Scorpio : Enhancing Embeddings to Improve Downstream Analysis of DNA sequences

Mohammad S. Refahi[1], Bahrad A. Sokhansanj[1], Joshua C. Mell[2], James R. Brown[1], Hyunwoo Yoo[1], Gavin Hearne[1], and Gail L. Rosen[1,*]

[1]Drexel University, Electrical and Computer Engineering, Philadelphia, 19104, USA
[2]Drexel University, College of Medicine, Philadelphia, 19104, USA
[*]corresponding author(s): Gail Rosen (glr26@drexel.edu)

ABSTRACT

Analyzing genomic and genetic sequences on the DNA level can be challenging due to the limited alphabet and sequence similarity varies depending on the labeling task, which makes tasks dependent on different evolutionary rates. In addition, metagenomic data poses significant challenges due to the vast diversity of taxa and genes within a microbiome. Here, we present Scorpio, a novel framework that employs triplet networks with contrastive learning, utilizing both pre-trained language models and k-mer frequency embeddings, to effectively a) discern taxonomic and gene information in metagenomic data and can be fine-tuned to b) identify drug resistance, etc. from AMR genes and c) identify promoters. Our approach demonstrates robust performance across a variety of tasks. It has notable performance in generalizing to novel taxonomic and gene classification (e.g. identifying known gene labels of sequences from novel taxa).The versatility of our triplet network framework for multitask classification highlights its potential for advancing health and environmental diagnostics. This method enhances our ability to process and interpret complex microbiome metagenomic data, offering significant implications for biomarker identification and the monitoring of disease and environmental health.

Introduction

Next-generation sequencing technologies have revolutionized the field of genomics by providing unprecedented access to vast pools of data from diverse environments. Metagenomic data offers the potential to gain insight into the composition and function of microbial communities ("microbiome"). Specifically, by sequencing the genomes of a full microbial community, rather than individual species in isolation, it becomes possible to quantify microbial diversity and understand the interaction between different species and strains, which is essential for understanding their evolution together, how community dynamics change in response to environment changes, and thereby better understand the community's function in human health and environmental ecosystems. Analyzing metagenomic data presents significant challenges. Unlike 16S rRNA amplicon sequencing, which primarily identifies species presence, shotgun metagenomics provides a more detailed view by capturing sequence-level changes[1]. These changes can represent evolutionary adaptations to coexistence with other species and environmental stimuli, and potentially impact phenotypic traits. Handling high-throughput reads, managing the complexity of diverse microbial populations, and resolving gene or regulatory region differences at the strain level are crucial for understanding these functional consequences[2]. Gene-centric analysis, which usually focuses on gene elements assembled from short-reads (or gene regions annotated from long-read technologies) are beneficial and can offer functional insights compared to short-read only analysis.

Traditional sequence alignment methods, which align unknown sequences to reference genomes, become computationally prohibitive (with respect to time and memory) with the increasing volume of metagenomic data[3–5]. This motivates the development of alignment-free methods that can more efficiently characterize sequences found in metagenomic data without requiring direct alignment methods. There are a plethora of alignment-free methods that rely on k-mer features, such as[2,6,7]. Recently, for example, there have been proposals to extract all k-mer (i.e., all genomic subsequences of length k) composition and abundance features to create sequence representations[8,9]. However, k-mer frequency and representation methods often lose crucial positional information—the context and order of nucleotides within a sequence—that is vital for accurate gene identification and functional analysis.

To address these limitations, representation learning techniques from natural language processing (NLP) have been adapted for genomic data. By treating nucleotides and amino acids as words in a sentence, models such as Bidirectional Encoder Representations from Transformers (BERT)[10], Embeddings from Language Models (ELMo)[11], and Generative Pre-trained Transformer (GPT)[12] can generate lower-dimensional sequence representations through language modeling tasks. These models effectively capture both sequence and evolutionary features but typically require fine-tuning for specific tasks to achieve optimal performance[13–16,16,17]. In recent years, contrastive learning has emerged as a robust technique for refining these

representations[18].This approach involves creating an embedding space where similar sequences are brought closer together, while dissimilar sequences are pushed apart. Contrastive learning, especially when implemented using triplet networks, enhances the ability to compare sequences rapidly and accurately without relying on traditional alignment methods. A triplet network consists of three parallel neural networks that process three inputs: an anchor, a positive example (similar to the anchor), and a negative example (dissimilar to the anchor). This structure allows the network to learn to distinguish between similar and dissimilar sequences effectively. It has been successfully applied to various biological tasks, including enzyme prediction, disorder region identification, and protein structural classification[8,19–21], by leveraging sequence similarity metrics to optimize the embedding space.

To tackle the challenges of metagenomic data analysis, we introduce Scorpio, an adaptable framework leveraging triplet networks for a range of genomics tasks. Scorpio is designed to be versatile, allowing it to be fine-tuned for multiple applications such as taxonomy classification, gene identification, antimicrobial resistance (AMR) detection, and gene promoter detection. By encoding DNA sequences with 6-mer frequency and BigBird[22] embeddings (a transformer model optimized for long sequences), Scorpio maintains the necessary hierarchical levels for robust generalization and accurate classification. Scorpio employs a hybrid query-distance/class-probability confidence scoring method to enhance prediction reliability. For efficient embedding retrieval, the framework utilizes FAISS (Facebook AI Similarity Search)[23], which supports applications like generating detailed gene reports and AMR prediction. This capability to handle both known and novel sequences without requiring extensive curation makes Scorpio a potentially powerful tool for metagenomics research. We validate Scorpio's performance across various tasks, showing its flexibility and generalization power, particularly in classifying out-of-domain datasets. Our approach demonstrates that triplet networks can effectively extract meaningful biological data representations. By integrating advanced NLP techniques and contrastive learning, Scorpio offers a comprehensive solution for the complex challenges of metagenomic analysis, providing significant insights into microbial community functions and their implications for human health and the environment.

Results

Overview of Scorpio

The development of Scorpio begins with creating a Gene-Taxa dataset from the NCBI taxonomy database, illustrated in Figure 1 (a). The Woltka pipeline[24] is used to download genomes, ensuring only one representative genome per genus. This unique dataset presents a challenge for state-of-the-art taxonomy and gene prediction models due to low sequence representation across taxa[25]. To focus solely on prokaryotic genomes, the dataset was further refined to include only nucleotide sequences from the superkingdoms Bacteria and Archaea. We applied additional filtering to enhance the dataset's reliability, focusing on gene names with high representation (more than 1000 instances) and removing unknown and hypothetical proteins. By such filtering, our training set, gene categories reduced down to 497 genes from 1929 different genera. Through this filtration method, we selectively include housekeeping marker genes and the most common pathways that are critical for maintaining the basic functions of microbial ecosystems. This targeted approach enables the model to focus on these essential components, facilitating the discovery of fundamental biological insights and enhancing our understanding of microbial ecosystems. The detailed distribution of instances per class at each level is shown in Figure 1 (a). One of the key advantages of our model is the dataset preparation process and the architectural ability to train on both gene and taxonomy hierarchies. This dual function/taxa focus enables the model to learn multimodal information together, categorized across different hierarchical levels such as phylum, class, order, family, and genus. This preparation of the dataset forms the foundation for training our model effectively, ensuring it captures the complex relationships and variations present in metagenomic data. Our training set is carefully balanced at both the gene level, which is the highest hierarchical level, and the genus level, which is the lowest. This balance ensures that we have enough samples for effective triplet training and the accurate selection of positive and negative examples. Balancing the dataset at these key levels is essential because it provides sufficient representation of each class within a level, which is crucial for hierarchical sampling. The other taxonomic levels have varying numbers of instances per class, adding necessary diversity to the dataset and enhancing the model's ability to generalize across different taxa.

Our framework employs a triplet training approach, where DNA sequences are transformed into embeddings using an encoder mechanism (Figure 1(b)). We have three distinct encoder mechanisms for triplet training: one based on 6-mer frequency (Scorpio-6Freq), and two others based on the embedding layer of BigBird. In one of these BigBird-based mechanisms, we have a fine-tunable embedding layer (Scorpio-BigDynamic), while in the other, all BigBird layers are frozen (Scorpio-BigEmbed). All combinations of positive, anchor, and negative samples are fed into the network to train the triplet network, which processes them through multiple linear layers to fine-tune the embeddings based on hierarchical information. One of the unique aspects of the Scorpio framework is its flexibility; users can adjust the number of hierarchical levels, select the type of level, and change the order of levels to train the model on. This adaptability is crucial because genes often do not fit neatly into natural hierarchical categories like taxonomy levels. By incorporating this flexibility, Scorpio shows promising results in enhancing the model's ability to generalize and perform multiple tasks, integrating phylogenetic and functional information effectively.

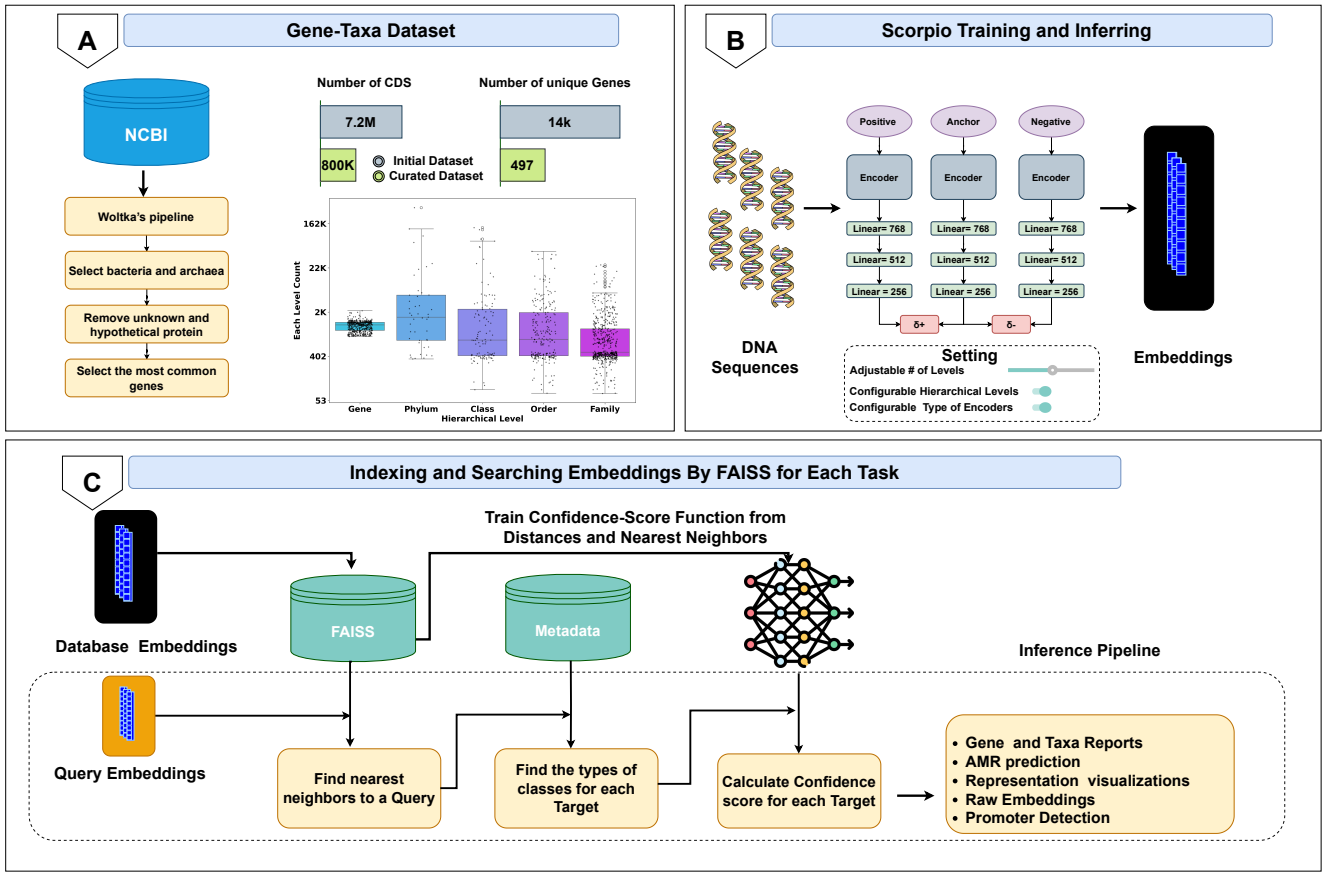


Figure 1. Overview of the Scorpio Framework. (A) Gene-Taxa Dataset Creation: Data from NCBI is filtered and refined to include 497 genes from 1929 genera. This involves removing unknown and hypothetical proteins and focusing on the most common genes, specifically housekeeping genes. The distribution of samples per level is shown in a bar plot, indicating a balanced dataset at the gene level. (B) Training and Inferring with Scorpio: DNA sequences are encoded using 6-mer frequency and BigBird embeddings. This adaptable configuration supports different Scorpio models, such as Scorpio-6Freq, Scorpio-BigDynamic, and Scorpio-BigEmbed, with adjustable hierarchical levels for enhanced generalization to adapt to different datasets and hierarchies. During inference, we use one of the Triplet branches to obtain the embedding vector, which is the final layer of the Triplet network. (C) Indexing and Searching: FAISS is utilized for efficient embedding retrieval and finding the nearest neighbor for each query. Based on the nearest neighbor from the validation set, we train a confidence score model for each level. During inference, this model calculates the confidence for each query. Depending on the application, we report classification results and confidence scores, supporting various applications such as gene and taxonomy reports, antimicrobial resistance prediction, and promoter detection.

