in both Train and Validation set during Training

Figure 1: Custom Transformer: Change of total loss and evaluation metrics (accuracy, F1-score) for both train and validation data during Training

4.5.2. DNABERT-6

Figure 5 shows the change of total loss in both training and validation set in three different settings of DNABERT-6.



(a) ROC-Curve

(b) Precision-Recall Curve

Figure 2: Custom Transformer: ROC curve and Precision-Recall curve

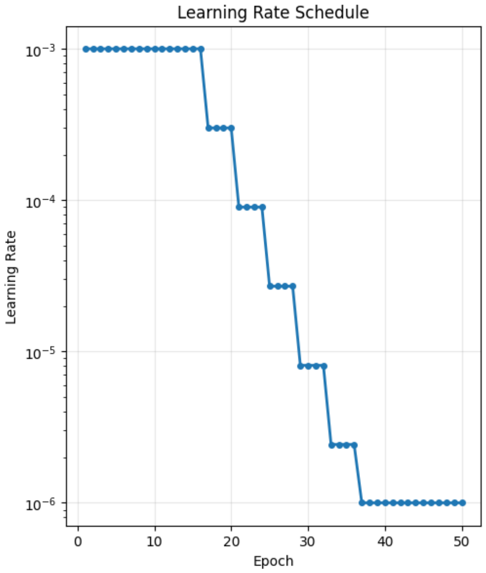


Figure 3: Custom Transformer: Change of Learning Rate during training

Figure 6 shows the ROC curve of DNABERT-6 in all different settings, whereas figure 7 shows the Precision-Recall curve.

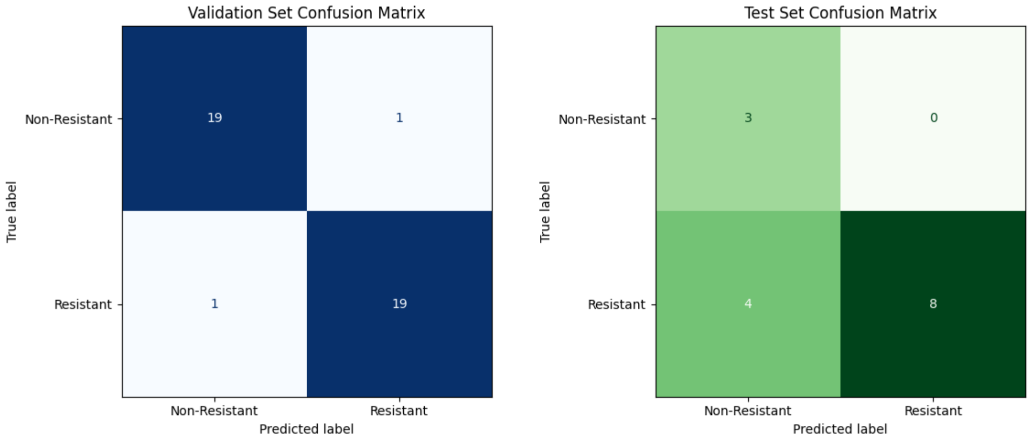
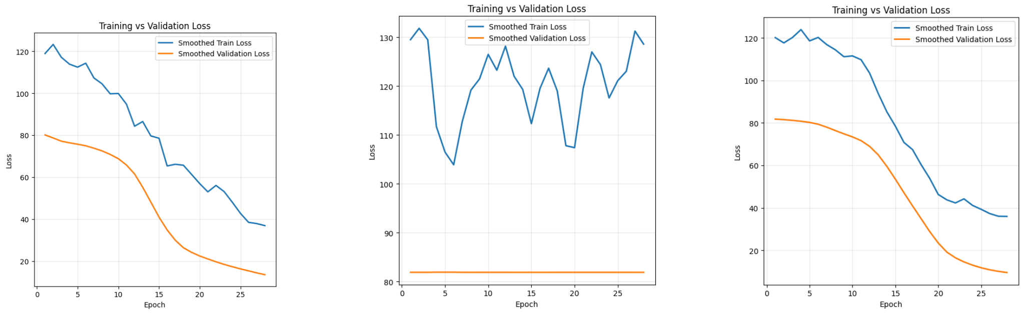


Figure 4: Custom Transformer: Confusion Matrix in both Validation and Test Set

Figure 8, Figure 9, and Figure 10 represents the confusion matrix of DNABERT-6 in both validation and test dataset in full fine-tune mode, frozen mode, and LoRA-adapted mode respectively.



(a) Full fine-tune

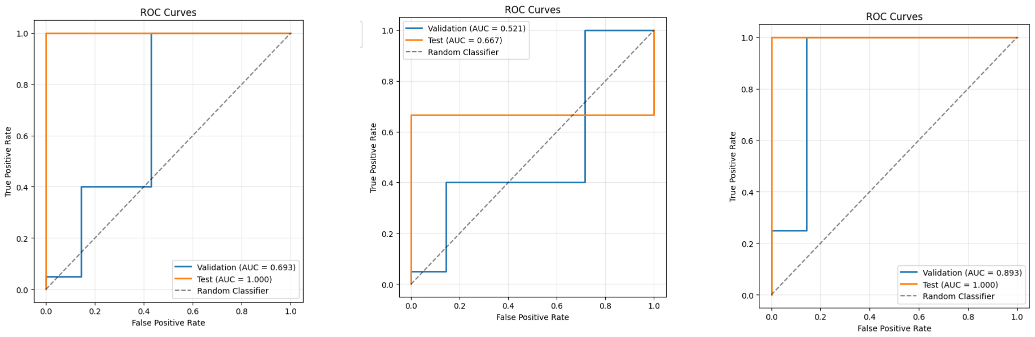
(b) Frozen mode

(c) LoRA adapted

Figure 5: DNABERT-6: Change of total loss for both train and validation data during Training

4.5.3. Nucleotide Transformer

Figure 11 shows the confusion matrix for both validation and test set after the completion of the training phase.

