Transfer Learning on Protein Language Models Improves Antimicrobial Peptide Classification

Elias Georgoulis[1,2], Michaela Areti Zervou[3,4], and Yannis Pantazis[1,*]

[1]Institute of Applied and Computational Mathematics, FORTH, Heraklion, 700 13, Greece
[2]Department of Mathematics & Applied Mathematics, University of Crete, Heraklion, 700 13, Greece
[3]Department of Computer Science, University of Crete, Heraklion, 700 13, Greece
[4]Institute of Computer Science, FORTH, Heraklion, 700 13, Greece
[*]email: pantazis@iacm.forth.gr

ABSTRACT

Antimicrobial peptides (AMPs) are essential components of the innate immune system in humans and other organisms, exhibiting potent activity against a broad spectrum of pathogens. Their potential therapeutic applications, particularly in combating antibiotic resistance, have rendered AMP classification a vital task in computational biology. However, the scarcity of labeled AMP sequences, coupled with the diversity and complexity of AMPs, poses significant challenges for the training of standalone AMP classifiers. Self-supervised learning has emerged as a powerful paradigm in addressing such challenges across various fields, leading to the development of Protein Language Models (PLMs). These models leverage vast amounts of unlabeled protein sequences to learn biologically relevant features, providing transferable protein sequence representations (embeddings), that can be fine-tuned for downstream tasks even with limited labeled data. This study evaluates the performance of several publicly-available PLMs in AMP classification utilizing transfer learning techniques and benchmarking them against state-of-the-art neural-based classifiers. Our key findings include: (a) Model scale is crucial, with classification performance consistently improving with increasing model size; (b) State-of-the-art results are achieved with minimal effort utilizing PLM embedding representations alongside shallow classifiers; and (c) Classification performance is further enhanced through efficient fine-tuning of PLMs' parameters. Code showcasing our pipelines is available at https://github.com/EliasGeorg/PLM_AMP_Classification.

Introduction

Antimicrobial peptides (AMPs) are naturally occurring peptides found in a wide range of living organisms. They serve as a critical line of defense in the innate immune system against pathogens[1,2]. With their broad-spectrum antimicrobial activity, AMPs have garnered attention as promising candidates for therapeutic applications, particularly in addressing the global challenge of antibiotic resistance[3]. However, AMP discovery and characterization remain difficult due to their sequence diversity, limited availability in public databases (only a few thousand are known), and intricate structure-function relationships[1]. AMP classification has predominantly relied on supervised learning, including approaches that either leverage manually curated features[4–11] or sequence-based neural networks[12–16]. While these methods have achieved reasonable success, they are often hindered by the *scarcity of labeled AMP data*, leading to issues such as overfitting and suboptimal generalization[17,18]. Additionally, these methods typically rely on complex architectures and training procedures that are challenging to implement and require advanced expertise.

Another challenge, distinct from the limitations of supervised approaches, is how to effectively leverage the vast amounts of publicly-available protein sequences while capturing their inherent complexity –a problem that has been recently addressed through self-supervised learning. Self-supervised learning represents a significant paradigm shift in deep learning, enabling researchers to extract valuable insights from large, unlabeled datasets. Largely inspired by the success of Large Language Models (LLMs) in natural language processing[19–25], Protein Language Models (PLMs)[26–31] have been established as powerful tools in AI-driven protein research. Trained on large-scale datasets and leveraging significant computational resources, PLMs effectively model the protein landscape and have excelled at learning meaningful protein representations. Like LLMs, which have demonstrated proficiency as few-shot learners through transfer learning techniques[19–23], PLMs have shown similar few-shot learning capabilities achieving state-of-the-art performance across a variety of tasks, including protein structure prediction[29,30], function annotation[26,32,33], and de novo protein design[31,34].