Additionally, we evaluate the model's adaptability to single hierarchical levels, such as promoter detection. Our evaluations, detailed in the following sections, demonstrate its effectiveness in training for different tasks and its robust adaptability to various genomics and metagenomics-based applications.

Indexing and searching embeddings efficiently is a critical component of the Scorpio framework (Figure 1(c)). We use FAISS (Facebook AI Similarity Search) to store and retrieve precomputed embeddings quickly. When a query is made, the framework finds the nearest neighbors to the query embedding, and a confidence score is calculated. This score is generated by training a model on distance thresholds to predict normalized metrics like the F1 score. This process normalizes the confidence values between 0 and 1 and forces the model to understand relative distance thresholds derived from FAISS, ensuring robust and reliable confidence score. The comprehensive inference pipeline supports diverse applications, including generating gene and taxa reports, predicting antimicrobial resistance, visualizing representations, and providing raw embeddings for further analysis.

Scorpio embeddings can uncover both the gene's type and taxonomy levels from full-length gene sequences

Gene-centric metagenomics is a technique used to infer microbiome contents by assembling or extracting gene sequences. We evaluated the performance of Scorpio embeddings for gene and taxonomy classification of full-length DNA gene sequences against leading methodologies, Kraken2[5] (kmer-based technique widely used for taxonomy), MMseqs2[26] (a fast and efficient alignment search), and DeepMicrobes[9] (a comparable deep learning technique) (see Methods). For MMseqs2 and all methods generating embeddings, we employ the best hit results for classification (see Methods).

In the Test set (Table 1a), we included sequences such that each gene or genus represented was present in the training set, but the specific combinations of genes and genera were not repetitive. MMseqs2 maintained its leading position, which was anticipated as alignment-based techniques typically excel with sequences similar to their training database. Our Scorpio models surpassed Kraken2 and DeepMicrobes, demonstrating superior performance. Kraken2's performance was notably affected by the dataset's inclusion of only one preparation per gene in each genus, posing a significant classification challenge.

We then focused on the generalization capabilities of the methods, using the Gene Out and Taxa Out datasets to test performance on unseen gene types and taxa. In generalizing to unknown genes (Table 1b), defined here as novel genes absent from similar genera in the training set, our Scorpio embeddings demonstrated superior performance compared to Kraken2, MMseqs2, and DeepMicrobes. This analysis highlighted Scorpio's ability to capture nuanced patterns within gene sequences, surpassing traditional alignment-based methods that struggle with novel gene classes due to lower sequence similarity with the training set. Even basic embeddings, such as the 6-mer frequency representation, showed high accuracy across lower taxonomy levels, suggesting the robustness of motif frequency in taxonomy prediction beyond exact motif matches. Scorpio models consistently outperformed others across higher levels of taxonomy, achieving significantly better accuracy performance: 17 times higher than MMseqs2, 11 times higher than Kraken2, and approximately 3 times higher than DeepMicrobes at the phylum level. Notably, the Scorpio-6Freq model did not outperform the raw 6-mer frequency model at lower levels, likely due to the hierarchical training nature of Scorpio embeddings. When sequence repetition is minimal, hierarchical training can disproportionately impact performance at lower taxonomy levels, particularly in highly imbalanced datasets. The potential of Bigbird embeddings was also explored, showing significant promise in capturing complex patterns relevant to gene and taxonomy classification, even without fine-tuning, indicating advantages over traditional methods like Kraken2 and MMseqs2. In the Taxa Out dataset, which includes similar genes but from different phyla than those in the training set, MMseqs2 exhibited superior performance in gene classification. Our Scorpio-6Freq model achieved higher accuracy (88.4%) compared to MMseqs2 (87.3%), although MMseqs2 outperformed in terms of the F1-macro score by 3%. This underscores the resilience and effectiveness of alignment-based models in gene identification, while also highlighting the high accuracy of our Scorpio models in out-of-domain datasets. A significant advantage of Scorpio models over DeepMicrobes-based models is the ability to achieve both taxonomy and gene classification within a single training task. Our approach optimizes the loss for all gene and taxonomy levels simultaneously, reducing training time and achieving better performance without the need for separate models for family and gene classification, as required by regular supervised based models.

(a)

Level Model	Accuracy (%) phylum	class	order	family	gene	F1-macro (%) phylum	class	order	family	gene
6mer Freq.	57.6	45.8	35.9	27.5	29.3	30.4	21.1	19.6	15.6	34.5
BigBird	60.4	46.6	31.8	21.5	48.7	26.4	15.2	12.6	8.7	46.7
DeepMicrobes_family	24.2	8.8	3.7	0.5	0.2	2.7	0.7	0.2	0.1	0.1
DeepMicrobes_gene	25.9	10.5	3.2	1.0	94.1	5.1	1.8	0.7	0.3	93.8
MMseqs2	**93.3**	**89.8**	**79.7**	**61.5**	**97.4**	**79.4**	**54.8**	**47.0**	**31.2**	**98.2**
Kraken2	64.8	58.0	30.9	1.09	N/A	36.4	23.5	19.3	13.7	N/A
Scorpio-6Freq	85.8	75.3	49.8	29.1	95.1	49.6	27.0	18.5	9.9	94.9
Scorpio-BigEmbed	79.4	68.6	51.3	35.6	81.3	48.7	30.0	23.8	15.2	80.3
Scorpio-BigDynamic	80.2	69.4	51.1	34.8	89.7	49.7	30.5	23.8	14.6	89.2

(b)

	Taxonomy Generalization								**Gene Label Generalization**	
	Accuracy (%)				F1-macro (%)				Accuracy (%)	F1-macro (%)
Model	phylum	class	order	family	phylum	class	order	family	gene	gene
6mer Frequency	49.2	31.0	17.0	**10.8**	20.9	13.5	**10.6**	**8.4**	2.3	2.4
BigBird	44.3	23.9	9.0	4.1	11.2	5.4	3.3	1.9	23.5	22.0
DeepMicrobes_family	25.4	9.2	4.1	0.5	2.4	0.6	0.1	0.0	0.2	0.1
DeepMicrobes_gene	19.4	7.5	1.9	0.6	3.6	1.1	0.4	0.2	87.8	89.0
MMseqs2	4.3	2.7	1.1	0.5	2.2	1.4	0.9	0.5	87.3	**90.8**
Kraken2	1.1	0.6	0.26	0.17	0.5	0.7	0.4	0.3	N/A	N/A
Scorpio-6Freq	**73.8**	**56.3**	**21.9**	9.5	**29.3**	**13.7**	6.7	2.9	**88.4**	87.4
Scorpio-BigEmbed	50.8	29.4	10.4	4.6	14.2	6.7	4.0	2.4	58.0	55.6
Scorpio-BigDynamic	58.6	41.1	16.4	7.8	20.9	11.8	6.7	4.0	75.2	73.3

Table 1. Full gene length results: (a) Memorization Test: Identifying additional examples of training-data-known taxonomy and genes (Test Set); (b) Generalization Test: Taxonomy Generalization (Gene Out Set) and Gene Label Generalization (Taxa Out Set) Tests

Transferability of Scorpio Embeddings to Other Domains: Antibiotic Resistance Prediction

Next, we evaluated Scorpio models originally trained on a **Gene-Taxa dataset** (using Fig. 1 A and B) to generate embeddings for antimicrobial resistance prediction tasks. We expected that gene and taxonomy information could also help determine antimicrobial resistance classes and gene families, and consequently, the drug class of particular genes. It is essential to know that antimicrobial resistance sequences are in a similar range of sequence length, which we want to test the transferability of our model to other AMR prediction tasks.

For evaluation, we used a combination of the MEGARes[27] and CARD[28] datasets, with details provided in the Data section. MEGARes and CARD are global antimicrobial resistance (AMR) databases that integrate relevant data on bacterial taxonomy, genomics, resistance mechanisms, and drug susceptibility. To ensure a fair comparison, we created a custom database by integrating sequences from both MEGARes and CARD, standardizing the data to maintain consistency. This approach allowed us to evaluate our model against state-of-the-art models using a common reference. We evaluated models based on accuracy and F1-score, choosing benchmark approaches that allowed for database customization. Thus, we evaluated our models with MMseqs2[26], Abricate[29] (a well-known tool for mass screening of contigs for antimicrobial resistance genes, virulence factors, and other important genetic markers in microbial genomes), and BLASTn[30].

For embedding-based methods like BigBird, and 6-mer Frequency and our models, we used FAISS and the best hit to determine the class. We present the results in Figure 2a, showing the accuracy and F1-macro scores for each model across the three tasks: Gene Family (classification of genes into families), Resistance Mechanism (identification of mechanisms of resistance), and Drug Class (categorization of drugs based on class). We observed that Scorpio resulted in greater class prediction accuracy than baseline models for all three tasks. Specifically, the Scorpio-BigDynamic model, which is Scorpio on BigBird embeddings with a tunable embedding layer while the rest are frozen, achieved the highest accuracy and F1-macro scores across all tasks, with accuracy values of of 92.3%, 97.8%, and 95.9% and F1-macro scores of 73.4%, 96.9%, and 93.5% for Gene Family, Resistance Mechanism, and Drug Class prediction, respectively. In comparison, the MMseqs2 model, while showed lower accuracy values of 91.9%, 95.3%, and 95.0% and F1-macro scores of 75.2%, 83.9%, and 86.5%, respectively. The BLASTn model also performed well but was outperformed by our model. Abricate showed significantly lower performance, highlighting the limitations of this algorithm in detecting all AMR genes, as all methods were indexed on the same datasets.