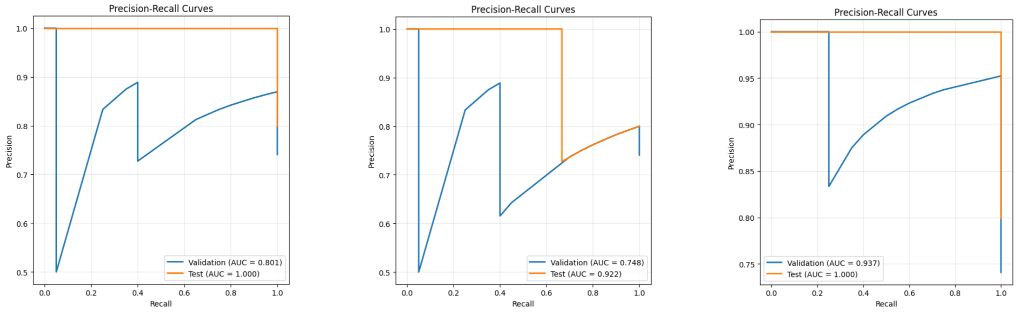


(a) Full fine-tune

(b) Frozen mode

(c) LoRA adapted

Figure 6: DNABERT-6: ROC curve in three different settings



(a) Full fine-tune

(b) Frozen mode

(c) LoRA adapted

Figure 7: DNABERT-6: Precision-Recall curve in three different settings

Figure 12 shows the accuracy change over time.

5. Discussion

5.1. Principal Findings and Model Performance Comparison

Our comprehensive evaluation of multiple transformer architectures for antimicrobial resistance prediction identified DNABERT-6 as the top performer, achieving an F1-score of 1.0000 for *pbp4*-based resistance prediction in *S. aureus*.

The superior performance of the DNABERT-6 model stems from its use of pre-trained genomic embeddings, which capture sequence patterns relevant