This study conducts an extensive comparative analysis of various publicly-available PLMs[26,27,29,35–38], focusing on their performance in AMP classification through transfer learning techniques. We implement and evaluate two primary transfer learning approaches: embedding-based transfer learning, which integrates PLM-generated embeddings with shallow classifiers,

and parameter fine-tuning, which efficiently adapts the weights of the PLM to the specific task. Utilizing multiple AMP datasets, we thoroughly evaluate a series of pre-trained PLMs aiming to address the following questions: (i) Which PLMs deliver the most accurate predictions for the AMP classification task? (ii) Which characteristics of the PLMs highly correlate with performance metrics? (iii) How does transfer learning on PLMs compares with state-of-the-art standalone neural-based AMP classifiers[7,8,10–12]? (iv) Can efficient fine-tuning of PLMs' parameters further enhance performance?

Our findings demonstrate that transfer learning on PLMs consistently outperforms current state-of-the-art models for AMP classification, even with minimal fine-tuning and computational resources. This establishes a robust, scalable, and user-friendly framework for AMP classification, reinforcing the use of PLMs as transformative tools for advancing AI-driven protein research.

Methods

Two distinct transfer learning pipelines were implemented to leverage the capacity of PLMs for the task of AMP classification: an embedding-based approach and a parameter fine-tuning approach. Figure 1 demonstrates the two pipelines. The upper diagram shows the embedding-based approach, where the pre-trained model generates fixed-size vectors for input sequences, which are subsequently used as features for a separate, trainable classifier. The lower diagram illustrates the fine-tuning approach, where the PLM's weights are fine-tuned by updating them through an efficient adaptation mechanism, allowing the model to learn specific traits related to the AMP classification task.

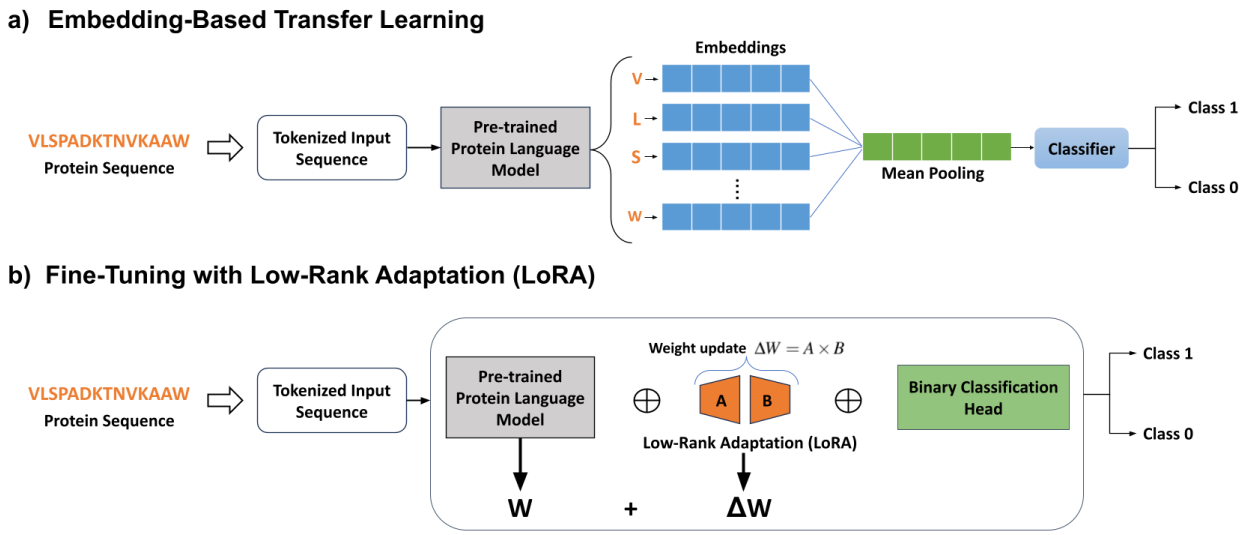

Figure 1. Illustration of the transfer learning pipelines. (a) Protein sequences are initially fed into a pre-trained PLM to generate embeddings for each amino acid within the sequence. To obtain fixed-size representations for each sequence, the mean of these embeddings across the sequence length dimension is computed. These embeddings, along with their corresponding protein class labels, are used to train a classifier to perform AMP classification. (b) A binary classification head is appended to the final layer of the PLM's encoder. New trainable parameters (ΔW) are introduced in a parameter-efficient manner called Low-Rank Adaptation (LoRA). These parameters are trained to adapt the PLM specifically for the AMP classification task.