Fine tuning of Scorpio Embeddings for Bacterial Promoter Prediction

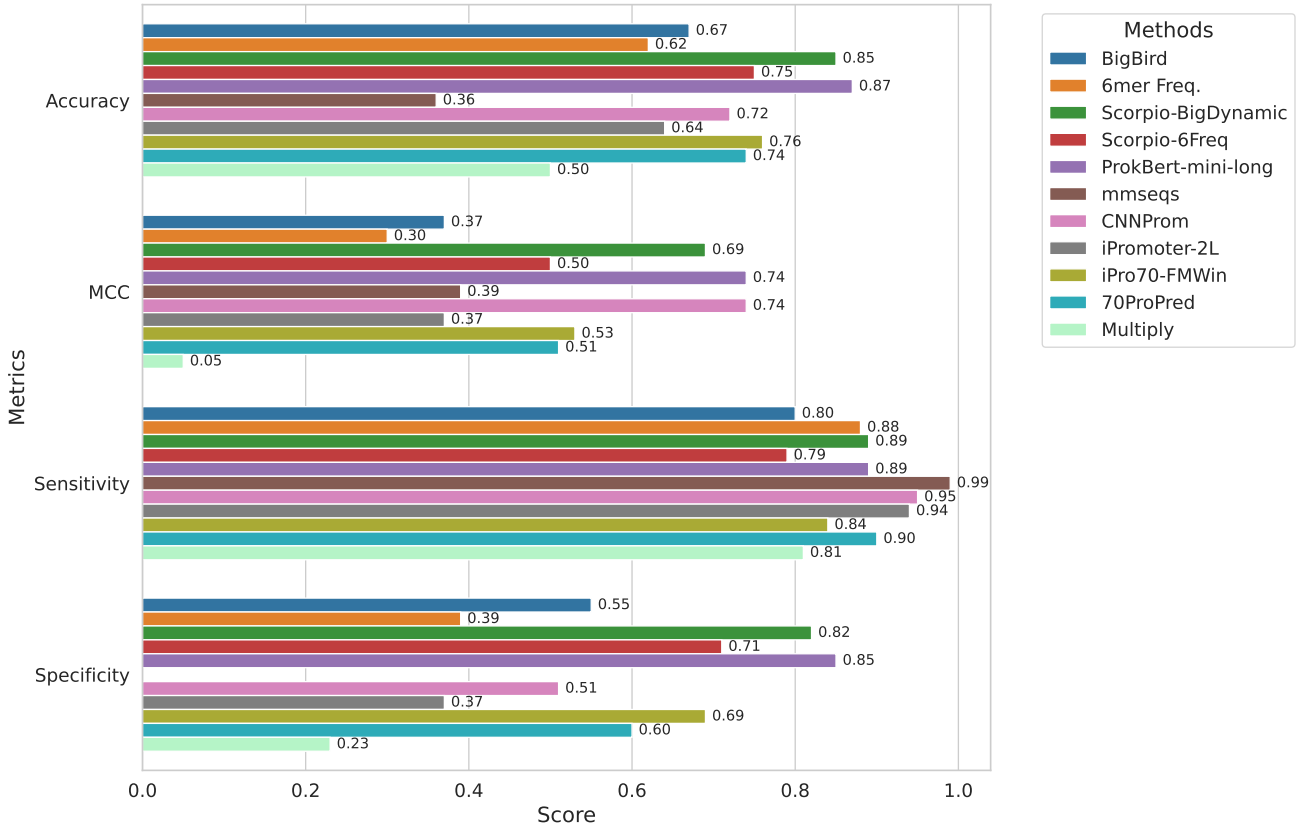
Next, we evaluated Scorpio's performance in predicting promoter regions involved in gene expression regulation. The pre-trained model, BigBird, was originally trained on gene sequences of fewer than 4,096 base pairs, making it challenging to detect very short sequences like promoters. Notably, since BigBird was trained exclusively on gene-encoding regions, it may not have encountered promoter regions before fine-tuning on promoter sequences. We aimed to investigate the impact of Scorpio-BigDynamic on fine-tuning for promoter sequences, considering the significant shift in hierarchical information from gene-taxonomy to promoter detection. For promoter prediction, the hierarchical representation in Scorpio simplifies to a single level, distinguishing between promoter and non-promoter sequences.

We initiated the evaluation by collecting promoter and non-promoter sequences from the ProkBERT dataset[31]. Prokaryotic promoter sequences typically span 81 base pairs. Our model's performance was independently evaluated using an *Escherichia coli* Sigma70 dataset, providing an objective assessment of its capabilities. This dataset, obtained from the study by Cassiano et al.[32], comprises *E. coli* sigma70 sequences. Positive samples, sourced from Regulon DB[33], are experimentally validated and widely recognized promoter sites.

In Figure 2b, we compare the performance results of different methods on promoter detection against our Scorpios method. Firstly, we observe that both Scorpio-BigDynamic and Scorpio-6Freq. show improvements in accuracy and Matthews correlation coefficient (MCC) metrics by more than 18% compared to raw pre-trained model. This illustrates that further fine-tuning of BigBird could also enhance performance for on promoter detection. Also, in comparison with state-of-the-art methods on promoters detection, our model performs better on average than other methods except Prokbert, which achieves 2% higher accuracy than our model. We have similar sensitivity but slightly lower specificity about 3% compared to Prokbert. This difference may arise because our BigBird model is solely trained on genes, unlike the Prokbert model which is trained on fragments from the whole genomes. Language modeling (capture high-level initial features, and since our model was never exposed to promoter regions during its initial training, it may confuse promoters with non-promoters. Upon further investigation, we found that some sequences misclassified by our model as non-promoters are actually promoters, such as the sequence, *"CGGTTGCCAACAACGTTCATAACTTTGTTGAGCACCGATACGCATTGTTGGAATTATCGCTCCTGGGCCAGGACCAAGATG"*, which also appears in the coding sequence (CDS) region of *NlpI*. This also suggests that, due to the compact nature of bacterial genomes, embedded promoters may be part of coding sequences. Consequently, a model trained solely on genes might misclassify these sequences because it wasn't exposed to the broader genomic context during training.

Level Model	Accuracy (%) GeneFamily	 Resist-Mech	 DrugClass	F1-macro (%) GeneFamily	 Resist-Mech	 DrugClass
Abricate	55.7	55.7	55.7	4.3	13.9	20.0
Mmseqs2	91.9	95.3	95.0	**75.2**	83.9	86.5
BLASTn	87.4	90.2	90.0	71.1	80.9	82.2
BigBird	40.0	58.7	35.5	4.2	17.0	10.0
6mer Freq.	89.6	95.3	92.3	63.5	86.8	89.0
Scorpio-6Freq (gene-taxa)	91.4	96.8	94.5	66.7	86.8	90.0
Scorpio-BigDynamic (gene-taxa)	**92.3**	**97.8**	**95.9**	73.4	**96.9**	**93.5**

(a) Antimicrobial Resistance Prediction



(b) Promoter Prediction

Figure 2. (a) Comparison of Antimicrobial Resistance (AMR) prediction performance metrics across different models. (b) Performance comparison of different promoter detection methods against our Scorpio approach. Both Scorpio-BigDynamic and Scorpio-6Freq improve accuracy and MCC by over 18% compared to their baselines. Our model outperforms all other methods except ProkBERT, which has 2% higher accuracy. We have similar sensitivity but 3% lower specificity compared to ProkBERT. We believe this is because BigBird was originally trained on genes rather than whole genomes, making it more robust to gene sequences. ProkBERT, on the other hand, was trained on whole genomes and fine-tuned for promoter detection.

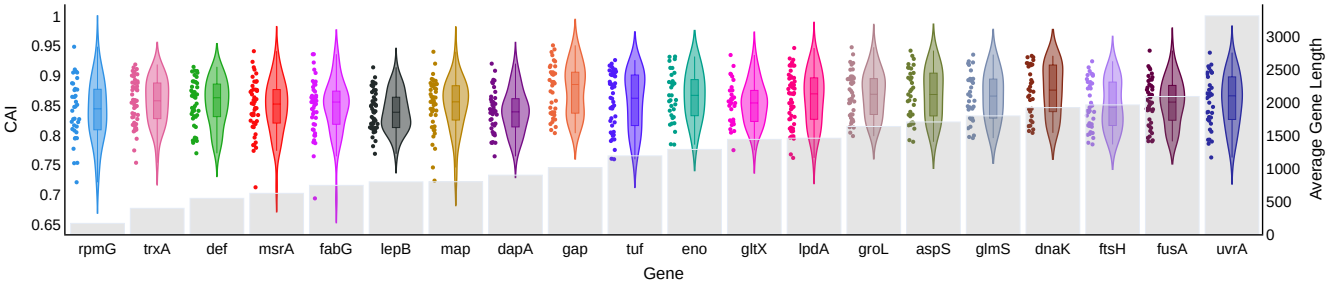
Unveiling the Complex Relationship Between Codon Adaptation and Sequence Similarity through Scorpio Embedding Analysis

In molecular biology, the Codon Adaptation Index (CAI) is an important metric that reflects the frequency of specific codons within a gene and indicates how well-adapted a gene is to its host organism's translational machinery. High CAI values generally correlate with efficient translation, suggesting a higher potential for gene expression and protein synthesis[34]. Recent research[35] has revealed a complex relationship between gene sequences, codon adaptation, and resulting protein structure . Codon usage bias, the preference for certain synonymous codons, has been shown to regulate co-translational protein folding, translation efficiency, and accuracy, as well as mRNA stability and transcription[36]. Nucleotide-based models could transform our understanding of gene expression and protein synthesis[37]. These models capture subtle variations in nucleotide sequences, revealing hidden relationships and providing deeper insights into how genetic code affects protein folding, stability, and function. To test our hypothesis about the potential relationship between gene embeddings and the Codon Adaptation Index (CAI), we followed a structured approach. First, we selected the 20 most common genes from our training set. Then, we identified all genera that include these genes, resulting in 31 genera in our dataset. For each genus, we calculated the CAI of each gene relative to its corresponding genome, using that genome as the reference. This means we have 31 different reference genomes, and we calculated the CAI for each gene within each genome.

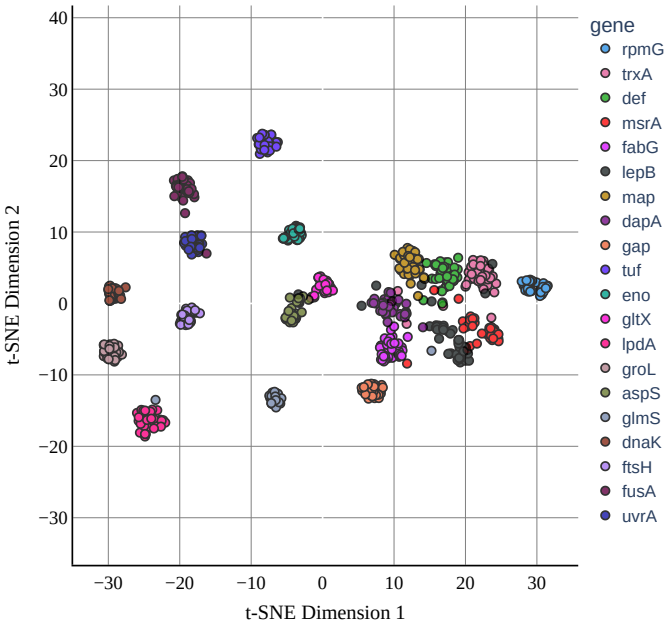
As shown in Figure 3a, we present the distribution of the Codon Adaptation Index (CAI) for each gene, along with the average gene length depicted in bar plots. The figure illustrates that the CAI index is independent of gene length, highlighting its critical role in gene expression. This independence is particularly important, as our embeddings might otherwise be influenced by gene length. Next, to obtain a global representation of the sequences, we employed t-SNE. Although t-SNE transformations of embeddings (Figure 3b) can be nonlinear and may depend on parameters like perplexity and the number of iterations, we used a perplexity value of approximately 50 to capture global structural information in our embedding space[38,39]. A higher perplexity value helps obtain more global rather than local information about each cluster and their distribution in the space. In Figure 3c, we compared the average CAI against the average of the first t-SNE dimension of our embeddings. A negative correlation emerged, with Pearson and Spearman's rank correlation coefficients of -0.60 ($p = 5.11e^{-3}$) and -0.67 ($p = 1.25e^{-3}$), respectively. This indicates a significant correlation between the overall representation of gene embeddings in space and the CAI.

Although we observed this correlation, it should be emphasized that embeddings do not linearly represent sequences. Understanding the causal influences for the observed gene distributions in the embedding space is highly complex and involves multiple factors. For example, in Figure 3b, we noticed that AspS and GltX genes, both encoding aminoacyl-tRNA synthetases which are key enzymes in the translation of the genetic code, are clustered closely together in embedding space despite not having a close CAI. Additionally, we observed similar clustering when considering a substantial portion of the aminoacyl-tRNA synthetases in our datasets, along with some others potentially involved in the synthesis pathway. Other proteins, including some ribosomal proteins, also tend to cluster together or appear near each other in the embedding space (Supplementary Figure 5). This observation suggests a potential correlation between gene expression, function, taxonomy information, and other biological factors within embeddings.

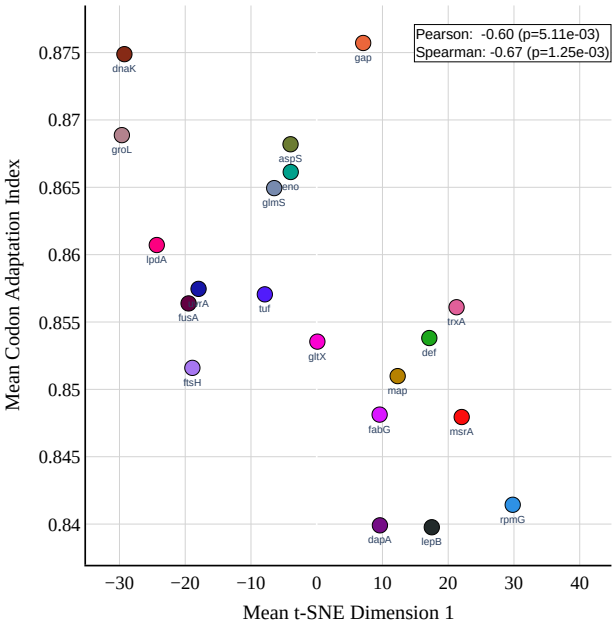
Also, Our analysis indicated a correlation between the embedding distance of genes and their sequence similarities. We used edit distance to measure gene distance and Euclidean distance to measure embedding distance, as shown in (Supplementary Figure 6). Our examination of sequence similarity within each gene shows that genes in embedding space are distributed based on their sequence similarity. On average, the R^2 value is about 0.41, indicating a moderate correlation between the edit distance of gene pairs and the Euclidean distance of their embeddings.