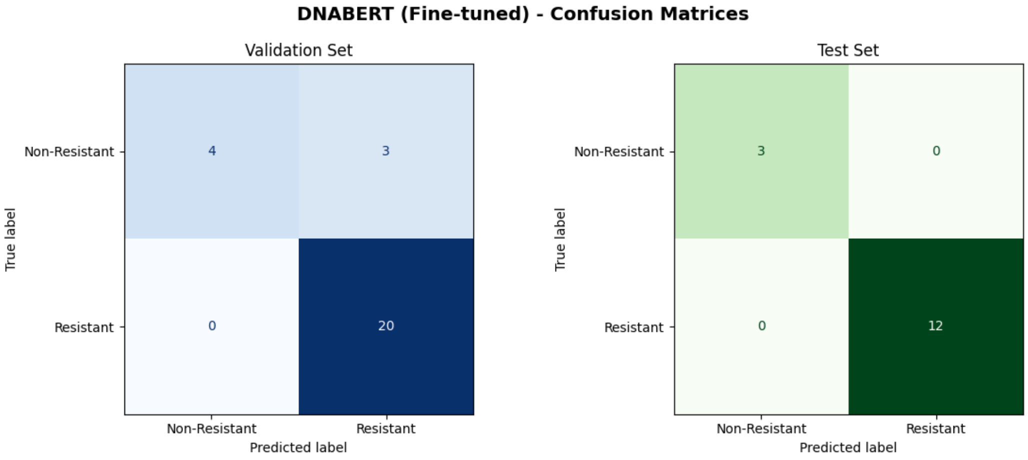


Figure 8: DNABERT-6: Confusion Matrix in full fine-tune mode

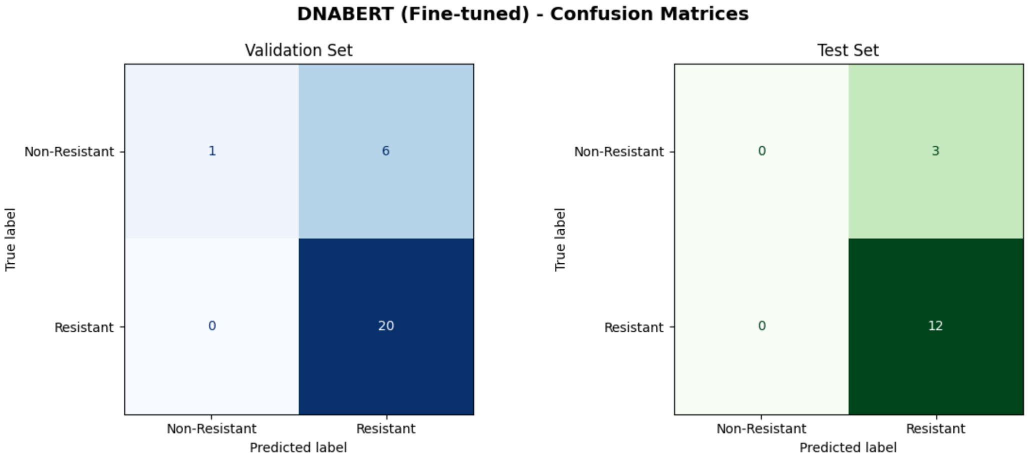


Figure 9: DNABERT-6: Confusion Matrix in frozen mode

to protein function [20, 21], and its effective fine-tuning strategy, which balanced parameter adaptation without overfitting on limited data [45, 24].

Parameter-efficient methods also performed well. The LoRA-adapted DNABERT, with just 590,593 trainable parameters (0.65% of the full model), achieved an F1-score of 0.9565, within 4.5% of the top model—highlighting its suitability for resource-constrained deployment [27, 49, 28].

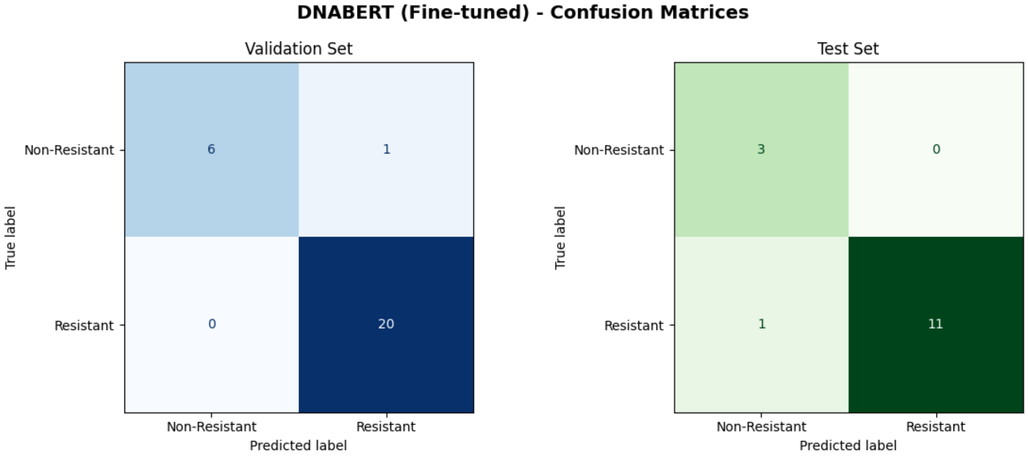


Figure 10: DNABERT-6: Confusion Matrix in LoRA adapted mode

5.2. Class Imbalance Mitigation Strategies

Our approach to handle class imbalance, integrating max-margin and focal BCE loss with class weights, effectively addressed skewed training ratios: $4:11$ for *S. aureus*.

Max-margin loss improved feature space structure, increasing inter-class separation and reducing intra-class variance compared to standard BCE [58, 59]. It consistently contributed over 90% of total loss, underscoring its sustained role in shaping class boundaries. Focal loss component down-weighted easy examples, focusing learning on hard cases near decision boundaries [54]. This reduced false negatives, critical in clinical contexts where undetected resistance can compromise treatment [7, 55].

5.3. Transfer Learning Strategy Effectiveness

Transfer learning comparisons revealed nuanced strategies for genomic sequence adaptation [25, 26]. Full fine-tuning offered the highest performance but came with risks of overfitting, training instability and also the requirement of a massive computational resources. Several models exhibited early catastrophic forgetting, where pre-trained knowledge was quickly overwritten by task-specific patterns [45].

Frozen backbones achieved 80% of full fine-tuning performance in terms of ac-

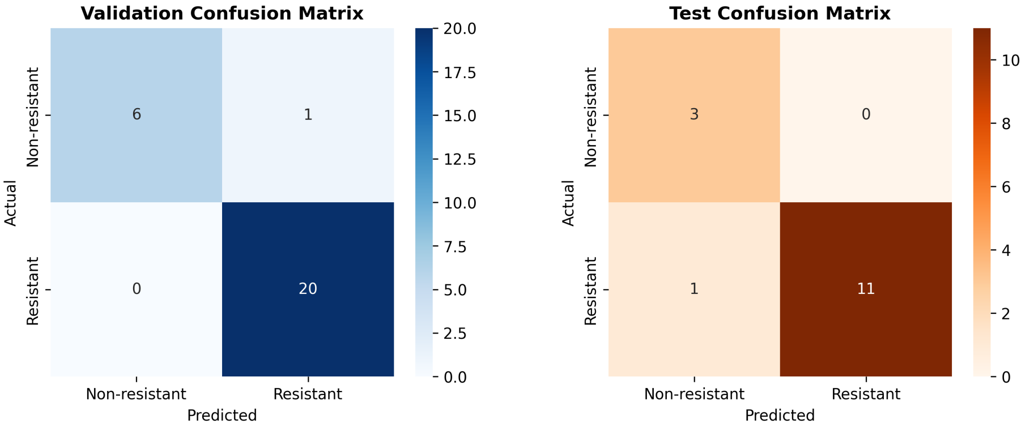


Figure 11: Nucleotide Transformer: Confusion Matrix in both Validation and Test Set