Protein Language Models

PLMs constitute a class of deep neural networks trained on massive amounts of protein sequence data, employing self-supervision at a large scale. This study evaluates the representational capabilities of publicly-available, pre-trained PLMs enlisted in Table 1 with respect to their performance in AMP classification. PLMs, as the great majority of recent sequence-to-sequence models, have been dominated by the transformer architecture and the associated self-attention mechanism[39]. This mechanism enables the models to process sequences in an adaptive, non-linear and context-sensitive manner, which could be particularly advantageous for deciphering the complex interactions between amino acids. Indeed, the transformer architecture have consistently excelled at identifying long-range dependencies within sequences.

In this work, we explore various models from the Evolutionary Scale Modeling (ESM)[26] family and its sequel, ESM2[29]. While ESM2 retains core features of ESM, it introduces improvements in both architecture and training process. ESMs are trained on evolutionary-scale data utilizing a variant of the Bidirectional Encoder Representations from Transformers[40] (BERT) architecture, which is an *encoder-only* architecture with a masked language modeling loss function. In masked language

modeling, a certain percentage of tokens from the input sequence are randomly masked, and the model is trained to predict the masked tokens based on the context provided by the rest of the sequence. This approach encourages the model to learn representations that capture informative features of the protein sequences, enabling it to perform well on downstream tasks. ESM2 is trained on an enriched version of Uniref50[41] dataset with over 60M sequences and comes in various sizes and training computational budgets. Notice that the standard Uniref50 dataset clusters sequences with at least 50% sequence identity and comprises approximately 45M sequences. We also investigate ESM-1b[26] trained on the UniRef50 dataset as well as ESM-1v[35] which is trained on the UniRef90[41] dataset. The UniRef90 dataset clusters sequences with at least 90% sequence identity and contains about 138M sequences.

We further delve into the models from the ProtT5[27] family. ProtT5 employs a variant of the Text-To-Text Transfer Transformer (T5) training procedure[21] which utilizes an *encoder-decoder* architecture. ProtT5xl-bfd and ProtT5xxl-bfd models are trained on the BFD100 dataset. The BFD100 dataset is a clustered version of the Big Fantastic Database (BFD)[42,43], where sequences with 100% identity are grouped, effectively removing duplicates. The dataset consolidates protein sequences from various sources and contains approximately 2.1B unique sequences. Subsequently, these models are fine-tuned on the UniRef50 dataset, resulting in the ProtT5xl and ProtT5xxl models. To provide an extensive outlook, we additionally examine ProtBert[27], ProtAlbert[27], ProtXLNet[27], ProteinBert[36] and Tasks Assessing Protein Embeddings (TAPE)[37] models. ProtBert is trained on the UniRef100[41] dataset, which consolidates 216M sequences from the UniProt Knowledgebase (UniProtKB)[44], while ProtBert-bfd and ProteinBert on BFD100 and UniRef90, respectively. ProtAlbert and ProtXLNet utilize the BERT-based architectures ALBERT[45] and XLNet[46], respectively, and were trained on UniRef100. TAPE Bert[37] is a BERT model, while TAPE Babbler[37] is a unidirectional mLSTM model. Both TAPE models are trained on a dataset of approximately 32M protein sequences retrieved from the Pfam[47] database. Finally, we consider SeqVec[38], a model based on the bi-directional LSTM architecture of Embeddings from Language Models (ELMo)[48], trained on the UniRef50 dataset.