(a) Distribution of CAI values for each gene, sorted based on gene length from left to right. rpmG is the shortest gene, while urvA is the longest.



(b) t-SNE visualization of gene embeddings (Scorpio-BigDynamic Embeddings).



(c) Correlation between the first dimension of t-SNE and the average CAI per gene.

Figure 3. Exploratory analysis of gene embeddings and Codon Adaptation Index (CAI). (a) The violin plot shows the distribution of CAI values across genes, indicating variations in codon usage bias. The shaded bars demonstrate that CAI is not dependent on gene length. (b) The t-SNE visualization illustrates gene embeddings in a lower-dimensional space, revealing patterns of similarity and clustering. A high perplexity value was used to capture the global structure of the data, showing how genes relate to each other in space. (c) The correlation analysis between the first dimension of t-SNE embeddings and CAI values provides insights into the relationship between gene spatial organization and CAI. This analysis suggests a significant correlation between gene expression levels and CAI, with Pearson and Spearman's rank correlation coefficients of -0.60 ($p = 5.11 \times 10^{-3}$) and -0.67 ($p = 1.25 \times 10^{-3}$), respectively.

Assessing Confidence Scores: A Comparative Analysis of Gene and Taxonomy Classification Methods

For benchmarking against existing algorithms, we introduce a novel confidence scoring method for our Scorpio embedding, aiding biologists in making informed decisions and ensuring accuracy in scientific research[40]. Most methods like Kraken2 and MMseqs2 have a cutoff threshold to report their results. Based on a value of confidence score and E-value, they return some sequences and also report their scores. However, it's important to note that if a method reports a confidence score, there's a trade-off between the number of sequences it classifies (or outputs) and the precision.

We evaluated the gene and taxonomy identification performance of our method against established approaches like Kraken2 and MMseqs2. For Kraken2, our training set comprising 540k sequences was indexed, setting the confidence parameters as minimum-hit-groups 1 and confidence as 0. With MMseqs2, we employed the easy-search method on same index dataset, specifying search-type 3 for nucleotide/nucleotide searches, while retaining default values for other parameters. We adjusted our threshold for confidence reporting to observe differences in the number of classified sequences and precision (see Methods)

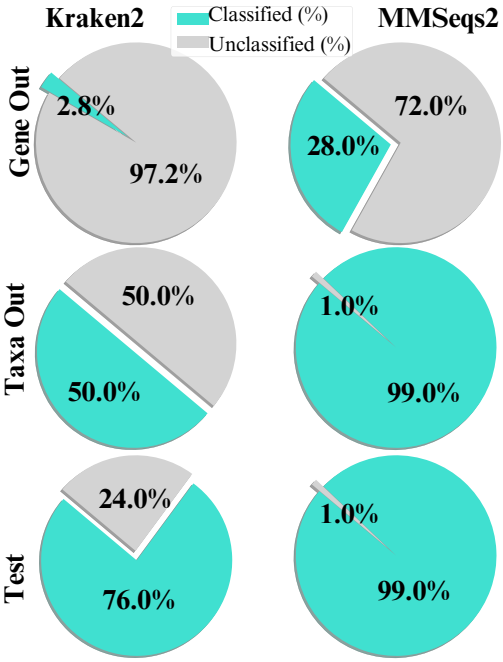
We show the results in Figure 4e, displaying the number of classified sequences for each method. Both Kraken2 and MMseqs2 encounter challenges in classifying genes from the Gene Out dataset, with Kraken2 detecting only 2.8% and MMseqs2 detecting 28% of sequences. This underscores the drawbacks of methodological factors such as sequence similarity and long k-mer search in finding novel sequences. When encountering novel sequences or genes, these methods may encounter significant problems. In Table 4a, we show the precision of our model compared to other models with respect to different threshold values. Even though the number of classified sequences in the Gene Out dataset is very low, the precision for both Kraken2 and MMseqs2 is also very low compared to our model across various thresholds. We could see our model achieved 94% precision and 90% precision when we consider a threshold to return the same number of classified sequences as Kraken2 and MMseqs2, respectively.

For the other dataset, Taxa Out, which consists of sequences from similar genes but from novel taxa, we observe that MMseqs2 is highly effective, classifying 99% of sequences, while Kraken2 classifies only 50%. From a precision perspective, which is crucial as it reflects the accuracy of classifications without being influenced by the dataset's total size, MMseqs2 is particularly strong in identifying similar genes. Our model also performs well in terms of precision on this dataset. On the Test set, Kraken2 classifies more sequences because it shares similar phyla and genes with the training set. Our model, however, still outperforms Kraken2 in precision, even when returning a similar number of sequences. Compared to MMseqs2, our model shows lower precision on this dataset. We also provide our own threshold results. Unlike other methods, our approach selects thresholds by considering both the number of sequences to return and the precision. By calculating a confidence score specific to the dataset, we determine a cut-off based on test sets. This threshold represents an inflection point, balancing the number of classified sequences while maintaining high precision.

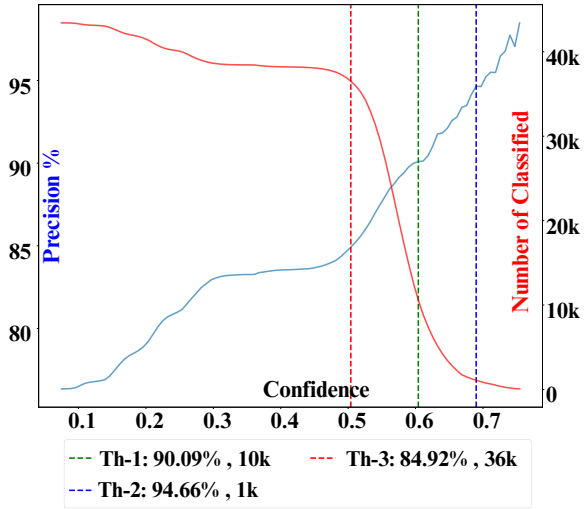
In Figures 4, we have shown our precision vs. confidence and the number of classified sequences vs. confidence for all three datasets. We also highlight the threshold points of ours and two others based on Kraken2 and MMseqs2 number of classified sequences. In all plots, the red threshold indicates the selection based on MMseqs2, and the blue threshold indicates the cutoff based on Kraken2. The red is the threshold we pick based on the infimum between the number of classified sequences and precision. Another interesting observation is that these figures ensure the reliability of the confidence score, which tracks the precision, making it reliable and helping us to be sure about the accuracy of results based on confidence score before doing any classification.

Dataset	Gene Out	Taxa Out	Test
	Precision Phylum (%)	Precision Gene (%)	Precision Phylum (%)
Kraken2	43	N/A	88
Scorpio-6Freq(green)	**94**	**99**	**91**
Mmseq2	50	**99**	**95**
Scorpio-6Freq(blue)	**90**	88	87
Scorpio-6Freq(red)	85	93	94

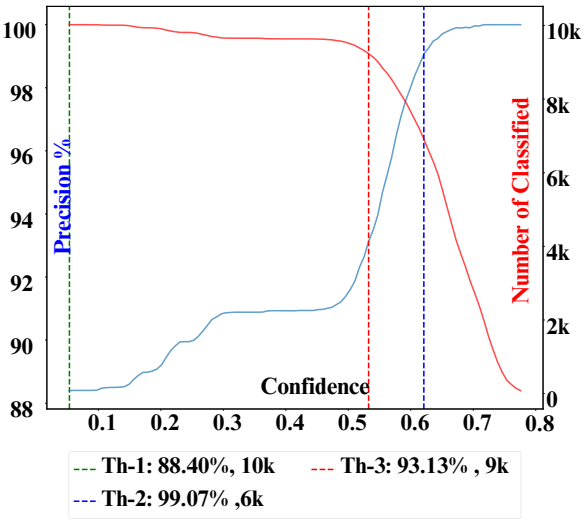
(a) Precision Levels for Different Datasets



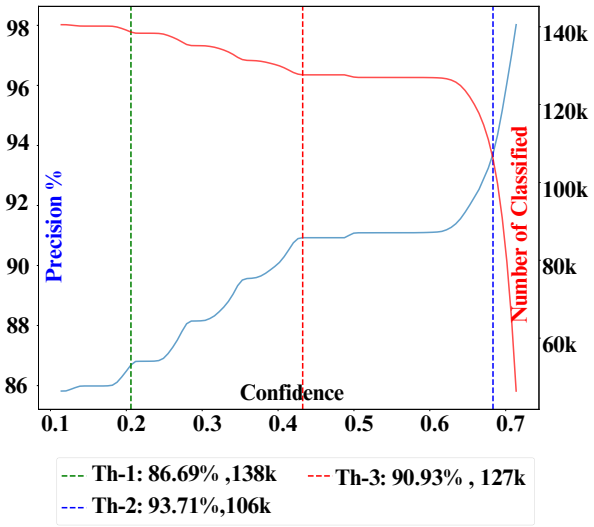
(b) Classification Confidences



(c) Gene Out Dataset: Phylum-Level Confidence Scores



(d) Taxa Out Dataset: Gene-Level Confidence Scores



(e) Test Dataset: Phylum-Level Confidence Scores

Figure 4. We examined Kraken2 and MMseq2 thresholds and their impact on the number of classified instances. As shown in subpanel (b), the number of classified instances varies across datasets for each method. In the "Gene Out" dataset, Kraken2 classified 2.8% of instances, while MMseq2 classified 28%. We compared our model's precision across these different thresholds (subpanels c, d, and e), with green representing Kraken2-like thresholds, blue for MMseq2-like thresholds, and red for our thresholds. In Table (a), at a 2.8% classification rate (green), our Scorpio model achieved 94% precision, compared to Kraken2's 43%. At a 28% classification rate (blue), our Scorpio model achieved 90% precision, while MMseq2 achieved 50%. This analysis demonstrates our model's effectiveness in maintaining high precision while balancing the number of classified instances in novel sequences.

Scorpio embeddings can identify both gene and taxonomy labels of short reads

In this section, we evaluate the effectiveness of Scorpio embeddings in identifying genes and classifying taxonomy in short reads, a critical task in metagenomics. We focus on a target read size of 400 bp, ensuring a fair comparison by training all models on this uniform size. Detailed information about the dataset can be found in the dataset section.

A significant advantage of our approach is the reduced training time compared to DeepMicrobes. As illustrated in (Supplementary Figure 4), the unified objective across all hierarchical levels in Scorpio eliminates the need for separate model training for each task, which is necessary for DeepMicrobes. This streamlined process enhances both efficiency and scalability, making Scorpio a powerful tool in metagenomic analysis.

For the test set (Table 2a), MMseq2 outperformed other methods at various taxonomic levels. Scorpio-BigDynamic excelled at the gene level and was second-best at the phylum level. The 6-mer Frequency representation, despite not using a learning procedure, performed well, indicating its effectiveness for memorization with similar training-testing sequences. Kraken2 showed high precision at lower taxonomic levels. However, Scorpio's hierarchical selection was significantly affected by dataset imbalance at lower taxonomic levels. Some studies suggest that using balanced datasets is essential for training contrastive learning-based models[20].

The test set illustrates methods' memorization capabilities, while the Gene Out and Taxa Out datasets demonstrate generalizability. In the Gene Out set (Table 2a), Scorpio outperformed others at the phylum and class levels in both accuracy and F1 score. DeepMicrobes_family excelled at lower levels like order and family, as it is specifically trained for the family level, unlike our model, which is trained on six levels of hierarchy. DeepMicrobes_gene, trained for the gene level, showed low performance across all taxonomic levels. Our model significantly outperformed MMseq2 and Kraken2, with over 60% improvement at the phylum level. This improvement is due to MMseq2 and Kraken2 struggling with novel gene sequences not in their databases, whereas Scorpio's embeddings, which capture k-mer frequency and sequence similarity, performed much better than sequence search methods. For the Taxa Out set (Table 2), our Scorpio-BigDynamic model outperformed other models in gene classification, achieving 92% accuracy, while MMseqs2 and DeepMicrobes_gene models achieved 78% and 76% accuracy, respectively. This set tests the generalization of the algorithm on more challenging sequences.