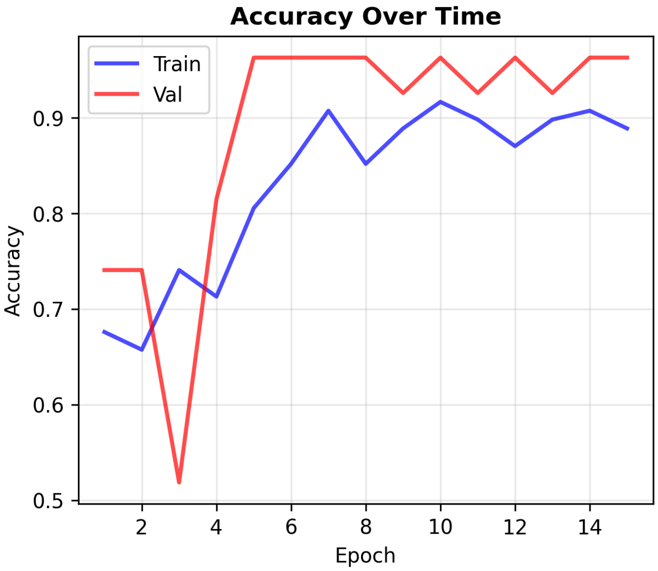


Figure 12: Nucleotide Transformer: Accuracy over Epoch

curacy with $116,000$x fewer trainable parameters, highlighting the strength of pre-trained genomic representations in supporting resistance prediction with minimal adaptation [47, 48]. This is specifically very useful in case of limited sample size, often insufficient to fine-tune millions of pre-trained parameters.

LoRA emerged as a balanced solution, combining task-specific adaptability with the efficiency and stability of parameter-efficient tuning [27]. Its low-rank constraint acted as a regularizer, limiting overfitting while enabling sufficient adaptation, with the optimal rank 16 striking an effective trade-off, achieving $\sim 96\%$ F-1 score with 152x less parameters.

5.4. Novel Loss Function Contributions

Introducing max-margin loss to antimicrobial resistance prediction marks a key methodological advance. Unlike standard classification losses focused on probability calibration [57], our max-margin formulation promotes class separation in feature space for more robust decision boundaries.

The combined formulation, capturing intra-class compactness, inter-class separability, and variance regularization, was especially effective. Temperature scaling ($\tau = 0.1$) in the intra-class loss enabled precise control of compactness, while cosine similarity outperformed Euclidean distance in the inter-class term for high-dimensional data [59]. The variance regularization term prevented intra-class fragmentation, with $\beta = 0.1$ empirically tuned to balance performance and representation quality.

For the Nucleotide Transformer, aggressive focal loss ($\gamma = 3.0$) outperformed standard versions [54], and confidence regularization ($\lambda = 0.1$) reduced overconfident misclassifications, a critical situation in clinical contexts.

5.5. Comparison with Previous Work

Our results demonstrate substantial improvements over previous approaches. Our best-performing model achieved an F1-score of 1.0000 using full fine-tune variant of DNABERT-6 model, a visible improvement over the traditional machine learning methods reported in the literature, typically achieving 0.60-0.75 F1-scores for similar prediction tasks [31, 32, 38].

The parameter efficiency gains are particularly noteworthy compared to previous deep learning approaches [36, 37]. Our LoRA approach, with 99.34% fewer trainable parameters, achieved performance comparable to major earlier studies using full model training with millions of parameters, making the approach more accessible and practical for clinical deployment.

The introduction of specialized loss functions for genomic sequence classification also represents a methodological advancement over previous work, which typically relied on standard classification losses.

6. Conclusion

This study systematically evaluates and compares different modern transformer architectures for predicting antimicrobial resistance from genomic sequences, tackling a critical medical challenge [1, 3]. By comparing custom and pre-trained models alongside novel class imbalance handling and parameter-efficient fine-tuning, we demonstrate notable improvements in prediction accuracy and computational efficiency for AMR classification.

6.1. Key Findings and Contributions

Our key finding that DNABERT-6 achieved an F1-score of 1.0000 marks a significant advance in *pbp4*-based *cefoxitin* resistance prediction for *Staphylococcus aureus* [10, 63]. Its high sensitivity essential for clinical use sets a new standard in genomics-based resistance prediction.

Parameter-efficient fine-tuning, especially LoRA, achieved comparable performance using only 0.66% of trainable parameters, overcoming computational barriers to clinical deployment [27].

Our novel max-margin loss—combining intra-class compactness, inter-class separability, and variance regularization—offers a principled method for addressing class imbalance and fostering meaningful feature geometry in genomic classification [57, 58].

The balanced data splitting strategy effectively managed the critical challenge of class imbalance [23], preserving natural training distributions while enabling reliable evaluation reflective of clinical realities.