Table 1. Overview of the studied PLMs. The table lists key characteristics of the 18 PLMs, including their embedding size, training datasets, architecture details, and the number of parameters.

Model		Embedding size	Training dataset	Number of layers	Model type	Number of parameters
ESM2[29]	**t48**	5120	UniRef50++	48	Transformer	15B
	t36	2560	UniRef50++	36	Transformer	3B
	t33	1280	UniRef50++	33	Transformer	650M
	t12	480	UniRef50++	12	Transformer	35M
	t6	320	UniRef50++	6	Transformer	8M
ESM	**1b**[26]	1280	UniRef50	33	Transformer	650M
	1v[35]	1280	UniRef90	33	Transformer	650M
ProtT5[27]	**xxl**	1024	BFD100/UniRef50	24	Transformer	11B
	xl	1024	BFD100/UniRef50	24	Transformer	3B
	xl-bfd	1024	BFD100	24	Transformer	3B
ProtBert[27]		1024	UniRef100	30	Transformer	420M
ProtBert-bfd[27]		1024	BFD100	30	Transformer	420M
ProtAlbert[27]		4096	UniRef100	12	Transformer	224M
ProtXLNet[27]		1024	UniRef100	30	Transformer	409M
ProteinBert[36]		1562	UniRef90	12	Transformer	16M
TAPE Bert[37]		768	Pfam	12	Transformer	92 M
TAPE Babbler[37]		1900	Pfam	2	mLSTM	18M
SeqVec[38]		1024	UniRef50	2	ELMo	93M

Embedding-Based Transfer Learning

The first transfer learning approach, as shown in Figure 1a), begins by tokenizing the protein sequences using the model's tokenizer. The tokenized sequence is then passed through the pre-trained PLM to generate token-level embeddings that capture both semantic and structural information. To create a fixed-size representation for each sequence, mean pooling is applied across the sequence length. Mean pooling is a straightforward yet effective approach that preserves the essential information

without requiring any additional training. These embeddings, along with their corresponding class labels, are then used to train a classifier to distinguish between antimicrobial and non-antimicrobial peptides (AMP classification).

For the AMP classification task, we utilize shallow classifiers such as Logistic Regression (LogReg), Support Vector Machines (SVMs), and Extreme Gradient Boosting (XGBoost). The incorporation of a diverse set of classifiers enables a thorough evaluation of how different machine learning models perform when leveraging PLM-generated embedding representations. For SVM and XGBoost classifiers, moderate hyper-parameter tuning is performed. The choice of shallow classifiers was primarily motivated by the relatively small size of the datasets in comparison to the embedding dimensions. Furthermore, the linear nature of the aggregated embedding space, empirically observed across various domains such as images and language, suggests that simple and shallow classifiers are capable of effectively capturing the underlying patterns and relationships within the data[19,49].

Efficient Parameter Fine-Tuning

Adjusting the weights of a pre-trained model, rather than training task-specific classifiers harnessing its embeddings, is another transfer learning approach used in deep learning. Given sufficiently large sample size, it typically results in enhanced performance on downstream tasks[32,50]. This method involves the fine-tuning of the pre-trained model's parameters towards the statistics of the new task while retaining its overall learned representations. For large models like PLMs and LLMs, full parameter fine-tuning can be computationally expensive, memory-intensive and may lead to overfitting. To address these issues, researchers have developed Parameter-Efficient Fine-Tuning (PEFT) techniques[51–53]. Among these, Low-Rank Adaptation (LoRA)[51] and its variants[54–56] have emerged as leading approaches due to their efficiency and simplicity. LoRA functions by introducing trainable low-rank matrices into specific layers of the pre-trained model (see also Figure 1b). These matrices serve as updates to the frozen model parameters, significantly reducing the number of trainable parameters during fine-tuning. Consequently, the low-rank nature of the updates, coupled with the preservation of the original model's weights, significantly reduces the risks of catastrophic forgetting[57] or overfitting which are common issues in many fine-tuning methods[58,59].