With Scorpio embeddings, we can gain valuable interpretability insights into our model's ability to discriminate both genes and the taxonomy of short reads. In Figures 5a and 5b, we compare the t-SNE representations of embeddings from both BigBird and Scorpio-BigEmbed models. In Figure 5a, distinct clusters of genes are clearly visible, a distinction that is not as apparent in the pre-trained model. This highlights the robust nature of our model in differentiating gene sequences. Additionally, when we colorize the visualization based on phyla, as seen in Figure 5b, it becomes evident that Scorpio is more adept at detecting taxonomic information than the pre-trained model. Although there are slight differences between embeddings of each taxon, Scorpio's hierarchical structure enhances its ability to distinguish taxa within each gene cluster. This hierarchical clustering is particularly effective, demonstrating that our model performs significantly better in taxonomy-based differentiation compared to the pre-trained model. One intriguing insight from this visualization, using the 10 most common phyla, is that in Figure 5b, Euryarchaeota displays a nearly distinct representation compared to other prokaryotic types. Despite the model's attempt to cluster the genes, the hierarchical clustering ability and the significant distinction between Archaea and Bacteria prevent the model from clustering sequences from the same genes but different Kingdoms together. This indicates that taxonomic information strongly influences the model's organization. To further explore these insights, we zoomed in on a region in Figure 6 that includes all fragments of the urvA gene. We colorized this region based on different hierarchical levels, each time displaying the 10 most common categories for each level. It is noteworthy how the model discriminates based on each category at lower hierarchical levels. However, it becomes apparent that as we delve into lower hierarchy levels, the sample size for each genus becomes limited, reflecting the initial gene-based discrimination.

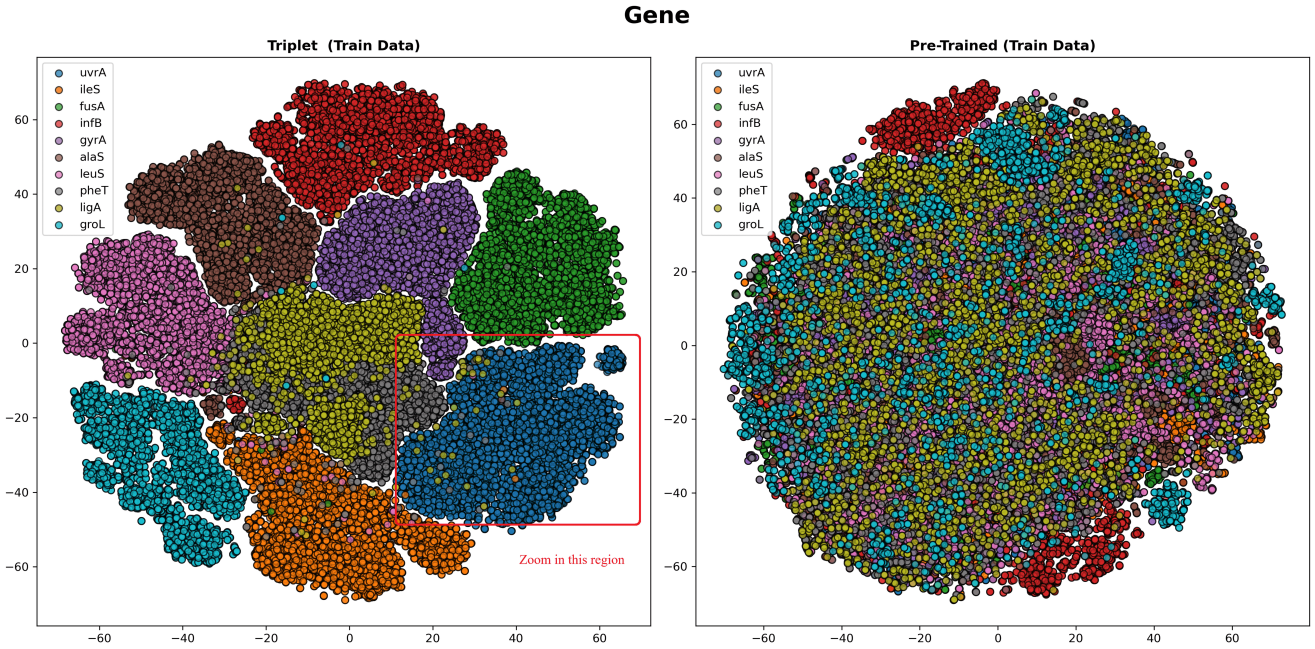
(a)

Level Model	Accuracy (%) phylum	class	order	family	gene	F1-macro (%) phylum	class	order	family	gene
6mer Freq.	90.3	86.1	76.7	65.6	92.4	78.3	69.0	63.4	55.8	91.9
BigBird	72.4	62.7	50.1	41.5	55.6	53.6	44.0	40.1	35.5	54.9
DeepMicrobes_family	72.7	61.6	43.6	30.8	3.1	42.7	28.3	24.2	19.7	2.4
DeepMicrobes_gene	21.5	8.9	2.4	0.7	93.0	4.3	1.4	0.5	0.2	93.2
Mmseqs2	**94.8**	**92.3**	**84.1**	**70.7**	97.7	**85.0**	**75.8**	**69.4**	57.9	97.2
Kraken2	77.8	74.4	66.7	59.6	N/A	70.0	67.2	64.4	**60.9**	N/A
Scorpio-BigEmbed	76.6	66.1	49.8	38.9	74.0	54.3	42.4	37.1	31.4	74.5
Scorpio-6Freq	81.3	70.4	47.7	32.6	92.2	49.8	34.6	29.1	22.9	92.3
Scorpio-BigDynamic	91.0	83.4	63.3	45.8	**98.8**	73.7	53.3	42.9	32.8	**98.9**

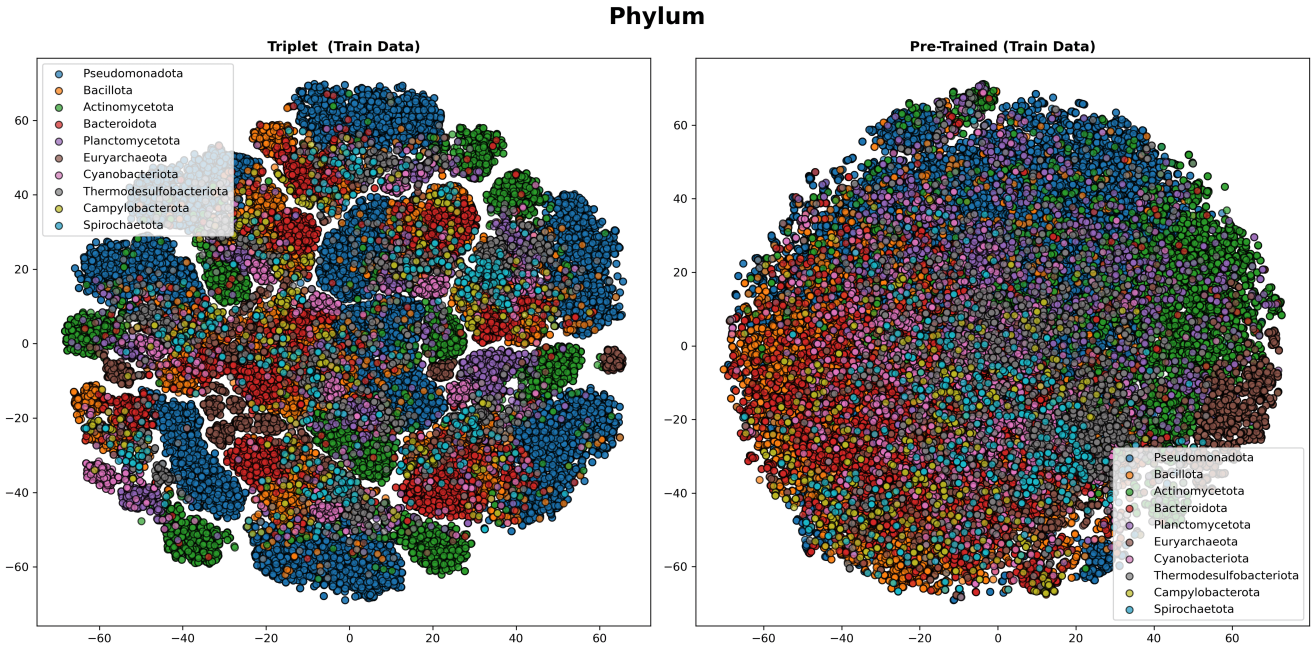
(b)

	Taxonomic Generalization								**Gene Generalization**	
	Accuracy (%)				F1-macro (%)				Accuracy (%)	F1-macro (%)
Model	phylum	class	order	family	phylum	class	order	family	gene	gene
6mer Freq.	47.0	29.9	14.9	8.4	11.4	8.1	6.8	5.1	54.7	52.7
BigBird	51.4	33.9	17.8	10.6	14.3	11.7	9.8	7.5	16.5	14.4
DeepMicrobes_family	55.8	40.3	**22.1**	**13.2**	15.7	**11.8**	**10.3**	**8.3**	2.8	1.9
DeepMicrobes_gene	14.2	5.9	1.8	0.5	2.1	0.9	0.4	0.1	76.0	77.0
Mmseqs2	2.6	1.8	0.8	0.4	1.0	0.6	0.5	0.3	78.5	86.1
Kraken2	0.93	0.63	0.27	0.11	0.2	0.1	0.09	0.06	N/A	N/A
Scorpio-BigEmbed	54.8	37.2	18.0	9.6	14.5	10.0	7.4	5.2	41.7	42.2
Scorpio-6Freq	50.0	31.1	9.4	4.0	10.7	6.1	3.1	1.5	72.4	73.5
Scorpio-BigDynamic	**61.2**	**43.1**	18.3	7.8	**17.8**	10.4	5.6	2.7	**92.3**	**93.1**

Table 2. Short gene length (400bp) results: (a) Memorization Test: Identifying additional examples of training-data-known taxonomy and genes (Test Set); (b) Generalization Test: Taxonomy Generalization (Gene Out Set) and Gene Label Generalization (Taxa Out Set) Tests

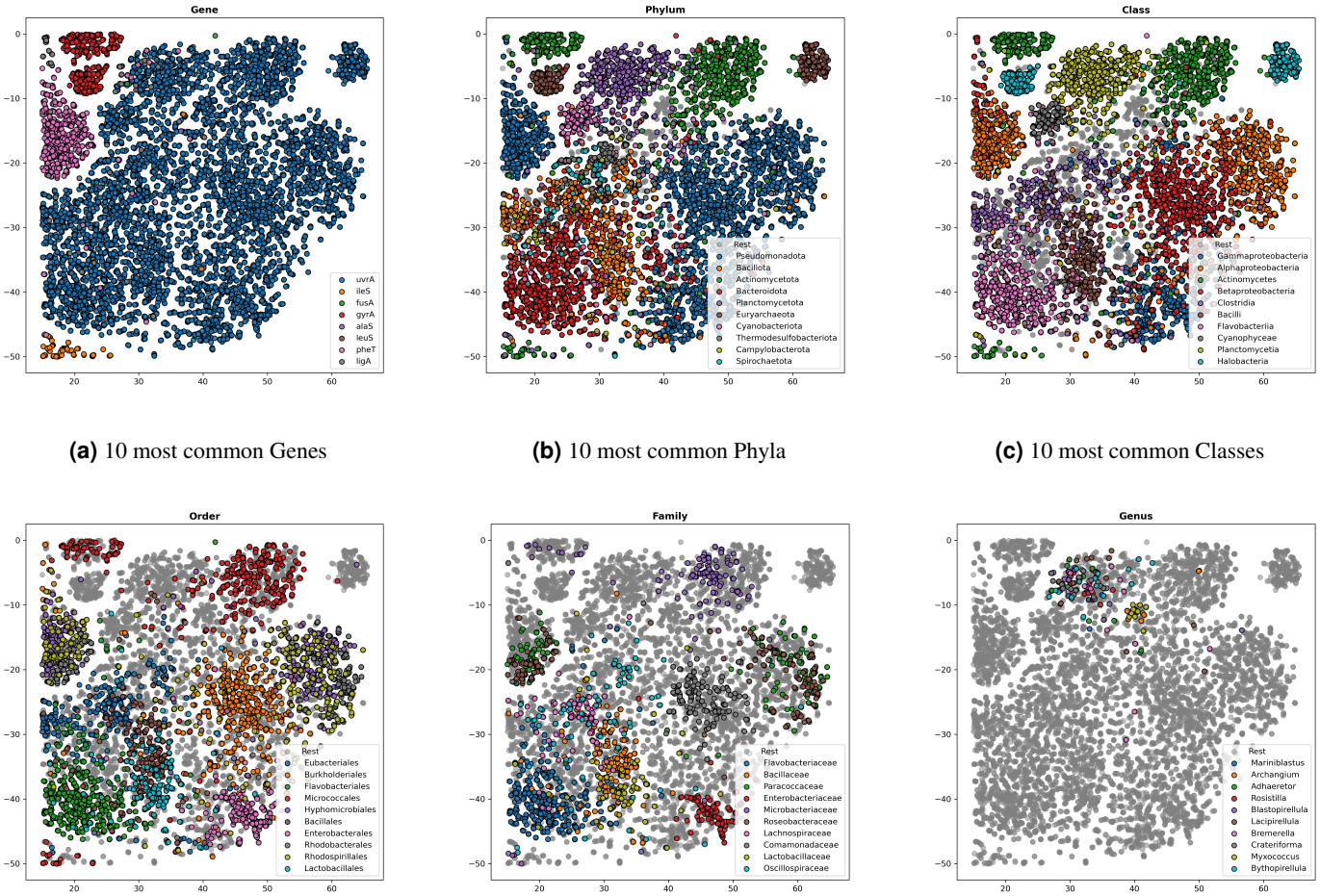


(a)