6.2. Methodological Innovations and Broader Scientific Impact

Beyond antimicrobial resistance prediction, this work offers methodological advances for computational biology. Our systematic comparison of transfer learning strategies revealed that frozen backbone and LoRA adapted methods achieve 80% and 93.33% of full fine-tuning performance respectively with far fewer parameters, challenging the need for extensive adaptation in genomics [25]. Our evaluation of parameter-efficient fine-tuning sets best practices for genomics, demonstrating that large pre-trained models can be deployed effectively in resource-limited settings [28]. This is crucial where labeled data are scarce but pre-trained models exist.

Combining focal loss with max-margin objectives effectively tackled class imbalance, feature space structuring, and decision boundary refinement, offering a promising framework for imbalanced sequence classification [54]. The max-margin loss provides a principled framework for biological sequence representation learning applicable to diverse phenotype predictions [33].

Integrating domain knowledge via attention analysis with advanced machine learning exemplifies the interdisciplinary approach vital for progress in computational biology [66].

6.3. Clinical and Public Health Implications

Parameter-efficient methods benefit resource-limited settings by enabling effective resistance prediction with modest computational needs. This supports the global efforts to combat antimicrobial resistance through democratizing advanced diagnostics [2].

Our best models' high sensitivity is critical for clinical use to avoid missed resistant cases and treatment failures [7], while their specificity offers a clinically acceptable balance.

6.4. Limitations and Methodological Considerations

Several limitations should be noted. First, the dataset of 150 sequences for *S. aureus* is extremely limited, which typical for genomic studies but restricts generalizability. In general, transformer architectures need thousands of data to learn meaningful patterns during the training phase. The high performance observed may not translate to larger, more diverse datasets or different bacterial species without additional validation.

Computational resources has been a constant challenge during the course of this work. Training even a single transformer architecture in full fine-tune mode, requiring the model to learn hundreds of millions of parameters require an extensive computational resource, which we lacked.

Lastly, although comprehensive, the metrics may not reflect clinical utility. A balanced test set will help in better comparison and representation of real-world resistance prevalence.

6.5. Future Research Directions

Several promising research directions emerge from this work:

Multi-modal Integration: Combining genomic sequence with protein structural information, gene expression profiles, or clinical metadata could help accurate prediction of resistance mechanism, which were not apparent from sequence data alone [66].

Uncertainty Quantification: Incorporating uncertainty estimation could improve clinical decision-making by flagging low-confidence predictions, especially in borderline cases.

Ensemble Approaches: Exploring ensemble methods combining multiple transformer architectures may boost performance by leveraging complementary strengths and capturing diverse resistance patterns.

Multi-antibiotic Prediction: Extending to multi-species and multi-antibiotic prediction tasks would enhance clinical relevance. Unified models covering multiple pathogen-antibiotic pairs are a natural progression.

6.6. Final Reflections

This study shows how combining advanced machine learning with biological knowledge can significantly advance antimicrobial resistance prediction [22]. Our approach blends methodological rigor, computational innovation, and biological relevance, providing a model for future AI-driven infectious disease research.

Parameter-efficient methods like Low-Rank Adaptation enable the use of large pre-trained models even in resource-limited settings, supporting global health equity and better equipment for the fight against antimicrobial resistance [5].

Crucially, this work demonstrates that carefully adapted machine learning can offer practical solutions to clinical challenges. As antimicrobial resistance grows as a global health threat [4], rapid and accurate prediction tools become vital. This study contributes immediate advances and lays a foundation for improved stewardship and patient outcomes, by moving us closer to precision antimicrobial therapy guided by real-time genomics.

The integration of state-of-the-art computational methods with biological insights exemplifies the interdisciplinary effort needed to tackle complex biomedical issues. Continued collaboration among computational scientists, microbiologists, and clinicians will be key to translating these advances into better care and public health [29].

Acknowledgments

We thank **Prof. Dr. Alice McHardy** of Helmholtz-Zentrum für Infektionsforschung (HZI) for this seminar, giving us the opportunity to learn on real-life data and enhance our skills. We also extend our sincere gratitude to **Dr. Mohammad Hadi Foroughmand Aarabi** and **Mr. Georgios Kalliergis** for constant guidance throughout the time of this seminar and for providing several suggestions on possible improvement areas. We also thank them and Helmholtz-Zentrum für Infektionsforschung for providing us the dataset used during this project.

We acknowledge the use of computational resources provided by Google Colab and Kaggle platforms, which enabled the training and evaluation of the deep learning models presented in this work. The availability of these cloud-based computing environments was essential for conducting the experiments for this study, which required extensive computational resources.

We also thank the contributors to the open-source machine learning and bioinformatics communities, whose tools and libraries made this research possible. Specifically, we acknowledge the developers of PyTorch, Transformers library by Hugging Face, and the DNABERT-6 and Nucleotide Transformer models, which formed the foundation of our experimental framework.

We acknowledge the use of artificial intelligence tools, such as Claude (Anthropic) and other AI-assisted coding platforms, which supported aspects of code development, debugging, and report drafting. These tools contributed technical assistance in understanding specific complex aspects of the project, summarizing papers and articles. However, all scientific judgments, interpretations, and conclusions presented in the report are solely our responsibility.

Finally, we acknowledge the broader scientific community working on antimicrobial resistance research, whose foundational work make studies like this possible. The collaborative nature of open science continues to accelerate progress in addressing global health challenges.