In this work, we deploy LoRA[51] and its variant Quantized LoRA[54] (QLoRA) for fine-tuning the PLMs to perform AMP classification. QLoRA introduces additional memory-saving innovations without compromising performance, including the 4-bit NormalFloat (NF4) data type for the weights, double quantization to further reduce memory usage, and paged optimizers to handle memory spikes. To extend the models towards the binary classification task (AMP vs non-AMP), a dense layer with tanh activation, dropout for regularization, and an output projection layer are attached to the final layer of the PLM's encoder, serving as the classification head.

Moderate hyperparameter tuning was conducted to determine an optimal set of LoRA parameters, including the scaling factor, which adjusts the contribution of the low-rank updates, the rank, r, determining the size of the trainable matrices, and the dropout rate, used to regularize the low-rank updates. Additionally, typical training parameters such as learning rate, batch size, number of training epochs, and weight decay rate were optimized. Following standard LoRA fine-tuning practices for LLMs, we introduced low rank matrices for the self-attention mechanism in the transformer layers of PLMs. Different combinations of query, key, value, and output projection matrices within the self-attention module were explored to identify the update configuration that demonstrates robust and consistent performance across all datasets under examination.

Datasets Description

This study evaluates the performance of PLMs on AMP classification using seven widely recognized datasets: XUAMP[5], APD3[60], CAMP[61,62], dbAMP[63], DRAMP[64,65], LAMP[66,67], and YADAMP[68]. These datasets vary considerably in size, ranging from as few as 406 sequences in CAMP to 3,072 sequences in XUAMP, underscoring the challenges posed by limited data availability in AMP classification. Each dataset has been carefully curated to balance the number of AMP and non-AMP sequences, with lengths ranging from 11 to 100 residues. Additionally, strict preprocessing ensures no sequences within the same subset share more than 40% sequence identity, measured using global alignment tools. Table 2 presents a detailed summary of these datasets, with visualizations of the sequence embeddings available in the 'Datasets Overview' section in the Supplementary.

Results

Comparisons between PLMs: Scale matters

This section compares the performance of PLMs on several AMP datasets using embedding-based transfer learning and shallow classifiers trained on the mean embedding vector generated by each PLM. Figure 2 showcases the relationship between average accuracy and model size in logarithmic scale for the XUAMP dataset. This evaluation employs three classifiers (LogReg, SVM and XGBoost), with moderate hyperparameter tuning performed for SVM and XGBoost to optimize their performance (see section 'Details on Training and Fine-Tuning' in Supplementary). The average accuracy and standard deviation are calculated by first performing 10-fold cross-validation for each classifier and then averaging the results across the three classifiers. It

Table 2. Description of AMP datasets in terms of size, sequence length, mean/median sequence length, and average sequence-sequence identity.

Dataset	Number of Peptides	Length Range (residues)		Mean/Median Length		Sequence Identity (%)	
		AMP	Non-AMP	AMP	Non-AMP	AMP	Non-AMP
XUAMP	3072	16-100	28-100	62.9/63.0	77.1/79.0	25.4	28.7
DRAMP	2816	16-100	31-100	62.7/62.0	76.5/78.0	25.7	29.0
LAMP	2108	13-100	30-100	58.0/57.0	74.9/76.0	24.0	28.7
dbAMP	1044	17-100	32-100	52.5/48.0	75.5/77.0	23.9	28.8
APD3	988	13-100	31-100	48.1/43.0	75.6/77.0	24.1	28.9
YADAMP	648	11-100	33-100	31.4/32.0	77.1/79.0	25.3	29.0
CAMP	406	11-100	29-100	19.6/20.0	75.8/78.0	24.7	28.7

is evident from Figure 2 that the model size plays a crucial role in improving the accuracy of the classifiers, indicating their elevated capacity to identify intricate patterns in protein sequences. As the PLM size increases, its proficiency in capturing and representing complex features of protein sequences improves, resulting in superior classification accuracy.