(b)

Figure 5. (a)t-SNE visualization of embeddings colorized based on the 10 most common phyla in the dataset using the Scorpio-BigEmbed(Triplet) and BigBird(Pre-Trained) Models(b) t-SNE visualization ofembeddings colorized based on the 10 most common gene in the dataset using the Scorpio-BigEmbed(Triplet) and BigBird(Pre-Trained) Models



(a) 10 most common Genes

(b) 10 most common Phyla

(c) 10 most common Classes

(d) 10 most common Orders

(e) 10 most common Families

(f) 10 most common Genera

Figure 6. Analyzing a region which includes the urvA gene in the t-SNE plot of Figure 5a and colorizing for each level.

Discussion

In this study, we introduced the Scorpio framework, which leverages triplet networks with contrastive learning to enhance the analysis of sequenced DNA data. The adoption of nucleotide-based models, unlike traditional protein-focused models[41,42], could revolutionize our understanding of gene expression and protein synthesis by capturing nuances in nucleotide sequences and revealing hidden relationships. By using both pre-trained language models and k-mer frequency embeddings, we aimed to demonstrate the robustness of our framework across multiple types of encoders, all of which showed promising results. We show that Scorpio, despite only one training on gene ORF sequences that have both gene and taxa labels, is among the top-performing algorithms across a variety of tasks. Specifically, Scorpio significantly improves taxonomic and gene classification accuracy, particularly in out-of-domain datasets, thereby showcasing the robustness and generalizability of our method. The superior performance of our model in taxonomic classification can be attributed to the ability of triplet networks to learn more discriminative features by optimizing the distance between positive and negative pairs. This capability is crucial when dealing with the high-dimensional and complex nature of metagenomic data, where traditional alignment-based methods often fall short.

One of the key strengths of the Scorpio framework is its versatility in handling multiple tasks. By extending the triplet network to applications such as antimicrobial resistance prediction , we demonstrated the adaptability of our model to various genomic analysis tasks. Our model generalization surpasses methods like MMseqs2 , Kraken2 and Abricate, which rely on k-mer and sequence similarity searches, by showcasing the effectiveness of embedding-based search.

The ability to transfer learning from one domain to another and fine-tune the model for specific tasks highlights the potential

of our approach for practical applications in health and environmental diagnostics. This adaptability is particularly valuable in metagenomics, where rapid identification and characterization of novel genes and taxa are critical. Our evaluation across diverse datasets with varying gene lengths demonstrates the robustness of our method. Consistent performance improvements across different datasets indicate that our model can effectively generalize from training data to unseen data, a crucial requirement for reliable metagenomic analysis. Additionally, our results show that fine-tuning on short promoter sequences (81 bp) remains among the top-performing methods, showcasing the model's flexibility and precision.

Additionally, our framework incorporates a novel confidence scoring mechanism, which provides a measure of the reliability of the results. This scoring method uses both the distance to the nearest neighbor and the class probabilities derived from neighboring embeddings, ensuring that confidence scores are robust and meaningful. This feature is particularly useful for researchers who need to assess the quality of their classifications and make informed decisions based on the confidence of the results.

Despite the promising results, there are several areas for future research. Firstly, while our model performed well in out-of-domain classification tasks, further improvements could be achieved by incorporating additional sources of biological information, such as functional annotations and protein interaction networks. Integrating such data could enhance the interpretability of the learned embeddings and provide deeper insights into the functional roles of genes and taxa. Secondly, the computational efficiency of our framework could be optimized further. Although the use of FAISS for efficient embedding retrieval was effective, and the possibility of running FAISS on GPU makes it faster, exploring more advanced indexing techniques like ScaNN (Scalable Nearest Neighbors)[43], which has demonstrated some promising results in our comparative analysis with FAISS in the Supplementary Materials, or parallel processing strategies, could reduce the computational overhead and enable the analysis of even larger datasets.

Lastly, while our approach has shown significant promise, it should be extended and fine-tuned to other domains beyond gene/taxa/AMR/promoter classification should be explored. The principles underlying our Scorpio framework could be applied to other types of biological sequence data, such as transcriptomics and proteomics, potentially opening up new avenues for research and application. We also plan to test Scorpio on experimental samples and make our tools available as practical applications for integration into metagenomics pipelines.

In conclusion, our study presents a robust and versatile framework for DNA sequence classification, leveraging triplet networks with contrastive learning and integrated embeddings from PreTrained language models and k-mer frequencies. This approach significantly advances our capacity to process and interpret complex microbiome data, offering valuable insights for health and environmental diagnostics. Future work will focus on further optimizing the model, integrating additional biological information, and exploring its applicability to other domains of genomic research.

Method

Scorpio: Architecture

The architectural design we present is straightforward and effective, featuring three layers in each branch. It maps either a 768-dimensional pretrained embedding or a 4096-dimensional 6-mer frequency vector to a 256-dimensional Scorpio embedding. Each encoder block consists of a linear layer followed by a ReLU activation function, producing 256-dimensional embedding vectors for downstream analyses. This architecture is versatile, accommodating k-mer frequency representations with minimal adjustments. Specifically, by altering the size of the first layer, the model can handle different input dimensions. For example, with 6-mers, the first layer is 4096-dimensional, while the rest of the architecture remains consistent. This flexibility demonstrates the model's ability to adapt to various data representations, as illustrated in Figure 7a. We employ three types of encoders. The first is the 6-mer frequency encoder, which calculates the 6-mer frequency(Scorpio-6Freq)representation and passes it through the architecture. The other two are variations of the BigBird model: one with a trainable final embedding layer(Scorpio-BigDynamic), and another with all layers frozen(Scorpio-BigEmbed). Both BigBird models use an average pooling layer to aggregate embeddings into a single vector, maintaining the model's simplicity and efficiency.

Scorpio: Triplet Training

In the realm of training contrastive learning models, the method for choosing triplet sets (anchor, positive, negative) is crucial for shaping the model's ability to understand semantic connections. Inspired by the Heinzinger et al.[20], our approach involves a refined selection process with adjustments to enhance versatility across different similarity levels.

In our training set, each sample serves as an anchor during every epoch. We deliberately randomize the selection of positive and negative samples for each anchor in each epoch, ensuring the model encounters new instances while revisiting the same triplet set. This repetition guards against overfitting and promotes generalization. Our hierarchical sample selection involves a two-step process depicted in Figure 7b. First, we randomly select the similarity level for each anchor. Based on this similarity level, we then randomly select positive and negative samples. Positive samples come from the same hierarchy level as the anchor, while negative samples are selected from one level higher than the anchor's similarity level. Importantly, similarity at

a lower hierarchical level does not necessarily guarantee similarity at higher levels. For example, choosing sequences from the same phylum for positive and negative samples implies similarity at the gene level as well. This careful consideration ensures that positive samples not only share specific traits but also align across all hierarchical levels, reinforcing a nuanced understanding of the hierarchy. In our special hierarchy of *Gene-Taxa* training, a notable distinction is made at the gene level, which, unlike other taxonomy levels, is not part of the natural hierarchy. Our model actively separates genes from taxonomy, enhancing its ability to differentiate between genetic characteristics and broader taxonomic classifications. Additionally, we drew inspiration from batch-hard sampling[20] to prioritize samples and batches, detecting harder instances throughout training that exhibit the most distance between anchor-negative and anchor-positive pairs.

A key improvement in our framework is its independence from the type and number of hierarchical levels. Our innovation lies in the framework's ability to handle various hierarchies and adapt to changes in the hierarchy structure, such as adding or removing levels or incorporating new tasks. To test this, we trained the model on a promoter dataset containing just one level—whether it is a promoter or not—and it demonstrated the framework's adaptation capabilities. We conducted extensive studies to determine the optimal batch size, number, and order of levels for hierarchical sampling. Detailed results of these studies are provided in the supplementary material.

We utilized the Margin Ranking Loss technique during our model training to optimize a novel embedding space. This approach aimed to draw anchor-positive pairs closer together, effectively reducing their distance, while simultaneously pushing anchor-negative pairs further apart, thereby increasing their Euclidean distance.

$$
\begin{aligned}
\mathrm{AN}_i &= \sqrt{(\mathrm{anchor}_i - \mathrm{negative}_i)^2} \\
\mathrm{AP}_i &= \sqrt{(\mathrm{anchor}_i - \mathrm{positive}_i)^2} \\
\mathrm{loss} &= \frac{1}{N}\sum_{i=1}^{N} \max\left(0, -y_i \cdot (\mathrm{AN}_i - \mathrm{AP}_i) + \mathrm{margin}\right)
\end{aligned}
\tag{1}
$$

Here, AN_i represents the Euclidean distance between the anchor and negative embeddings, while AP_i denotes the Euclidean distance between the anchor and positive embeddings. The loss function loss averages the maximum of zero and the difference between the anchor-negative distance and the anchor-positive distance, weighted by the labels y_i and a margin parameter.

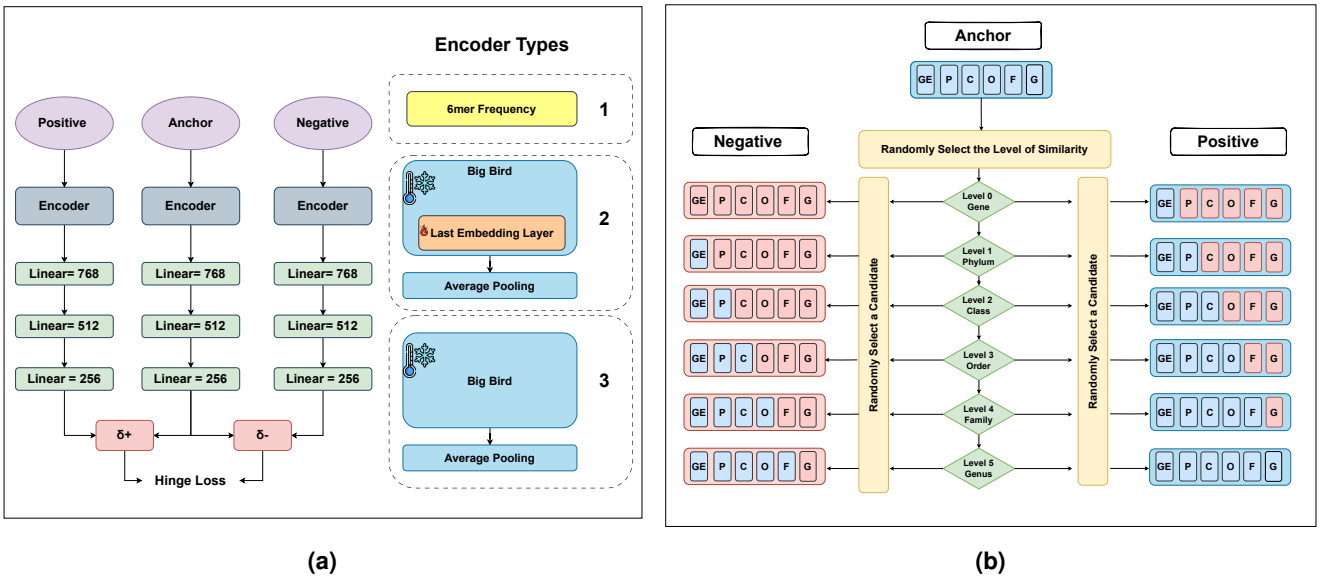


Figure 7. 7a Model Architecture: Each branch transforms 768 (or 4096)-dimensional encoded embeddings to a 256-dimensional triplet vector. We have three types of encoders: BigBird embedding vectors, 6-mer Freq., and a model where BigBird is used with the embedding layer. 7b The diagram illustrates the hierarchical selection process for a positive and negative for an anchor in our gene-taxa dataset. First, the level of similarity is randomly determined (e.g., Ge: Gene, P: Phylum, C: Class, O: Order, F: Family, G: Genus). Positive samples match the anchor at this level, while negative samples are chosen from one level up to ensure dissimilarity.