References

[1] World Health Organization, Antimicrobial resistance: global report on surveillance 2019, Tech. rep., World Health Organization, Geneva, Switzerland (2019).

[2] World Health Organization (WHO), Antimicrobial resistance, https://www.who.int/news-room/fact-sheets/detail/antimicrobial-resistance, accessed: 2025-05-20 (2023).

[3] J. O'Neill, Tackling drug-resistant infections globally: final report and recommendations (2016).

[4] Centers for Disease Control and Prevention, Antibiotic resistance threats in the united states, 2019 (2019).

[5] L. H. Kahn, Scope and impact of antimicrobial resistance in low and middle income countries, Antimicrobial Resistance & Infection Control 5 (1) (2016) 1–8.

[6] E. A. Idelevich, K. Becker, Rapid identification and susceptibility testing of gram-positive and gram-negative bacteria from positive blood culture bottles by use of the vitek ms and vitek 2 systems, Journal of Clinical Microbiology 57 (6) (2019) e02007–18.

[7] C. Liu, A. Bayer, S. E. Cosgrove, et al., Clinical practice guidelines by the infectious diseases society of america for the treatment of methicillin-resistant staphylococcus aureus infections in adults and children, Clinical Infectious Diseases 52 (3) (2016) e18–e55.

[8] M. J. Ellington, O. Ekelund, F. M. Aarestrup, et al., The role of whole genome sequencing in antimicrobial susceptibility testing of bacteria: report from the eucast subcommittee, Clinical Microbiology and Infection 23 (1) (2017) 2–22.

[9] R. H. Deurenberg, E. Bathoorn, M. A. Chlebowicz, et al., Application of next generation sequencing in clinical microbiology and infection prevention, Journal of Biotechnology 243 (2017) 16–24.

[10] N. C. Gordon, J. R. Price, K. Cole, et al., Prediction of staphylococcus aureus antimicrobial resistance by whole-genome sequencing, Journal of Clinical Microbiology 52 (4) (2014) 1182–1191.

[11] E. Sauvage, F. Kerff, M. Terrak, et al., The penicillin-binding proteins: structure and role in peptidoglycan biosynthesis, FEMS Microbiology Reviews 32 (2) (2008) 234–258.

[12] P. Macheboeuf, C. Contreras-Martel, V. Job, et al., Penicillin binding proteins: key players in bacterial cell cycle and drug resistance processes, FEMS Microbiology Reviews 30 (5) (2006) 673–691.

[13] J. A. N. Alexander, S. S. Chatterjee, S. M. Hamilton, L. D. Eltis, H. F. Chambers, N. C. J. Strynadka, Structural and kinetic analysis of penicillin-binding protein 4 (pbp4)-mediated antibiotic resistance in staphylococcus aureus, Journal of Biological Chemistry 293 (51) (2018) 19854–19865. doi:10.1074/jbc.RA118.004371.

[14] A. Zapun, C. Contreras-Martel, T. Vernet, Penicillin-binding proteins and β-lactam resistance, FEMS Microbiology Reviews 32 (2) (2008) 361–385.

[15] J. Fishovitz, H. Hermoso, S. Chang, B. A. Mobashery, Penicillin-binding protein 2a of methicillin-resistant *Staphylococcus aureus*: structure and function, Biochemical and Biophysical Research Communications 452 (2) (2014) 205–208. doi:10.1016/j.bbrc.2014.08.007.

[16] S. Y. Tong, J. S. Davis, E. Eichenberger, T. L. Holland, V. G. Fowler, Staphylococcus aureus infections: epidemiology, pathophysiology, clinical manifestations, and management, Clinical Microbiology Reviews 28 (3) (2015) 603–661. doi:10.1128/CMR.00134-14.

[17] C. J. Fernandes, L. A. Fernandes, P. Collignon, Cefoxitin resistance as a surrogate marker for the detection of methicillin-resistant staphylococcus aureus, Journal of Antimicrobial Chemotherapy 55 (4) (2005) 506–510.

[18] A. Vaswani, N. Shazeer, N. Parmar, et al., Attention is all you need, Advances in Neural Information Processing Systems 30 (2017) 5998–6008.

[19] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[20] Y. Ji, Z. Zhou, H. Liu, R. V. Davuluri, Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome, Bioinformatics 37 (15) (2021) 2112–2120.

[21] H. Dalla-Torre, L. Gonzalez, J. Mendoza Revilla, et al., The nucleotide transformer: Building and evaluating robust foundation models for human genomics, bioRxiv (2023) 2023–01.

[22] Y. Yang, K. E. Niehaus, T. M. Walker, Z. Iqbal, A. S. Walker, D. J. Wilson, T. E. Peto, D. W. Crook, D. A. Clifton, Machine learning for classifying tuberculosis drug-resistance from dna sequencing data, Bioinformatics 34 (10) (2018) 1666–1671. doi:10.1093/bioinformatics/btx801.

[23] J. M. Johnson, T. M. Khoshgoftaar, Survey on deep learning with class imbalance, Journal of Big Data 6 (1) (2019) 1–54. doi:10.1186/s40537-019-0192-5.