Apparently, the improvement scales relatively consistently with the logarithm of the model size. Indeed, the Spearman correlation between log-model size and AMP accuracy is 0.87, suggesting a strong relationship between the two quantities. Moreover, the performance of the ESM2 family of models consistently exceeds the average trend (represented by the red line in Figure 2), indicating that these models are particularly effective and achieve enhanced performance. This favorable behavior can be partially attributed to the high-quality training dataset that plays a significant role in enhancing the performance of the models. Similar trends are observed across all the tested AMP datasets, as shown in Supplementary Figures S4–S9, further reinforcing the consistency of our findings.

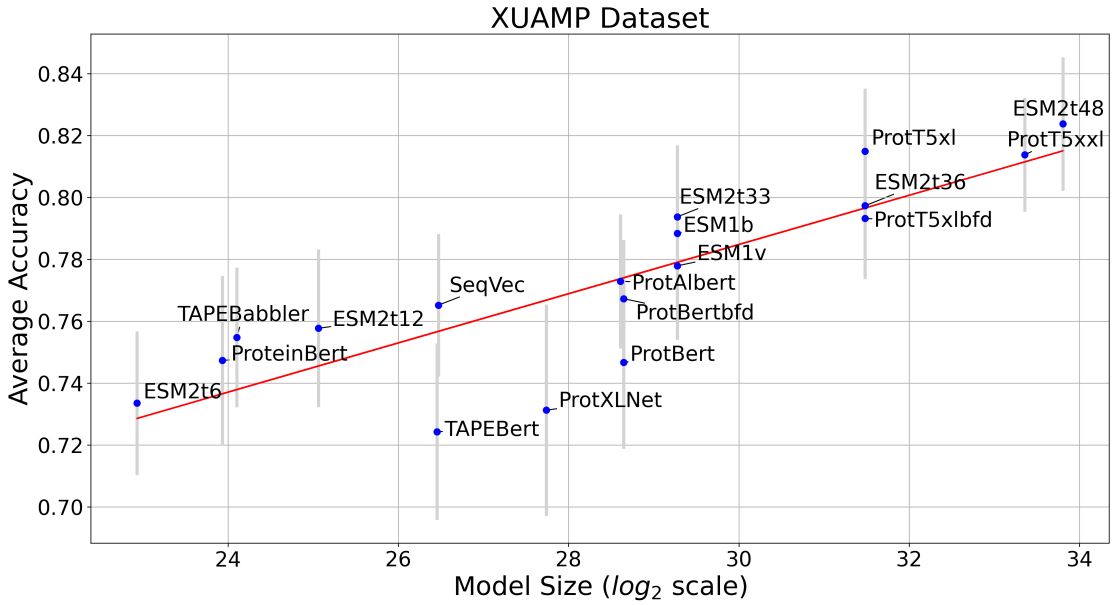

Figure 2. Average AMP accuracy plotted against model size on a logarithmic scale for the XUAMP dataset. Each point represents the mean accuracy, and the vertical grey lines indicate the respective standard deviation. The red line illustrates the linear fit described by the equation $y = 0.008x + 0.546$, which suggests that for every doubling of the model size, there is an associated 0.8% gain in accuracy.

Figure 3 generalizes the above observations via presenting the average accuracy over all AMP datasets using LogReg as a classifier (see Supplementary Figures S10–S11 for the other two classifiers). The ordering of the models in Figure 3 is determined by their size, from the largest PLM to the smallest one. ESMt48 with 15B parameters and ProtT5xl with 3B parameters exhibit the highest accuracy across all datasets. ProtT5xxl (11B) and ESM2t36 (3B) exhibit slightly lower accuracy while certain BERT-based models, such as ProtBert (420M), TAPE Bert (92M), and ProtBert-bfd (240M), deliver even lower accuracy. The fact that ProtT5xl performs better than ProtT5xxl, which is almost 4 times larger, could be at least partially

attributed to the computational budget constraints that were imposed for its training[27], highlighting the significant role of additional training epochs in improving the performance of downstream tasks. Another observation from Figure 3 is that ProtT5xl, which has been fine-tuned with UniRef50, is significantly better from the ProtT5xl-bfd which is trained solely on the BFD dataset. This outcome aligns with the findings from Elnggar et al.[27] on different downstream tasks, suggesting that a well-curated, diverse dataset like UniRef50 can be beneficial for improved performance.