Confidence Score

Our objective is to derive a confidence score for each prediction at every hierarchical level. To accomplish this, we employed a hybrid methodology leveraging two key techniques: I) A confidence predictor, which calculates the confidence score based on the raw distance value of the query point to the best match in the training set. II) Utilization of class probabilities derived from neighborhood embeddings specific to each query. Consider Figure 8a, which visually depicts the necessity of integrating both metrics in determining the confidence score. For instance, comparing two query points, yellow and green, which exhibit similar decision boundaries and nearest neighbors, the yellow point demonstrates a significantly lower distance to its best hit compared to the green point. Consequently, the confidence score for the yellow query should exceed that of the green query. Further, let's analyze the scenario involving the yellow and purple stars. Although both share equidistant nearest neighbors, the surrounding point probability differs; purple records 0.4 while yellow boasts 0.6. Consequently, the confidence score for the yellow query should surpass that of the purple query. However, outliers such as the blue star, which lie considerably distant from the training points, challenge the importance of neighborhood class probability due to their significant deviation from decision boundaries. In such outliers' cases, confidence scores may solely rely on distance to the nearest neighbor.

In light of these insights, we formulated a function to estimate confidence scores by harnessing the information encapsulated within neighborhood training points for each query. Let D represent the set of distances between query points and their closest target points in the validation set (conf_set). This conf_set includes instances from the training set but not appearing in the training set used for validation, ensuring that the distances are representative of the data distribution. Let H be the set of all hierarchical levels. For a specific level $\ell \in H$, ensure that the predictions at all upper levels $u \in H$ (where $u < \ell$) are correct. To calculate the F1 score for each level, we further filter the data based on hierarchical levels. For the most upper level, such as gene, the F1 score is calculated as an accuracy F1 score. However, for lower levels, the dataset is split based on upper levels, and we should consider inner distances as reliable distances to calculate the confidence score. The dataset is further filtered to include only rows where the predictions at all upper levels are correct:

$$\text{conf_set}_{\text{filtered}} = \{e \in \text{conf_set}_{\text{original}} \mid \forall u \in H, u < \ell, e[u_{\text{query}}] = e[u_{\text{target}}]\} \tag{2}$$

Using this filtered data, the precision and recall for the current level ℓ are calculated, leading to the final F1 score for the specified threshold. For each distance threshold t_i in D, we filter the conf_set data based on the condition that the distance

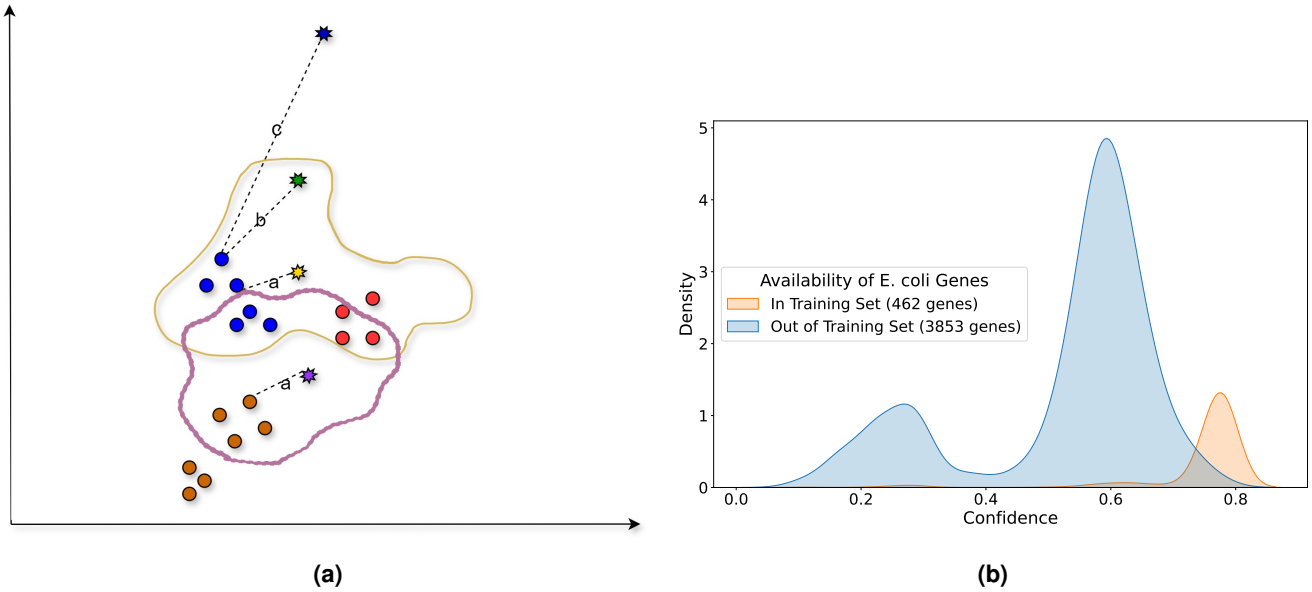


Figure 8. 8a Confidence Score Illustration: Circular points represent the training set with different colors for different classes, and star points represent the query sets. The distance of $a < b \ll c$, but the confidence score could vary. 8b The KDE plot illustrates the distribution of confidence scores for genes of an *E. coli*. Genes are categorized based on their availability in the training set, with "In Training Set (462 genes)" indicating genes present in the training data and "Out of Training Set (3853 genes)" indicating genes absent from the training data.

between query points and their closest target points is less than or equal to t_i. This filtered dataset is used to calculate the F1

score $F1(t_i)$ for the current hierarchical level ℓ, considering the correctness of predictions at upper levels. The F1 score is calculated as follows:

$$F1(t_i) = \frac{2 \times \text{precision}(t_i) \times \text{recall}(t_i)}{\text{precision}(t_i) + \text{recall}(t_i)} \tag{3}$$

where precision and recall are defined as:

$$\text{precision}(t_i) = \frac{\text{TP}(t_i)}{\text{TP}(t_i) + \text{FP}(t_i)} \tag{4}$$

$$\text{recall}(t_i) = \frac{\text{TP}(t_i)}{\text{TP}(t_i) + \text{FN}(t_i)} \tag{5}$$

Here, TP, FP, and FN denote the true positive, false positive, and false negative counts, respectively. Once we have calculated the F1 scores for all thresholds in D, we use them as the target variable y and the corresponding distance thresholds as the input variable X to train a simple neural network to predict F1 based on distance threshold. This neural network, denoted as N, $N : t_i \rightarrow F1$ is a function that maps distance thresholds to F1 scores: The architecture of the neural network N is a fully connected feedforward network with multiple hidden layers, with the output layer having a single neuron since it's a regression model. The activation function in the hidden layers is ReLU. The neural network is trained using Mean Squared Error (MSE), which measures the difference between the predicted F1 scores and the actual F1 scores:

$$\text{Loss} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{6}$$

After training the model, we use it to predict the confidence score $\hat{C}_1$ for a given distance threshold $\hat{t}$, providing us with a confidence score $\hat{C}_1 = N(\hat{t})$. This kind of representation helps us ensure that the confidence score is bounded between 0 and 1, whereas comparing with raw distance values, which are not normalized, may not provide meaningful low and high values. Additionally, very low F1 scores typically indicate low confidence, while high F1 scores suggest high confidence. This can be interpreted akin to a threshold finder. Next, we calculate class probabilities of target as $\hat{C}_2$, which represents the number of times class i (the most common class in the neighborhood) was present among the total number of nearest neighbors considered, K. $\hat{C}_2 = \frac{m_i}{K}$, and then we compute the final confidence as follows: $C_{\text{total}} = \hat{C}_1 \times \hat{C}_2$.

Dataset

Gene and Taxonomy Dataset

We obtained the complete Basic genome dataset using Woltka's pipeline[24], comprising 4634 genomes. A specific characteristic of this dataset is that only one genome is included for each genus, making it unique and challenging. After considering the taxonomic properties, we attempted to download all CDS files from the NCBI database for the Basic genome dataset. Subsequently, we extracted all coding sequences (CDS) from the Basic genomes dataset, resulting in 8 million distinct CDS. We then focused our study on bacteria and archaea, excluding genomes from viruses and fungi, which often lack sufficient gene information.

To ensure the initial annotation accuracy of genes in the dataset, we filtered out hypothetical proteins, uncharacterized protein and sequences lacking gene labels. Another issue encountered during our observation of gene labels in NCBI was the potential unreliability of gene names, possibly due to misspellings or differences in nomenclature. To address this, we retained only genes with more than 1000 samples and also imposed a filter to ensure a minimum number of sequences per phylum, considering only those with more than 350 sequences. This curation process yielded a dataset of 800,318 gene sequences, representing 497 genes across 2046 different genera.

To assess the generalizability of our model, we deliberately constructed four types of datasets. One dataset for training, collectively referred to as the *Train Set*. Additionally, we created another dataset for testing, referred to as the *Test Set*, which comprises the same classes at all levels (same genus and same gene with *Train Set*) but with different samples. We also excluded 18 different phyla, designated as the *Taxa Out Set*, which have the same gene as the training set but from different phyla. Furthermore, we excluded 60 different genes from the *Train Set*, all originating from the same phyla, forming the *Gene Out Set*. In all testing sets, we also made sure to include only CDS that have only one representation for a genome, because we observed that once we have downloaded the CDS files for different genomes like Hungatella hathewayi species, we may

have multiple gene sequences for one type of gene (*lepB*, for instance, has 34 representations for this species). So, we have removed such genes from our analysis because we found that in some species it may have multiple gene representations in NCBI but these genes may not be from the same species[44]. Therefore, to add more validity to our test datasets, we removed those sequences from the analysis as well. Our goal was to include holdout sets that represent diverse aspects, allowing us to evaluate the model's performance with unseen data. For detailed information regarding the exact number of samples and the range of values per class at each level, please refer to the appendix.

For the short read dataset, we extracted a total of 1.6 million 400bp fragments from an 800k sequence gene dataset. Our approach involved selecting 400bp fragments from various regions of the gene sequences, ensuring a minimum 50bp distance between them. This was done using the range ***Range:[0, Gap: 50, length(genome_sequence)]***. We believed this strategy was essential to avoid selecting fragments with minimal base pair differences and to mimic sequences that do not start with an open reading frame. Some of these fragments are in-frame, which is important because, in real metagenomics sequences, fragments can originate from any part of the genome and are not necessarily in the frame of an open reading frame. However, we used our gene dataset to ensure that the short read dataset also contains gene information beyond taxonomy.

Promoter Dataset

In this study, we utilized the promoter dataset provided by Ligeti et al.[31] for training and testing our promoter prediction models. The promoter dataset by Ligeti et al. consists of experimentally validated promoter sequences primarily drawn from the Prokaryotic Promoter Database (PPD), which includes sequences from 75 different organisms. To ensure a balanced and comprehensive dataset, non-promoter sequences were generated using higher and zero-order Markov chains.Additionally, an independent test set focusing on E. coli sigma70 promoters[45] was used to benchmark the models against established datasets. The non-promoter sequences were constructed using three methods: coding sequences (CDS) extracted from the NCBI RefSeq database, random sequences generated based on a third-order Markov chain, and pure random sequences generated using a zero-order Markov chain. The balanced distribution of these non-promoter sequences (40% from CDS, 40% from Markov-derived random sequences, and 20% from pure random sequences) was crucial for thorough evaluation and robust model training. The inclusion of the independent E. coli sigma70 test set, curated by Cassiano and Silva-Rocha (2020), further validated the effectiveness of the promoter prediction models, ensuring no overlap with the training data and providing a rigorous benchmark for model performance.