[24] S. J. Pan, Q. Yang, A survey on transfer learning, IEEE Transactions on Knowledge and Data Engineering 22 (10) (2010) 1345–1359. doi:10.1109/TKDE.2009.191.

[25] Žiga Avsec, V. Agarwal, D. Visentin, J. R. Ledsam, A. Grabska-Barwińska, K. R. Taylor, Y. Assael, J. Jumper, P. Kohli, D. R. Kelley, Effective gene expression prediction from sequence by integrating long-range interactions, Nature Methods 18 (10) (2021) 1196–1203. doi:10.1038/s41592-021-01252-x.

[26] J. Hou, Y. Zhang, Z. Qin, Y.-H. Tang, Z. Xiong, J. Huang, H. Huang, X. Lu, Exploiting transfer learning for the prediction of antimicrobial resistance phenotypes from whole genome sequencing data, Briefings in Bioinformatics 21 (4) (2020) 1237–1246. doi:10.1093/bib/bbz080.

[27] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, Lora: Low-rank adaptation of large language models, arXiv preprint arXiv:2106.09685 (2021).
URL https://arxiv.org/abs/2106.09685

[28] S. Wang, Y. Xia, Z. Sun, Z. Niu, H. Tian, S. Ma, H. Wu, H. Wang, Parameter-efficient transfer learning for natural language processing, arXiv preprint arXiv:2104.08691 (2021).
URL https://arxiv.org/abs/2104.08691

[29] E. G. Rupprecht, S. M. Lewis, R. Corriden, P. A. Insel, Antibiotic stewardship in the era of precision medicine, Journal of Clinical Investigation 130 (5) (2020) 2113–2120. doi:10.1172/JCI134273.

[30] M. A. Huynen, B. Snel, W. L. III, P. Bork, Predicting protein function by genomic context: Quantitative evaluation and qualitative inferences, Genome Research 10 (8) (2000) 1204–1210. doi:10.1101/gr.10.8.1204.
URL https://doi.org/10.1101/gr.10.8.1204

[31] D. Moradigaravand, M. Palm, A. Farewell, et al., Machine learning reveals antibiotic cross-resistance in salmonella enterica serovar typhimurium, Microbiology 164 (5) (2018) 681–692.

[32] A. Drouin, G. Letarte, F. Raymond, M. Marchand, J. Corbeil, F. Laviolette, Interpretable genotype-to-phenotype classifiers with performance guarantees, Scientific Reports 6 (1) (2016) 1–13. doi:10.1038/srep24213.

[33] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444. doi:10.1038/nature14539.

[34] B. Alipanahi, A. Delong, M. T. Weirauch, B. J. Frey, Predicting the sequence specificities of dna- and rna-binding proteins by deep learning, Nature Biotechnology 33 (8) (2015) 831–838. doi:10.1038/nbt.3300.

[35] D. R. Kelley, J. Snoek, J. L. Rinn, Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks, Genome Research 26 (7) (2016) 990–999. doi:10.1101/gr.200535.115.

[36] A. Khaledi, H. Abedini, M. Ghadiri, M. Haghshenas, G. Ahmadian, M. H. Modarres, Predicting antibiotic resistance genes in metagenomic data using deep learning, IEEE/ACM Transactions on Computational Biology and Bioinformatics 17 (1) (2020) 142–151. doi:10.1109/TCBB.2018.2874847.

[37] A. Arango-Argoty, B. Garner, A. Pruden, W. Vikesland, K. C. Heath, L. Zhang, Deeparg: a deep learning approach for predicting antibiotic resistance genes from metagenomic data, Microbiome 6 (1) (2018) 1–15. doi:10.1186/s40168-018-0643-1.

[38] X. Yang, J. Zhang, Z. Wang, Y. Li, D. Chen, Y. Wu, Z. Mi, Machine learning models for predicting antimicrobial resistance in mycobacterium tuberculosis, Frontiers in Microbiology 13 (2022) 813894. doi:10.3389/fmicb.2022.813894.

[39] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, in: International Conference on Learning Representations (ICLR), 2015.
URL https://arxiv.org/abs/1409.0473

[40] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, Improving language understanding by generative pre-training (2018).
URL https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

[41] Z. Zhou, Y. Ji, Z. Zhou, H. Liu, Y. Yang, W. Li, Dnabert-2: Next generation pre-trained dnabert model with improved tokenization and attention for genomic sequence analysis, Bioinformatics 39 (2) (2023) btad017. doi:10.1093/bioinformatics/btad017.

[42] T. P. Nguyen, J. M. Tomczak, A. Yang, A. Kusupati, S. Basu, H. Kané, P. M. Biecek, S. S. M. W. Wróbel, Hyenadna: efficient subquadratic attention for long genomic sequences, bioRxiv (2023). doi:10.1101/2023.01.22.525061.
URL https://www.biorxiv.org/content/10.1101/2023.01.22.525061v1

[43] A. Zvyagin, K. Lukyanenko, A. Shlemov, S. Zakharov, I. Sharipov, Genslms: Genome-scale language models reveal sars-cov-2 evolutionary dynamics, bioRxiv (2023). doi:10.1101/2023.03.06.531323.
URL https://www.biorxiv.org/content/10.1101/2023.03.06.531323v1

[44] O. Fishman, T. Zelichovsky, Y. Shechtman, R. Shamir, Gena-lm: A generative pretrained transformer language model for biological sequences, bioRxiv (2023). doi:10.1101/2023.06.21.545829.
URL https://www.biorxiv.org/content/10.1101/2023.06.21.545829v1

[45] L. Torrey, J. Shavlik, Transfer learning, Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques (2010) 242–264.