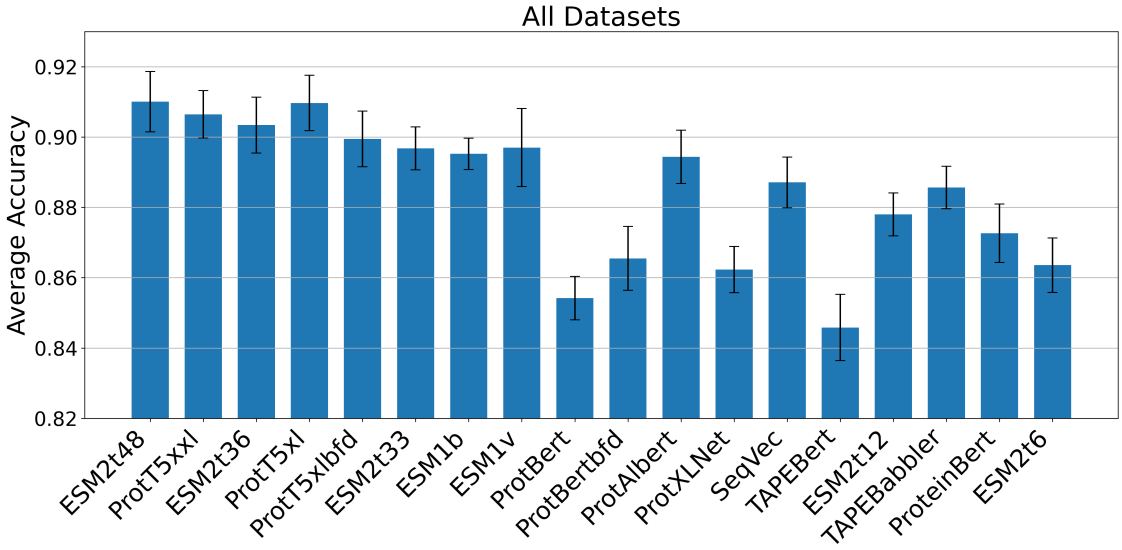

Figure 3. Average accuracy over all tested datasets with standard deviation using LogReg as a classifier. Models are arranged according to their size, from the largest (on the left) to the smallest (on the right).

Comparisons with Existing AMP Classifiers: Superior performance with minimal effort

Employing the same embedding-based transfer learning pipeline as previously, the top-performing PLMs are compared against state-of-the-art (SOTA) approaches in AMP classification, including sAMPpred-GAT[12], amPEPpy[7], AMPfun[8] and AMPEP[10]. To ensure a fair comparison, we utilize the curated datasets from Yan *et al.*[12] and follow the same evaluation protocol. The methodology employed for dataset construction deviates from conventional evaluation practices such as K-fold cross validation because special care is taken to avoid sequences in the training, validation and test sets with large sequence identity. Consequently, these dataset splits also incorporate aspects of generalization error and they actually correspond to a mixture of both classification and generalization error (we refer to Supplementary Figure S3 where those train and test sets are visualized using the UMAP algorithm[69]).