Antimicrobial Resistance Dataset

We utilized an integrated dataset combining the CARD[28] v2 and MEGARes[27] v3 datasets, referred to as the Antimicrobial Resistance Dataset, following methodologies from previous studies[46]. Classes with fewer than 15 samples were removed as they hindered the attainment of meaningful results during data splitting. The remaining data was divided into 75% for training, 20% for testing, and 5% for validation. After integrating the data using the EBI (European Bioinformatics Institute) ARO (Antibiotic Resistance Ontology) ontology search, it was similarly divided. Classes that yielded non-meaningful results were also excluded. The MEGARes dataset comprises 9733 reference sequences, 1088 SNPs, 4 antibiotic types, 59 resistance classes, and 233 mechanisms. The CARD dataset includes 5194 reference sequences, 2005 SNPs, 142 drug classes, 331 gene families, and 10 resistance mechanisms. The EBI ARO ontology provides hierarchical group information for genes, allowing gene family class information to be integrated into a higher-level hierarchy. For MEGARes, there are 589 gene family text information classes, while CARD has 331. There are 300 and 166 datasets with only one sample in their respective gene family classes for MEGARes and CARD, respectively.Resistance mechanisms categories are integrated based on the 6 categories of CARD. The original 8 categories were reduced to 6 by excluding various class combinations and those with very few samples. Drug classes are integrated using 9 common drug classes found in competing models. The integration is based on names and theories and has been validated by experts in the field.

FAISS

In our framework, FAISS (Facebook AI Similarity Search)[23] played a pivotal role as a cornerstone element for conducting similarity search tasks. This versatile library, designed by Johnson et al. (2019)[23], specializes in facilitating approximate nearest neighbor search (ANNS) on vector embeddings, addressing various domains and applications. Leveraging FAISS, we efficiently conducted similarity searches on our collection of query embeddings, benefiting from its indexing techniques that involve preprocessing, compression, and non-exhaustive search methods detailed in Supplementary Table 3. These strategies enabled swift retrieval of nearest neighbors, whether based on Euclidean distance or highest dot product[47]. This streamlined approach greatly aided in identifying similar embeddings within our expansive dataset while effectively managing computational resources and memory overhead.. Additionally, FAISS provides optimized versions for both CPU and GPU platforms[43] with the latter proving particularly advantageous, especially when dealing with high-dimensional vectors exceeding 1000 dimensions. This GPU acceleration, noted for its significant performance boost, accelerated our similarity search tasks, especially vital for processing large-scale datasets, leveraging the parallel processing capabilities inherent to GPUs.

Pre-training the BigBird Model

In this study, we utilized the BigBird model, a transformer-based architecture specifically designed to handle long sentences, to represent our gene sequences. The BigBird model enhances the standard transformer model by incorporating sparse attention mechanisms, allowing it to efficiently manage much longer contexts, which is particularly advantageous for genomic data characterized by lengthy sequences and complex dependencies[22]. We follow the approach in MetaBERTa[16] to select the parameters for the BigBird model. We trained the BigBird model using a sequence length of 4096, with a batch size of 16, and an embedding dimension of 768. The feed-forward neural network within the transformer layers was configured with a dimension of 3072. The model employed 4 attention heads and comprised 4 transformer layers, facilitating the learning of hierarchical representations. The Adam optimizer with an epsilon value of 1e-8 was used for training. Training was conducted over 2 epochs. The vocabulary size of our model is based on 6-mer, which corresponds to 4096 tokens. This selection can be attributed to the fact that each 6-mer represents two codons, which correspond to amino acids, ensuring more functional information[16]. These configurations were selected to optimize the model's performance for genomic sequence analysis, leveraging its unique capabilities to manage and learn from long sequences efficiently.

Acknowledgements

This work is supported by National Science Foundation under Grant Numbers #1936791, #1919691 and #2107108. We thank the University Research Computing Facility for their paid services.

Data availability

All data used in this manuscript originates from publicly available databases. The sequence data used for pre-training and Scorpio of Gene-Taxa are available in the Zenodo repository https://zenodo.org/records/12175913. Additionally, the dataset for promoter detection was downloaded from the ProkeBERT repository on Hugging Face, available https://huggingface.co/neuralbioinfo.

Code availability

The source code for the paper is freely available online under an open source license https://github.com/EESI/Scorpio .

References

1. McIntyre, A. B. *et al.* Comprehensive benchmarking and ensemble approaches for metagenomic classifiers. *Genome biology* **18**, 1–19 (2017).
2. Wood, D. E. & Salzberg, S. L. Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome biology* **15**, 1–12 (2014).
3. Miller, D., Stern, A. & Burstein, D. Deciphering microbial gene function using natural language processing. *Nat. Commun.* **13**, 5731 (2022).
4. Wichmann, A. *et al.* Metatransformer: deep metagenomic sequencing read classification using self-attention models. *NAR Genomics Bioinforma.* **5**, lqad082 (2023).
5. Wood, D. E., Lu, J. & Langmead, B. Improved metagenomic analysis with kraken 2. *Genome biology* **20**, 1–13 (2019).
6. Zhao, Z., Cristian, A. & Rosen, G. Keeping up with the genomes: efficient learning of our increasing knowledge of the tree of life. *BMC Bioinforma.* **21**, 412, 10.1186/s12859-020-03744-7 (2020).
7. Ounit, R., Wanamaker, S., Close, T. J. & Lonardi, S. Clark: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers. *BMC Genomics* **16**, 236, 10.1186/s12864-015-1419-2 (2015).
8. Zheng, W. *et al.* Sense: Siamese neural network for sequence embedding and alignment-free comparison. *Bioinformatics* **35**, 1820–1828 (2019).
9. Liang, Q., Bible, P. W., Liu, Y., Zou, B. & Wei, L. DeepMicrobes: taxonomic classification for metagenomics with deep learning. *NAR Genomics Bioinforma.* **2**, lqaa009, 10.1093/nargab/lqaa009 (2020). https://academic.oup.com/nargab/article-pdf/2/1/lqaa009/34054182/lqaa009.pdf.
10. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
11. Sarzynska-Wawer, J. *et al.* Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Res.* **304**, 114135, https://doi.org/10.1016/j.psychres.2021.114135 (2021).

12. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I. *et al.* Improving language understanding by generative pre-training. *OpenAI* (2018).

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

14. Madani, A. *et al.* Large language models generate functional protein sequences across diverse families. *Nat. Biotechnol.* 1–8 (2023).

15. Brandes, N., Ofer, D., Peleg, Y., Rappoport, N. & Linial, M. Proteinbert: a universal deep-learning model of protein sequence and function. *Bioinformatics* **38**, 2102–2110 (2022).

16. Refahi, M., Sokhansanj, B. & Rosen, G. Leveraging large language models for metagenomic analysis. In *2023 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*, 1–6, 10.1109/SPMB59478.2023.10372773 (2023).

17. Zhao, Z. *et al.* Learning, visualizing and exploring 16s rrna structure using an attention-based deep neural network. *PLOS Comput. Biol.* **17**, 1–36, 10.1371/journal.pcbi.1009345 (2021).

18. Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607 (PMLR, 2020).

19. Lu, A. X. *et al.* Discovering molecular features of intrinsically disordered regions by using evolution for contrastive learning. *PLOS Comput. Biol.* **18**, e1010238 (2022).

20. Heinzinger, M. *et al.* Contrastive learning on protein embeddings enlightens midnight zone. *NAR genomics bioinformatics* **4**, lqac043 (2022).

21. Memon, S. A., Khan, K. A. & Naveed, H. Hecnet: a hierarchical approach to enzyme function classification using a siamese triplet network. *Bioinformatics* **36**, 4583–4589 (2020).

22. Zaheer, M. *et al.* Big bird: Transformers for longer sequences. *Adv. neural information processing systems* **33**, 17283–17297 (2020).

23. Johnson, J., Douze, M. & Jégou, H. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data* **7**, 535–547 (2019).

24. Zhu, Q. *et al.* Phylogeny-aware analysis of metagenome community ecology based on matched reference genomes while bypassing taxonomy. *Msystems* **7**, e00167–22 (2022).

25. Duan, H. N., Hearne, G., Polikar, R. & Rosen, G. L. The naive bayes classifier++ for metagenomic taxonomic classification–query evaluation. *bioRxiv* 2024–06 (2024).

26. Steinegger, M. & Söding, J. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat. biotechnology* **35**, 1026–1028 (2017).

27. Bonin, N. *et al.* Megares and amr++, v3. 0: an updated comprehensive database of antimicrobial resistance determinants and an improved software pipeline for classification using high-throughput sequencing. *Nucleic acids research* **51**, D744–D752 (2023).

28. Jia, B. *et al.* Card 2017: expansion and model-centric curation of the comprehensive antibiotic resistance database. *Nucleic acids research* gkw1004 (2016).

29. Seemann, T. Abricate: Mass screening of contigs for antimicrobial and virulence genes; 2018 (2019).

30. Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. Basic local alignment search tool. *J. molecular biology* **215**, 403–410 (1990).

31. Ligeti, B., Szepesi-Nagy, I., Bodnár, B., Ligeti-Nagy, N. & Juhász, J. Prokbert family: genomic language models for microbiome applications. *Front. Microbiol.* **14**, 1331233 (2024).

32. Cassiano, M. H. A. & Silva-Rocha, R. Benchmarking bacterial promoter prediction tools: Potentialities and limitations. *Msystems* **5**, 10–1128 (2020).

33. Santos-Zavaleta, A. *et al.* Regulondb v 10.5: tackling challenges to unify classic and high throughput knowledge of gene regulation in e. coli k-12. *Nucleic acids research* **47**, D212–D220 (2019).

34. Sharp, P. M. & Li, W.-H. The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic acids research* **15**, 1281–1295 (1987).

35. Yu, C.-H. *et al.* Codon usage influences the local rate of translation elongation to regulate co-translational protein folding. *Mol. cell* **59**, 744–754 (2015).

36. Liu, Y. A code within the genetic code: codon usage regulates co-translational protein folding. *Cell Commun. Signal.* **18**, 145 (2020).

37. Jacques, F., Bolivar, P., Pietras, K. & Hammarlund, E. U. Roadmap to the study of gene and protein phylogeny and evolution—a practical guide. *Plos one* **18**, e0279597 (2023).

38. Van der Maaten, L. & Hinton, G. Visualizing data using t-sne. *J. machine learning research* **9** (2008).

39. Detlefsen, N. S., Hauberg, S. & Boomsma, W. Learning meaningful representations of protein sequences. *Nat. communications* **13**, 1914 (2022).

40. Wu, M., Chatterji, S. & Eisen, J. A. Accounting for alignment uncertainty in phylogenomics. *PloS one* **7**, e30288 (2012).

41. Asgari, E. & Mofrad, M. R. Continuous distributed representation of biological sequences for deep proteomics and genomics. *PloS one* **10**, e0141287 (2015).

42. Refahi, M. S., Mir, A. & Nasiri, J. A. A novel fusion based on the evolutionary features for protein fold recognition using support vector machines. *Sci. Reports* **10**, 14368 (2020).

43. Guo, R. *et al.* Accelerating large-scale inference with anisotropic vector quantization. In *International Conference on Machine Learning*, 3887–3896 (PMLR, 2020).

44. Hernández-Juárez, L. E. *et al.* Analyses of publicly available hungatella hathewayi genomes revealed genetic distances indicating they belong to more than one species. *Virulence* **12**, 1950–1964 (2021).

45. Anzolini Cassiano, M. H. & Silva-Rocha, R. Benchmarking bacterial promoter prediction tools: potentialities and limitations. *Msystems* **5** (2020).

46. Yoo, H., Sokhansanj, B., Brown, J. R. & Rosen, G. Predicting anti-microbial resistance using large language models. *arXiv:2401.00642 [cs.CL]* (2024). Or arXiv:2401.00642v1 [cs.CL] for this version.

47. Douze, M. *et al. The FAISS Library* (2024).

Author Contributions

M.S.R. conceived and designed the study, developed the algorithms and software, analyzed the data, interpreted results, wrote the manuscript, and prepared figures. B.A.S. conceived and designed the study, interpreted the data, analyzed the data, and reviewed the manuscript. J.C.M. designed the study and interpreted the data. J.R.B. designed the study, analyzed the data, interpreted results, and reviewed the manuscript. H.Y. prepared the data and reviewed the manuscript, G.H. ran the software and reviewed the manuscript. G.L.R. conceived and designed the study, interpreted results, analyzed the data,supervised the work, and wrote the manuscript.

Competing interests

The authors declare no competing interests.