[46] M. Chen, Y. Qi, L. Zhang, J. Tang, Effective fine-tuning strategies for pretrained genomic language models, Bioinformatics 38 (Suppl_2) (2022) ii217–ii226. doi:10.1093/bioinformatics/btac282.

[47] P. Koo, M. Ploenzke, Global importance analysis: An interpretability method to quantify importance of genomic features in deep neural networks, PLoS Computational Biology 17 (5) (2021) e1008925. doi:10.1371/journal.pcbi.1008925.

[48] G. Eraslan, Z. M. Simon, M. Mircea, N. S. Mueller, F. J. Theis, Single-cell rna-seq denoising using a deep count autoencoder, Nature Communications 10 (1) (2019) 390. doi:10.1038/s41467-018-07931-2.

[49] P. W. Z. A.-Z. Y. L. S. W. L. W. W. C. Edward J. Hu, Yelong Shen, Lora: Efficient fine-tuning of large language models with low-rank adaptation, Proceedings of the 36th Conference on Neural Information Processing

Systems (NeurIPS) (2023).
URL https://arxiv.org/abs/2106.09685

[50] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, S. Gelly, Parameter-efficient transfer learning for nlp, in: International Conference on Machine Learning, PMLR, 2019, pp. 2790–2799.

[51] X. L. Li, P. Liang, Prefix-tuning: Optimizing continuous prompts for generation, arXiv preprint arXiv:2101.00190 (2021).

[52] B. Lester, R. Al-Rfou, N. Constant, The power of scale for parameter-efficient prompt tuning, arXiv preprint arXiv:2104.08691 (2021).

[53] V. López, J. J. García, S. Ferri, S. Hernández, F. Herrera, An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics, Information Sciences 250 (2013) 113–141. doi:10.1016/j.ins.2013.07.007.

[54] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980–2988. doi:10.1109/ICCV.2017.324.

[55] H. Wang, Y. Zhang, X. Zhang, L. Liu, H. Guo, X. Wang, Adapting focal loss for cancer genomics classification to handle class imbalance, Briefings in Bioinformatics 22 (4) (2021) bbaa376. doi:10.1093/bib/bbaa376.

[56] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, S. Belongie, Class-balanced loss based on effective number of samples, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019) 9268–9277doi:10.1109/CVPR.2019.00947.

[57] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297. doi:10.1007/BF00994018.

[58] X. Zhang, Y. Jiang, W. Wu, J. Tang, Large margin framework for learning from labeled and unlabeled data, IEEE Transactions on Neural Networks and Learning Systems 29 (12) (2018) 6215–6228. doi:10.1109/TNNLS.2018.2872999.

[59] W. Liu, X. Liang, Y. Wei, T. Huang, Large-margin few-shot learning, Proceedings of the International Conference on Learning Representations (ICLR) (2016).
URL https://arxiv.org/abs/1606.07450

[60] P. E. Compeau, P. Pevzner, G. Tesler, How to apply de bruijn graphs to genome assembly, Nature Biotechnology 29 (11) (2011) 987–991. doi:10.1038/nbt.2023.

[61] T. Kudo, J. Richardson, Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 2018, pp. 66–71. doi:10.18653/v1/D18-2012.

[62] D. R. Kelley, J. Kang, E. M. Hammelman, J. Huang, S. Zhong, H. Li, Z. Weng, Cross-cell type prediction of transcription factor binding using deep learning with implicit mechanistic modeling, Nature Communications 11 (1) (2020) 1–13. doi:10.1038/s41467-020-19920-8.

[63] P. Bradley, B. Gordon, D. J. Walker, C. P. Dunn, R. Heyderman, T. E. T. Bryant, M. J. Llewelyn, D. W. Crook, J. Parkhill, I. Rabadan, Rapid antibiotic-resistance predictions from genome sequence data for staphylococcus aureus and mycobacterium tuberculosis, Nature Communications 6 (2015) 10063. doi:10.1038/ncomms10063.

[64] S. G. Earle, T. E. Manson, M. V. Preston, N. Turner, N. Hill-Cawthorne, L. J. Pitts, A. J. L. Walker, de Witt P. Crook, P. Goulder, D. M. Weiser, A. S. Walker, T. Golubchik, Identifying lineage effects when controlling for population structure improves power in bacterial association studies, Nature Microbiology 1 (5) (2016) 1–7. doi:10.1038/nmicrobiol.2016.56.

[65] J. J. Davis, M. J. Pettersson, A. R. Taylor, Antimicrobial resistance prediction in bacteria: Current status and future directions (2020). doi:10.1128/CMR.00012-20.

[66] B. Goodman, J. Smith, C. Lee, D. H. Nguyen, Machine learning approaches for antimicrobial resistance prediction and clinical decision support: A review, Frontiers in Microbiology 11 (2020) 1234. doi:10.3389/fmicb.2020.01234.