Table 3 reports the accuracy of the four best-performing PLMs, compared to the results from Yan *et al.*[12], which are derived by stand-alone deep learning models trained solely on AMP datasets. In six out of seven datasets, the embeddings from pretrained PLMs, when paired with shallow classifiers, outperformed the current SOTA stand-alone models. This emphasizes the benefits of embedding-based transfer learning approaches, suggesting that pretrained models can achieve superior performance in AMP classification with minimal effort. Moreover, the SVM classifier consistently produced higher accuracy values than other classifiers, although this difference was less pronounced for the ESM2 models compared to the ProtT5 models. Another key observation is that the largest model in each family consistently demonstrated the highest accuracy levels, indicating that these deeper models may reduce generalization error more effectively by providing more informative embeddings. Overall, we conclude that ESM2t48 and ProtT5xxl deliver the best performance in terms of AMP accuracy, a result that is further confirmed by other performance metrics reported in Supplementary Tables S9–S12.

Parameter Fine-Tuning Outperforms Embedding-Based Transfer Learning

Parameter fine-tuning, which allows the weights of the PLM to be updated, represents a more sophisticated transfer learning approach. In this section, we assess the accuracy of fine-tuned models on the AMP classification task and compare their performance to the embedding-based transfer learning approach. Specifically, we focus on LoRA and QLoRA, both of which are parameter-efficient fine-tuning techniques. Models from both ESM2 and ProtT5 families were selected for parameter fine-tuning due to their consistently high rankings in all previous experiments while the curated datasets and splits from Yan *et al.*[12] were utilized for the evaluation.

Table 3. Accuracy values across different AMP datasets for the train/test splits provided in the work of Yan *et al.*12. The last column corresponds to the average accuracy across all AMP datasets.

Model		Classifier	Dataset							Average
			XUAMP	DRAMP	LAMP	dbAMP	APD3	YADAMP	CAMP	
ESM2	t48	LogReg	**0.735**	0.766	0.884	**0.910**	0.922	0.968	0.978	0.880
		SVM	0.727	0.774	**0.888**	0.900	0.925	0.969	0.980	0.880
		XGBoost	0.727	0.764	0.853	0.885	0.898	0.961	**0.988**	0.868
	t36	LogReg	0.723	0.758	0.864	0.882	0.906	0.966	0.973	0.867
		SVM	0.725	0.761	0.868	0.884	0.909	0.958	0.973	0.868
		XGBoost	0.726	0.754	0.851	0.892	0.898	0.954	0.966	0.863
ProtT5	xxl	LogReg	0.723	0.752	0.871	0.899	0.912	0.957	0.970	0.869
		SVM	0.734	**0.779**	0.883	0.900	0.924	0.971	0.978	0.881
		XGBoost	0.724	0.762	0.873	0.908	0.917	**0.972**	0.975	0.876
	xl	LogReg	0.712	0.731	0.843	0.884	0.897	0.906	0.906	0.840
		SVM	**0.735**	0.762	0.864	0.895	0.907	0.944	0.936	0.863
		XGBoost	0.697	0.721	0.805	0.866	0.863	0.917	0.879	0.821
SOTA		sAMPpred-GAT	0.715	0.760	0.840	0.888	0.896	0.955	0.956	0.858
		amPEPpy	0.679	0.734	0.765	0.889	**0.939**	0.915	0.948	0.838
		AMPEP	0.661	0.712	0.755	0.766	0.936	0.969	0.973	0.825
		ADAM-HMM	0.684	0.736	0.872	0.886	0.886	0.927	0.869	0.837
		AMPFUN	0.702	0.743	0.810	0.892	0.905	0.950	0.933	0.848

Experimentation with different combinations of target LoRA modules revealed that updating the query, key, and value matrices of the self-attention mechanism together produced the best results for the ESM2 family of models. For the ProtT5 models, we also updated the output weight matrix, which follows the self-attention mechanism. We set the rank $r = 1$ for the weight update matrices, as this configuration not only yielded among the highest performance across all datasets but also introduces the fewest trainable parameters. We associate the low rank value to the limited number of samples available for fine-tuning, which is relatively small compared to the size of the PLM. Additional details on the hyperparameter values and tuning can be found in the 'Details on Training and Fine-Tuning' section in the Supplementary.

Figure 4 displays the receiver operating characteristic (ROC) curves for representative fine-tuned ESM2 and ProtT5 models on the DRAMP (left) and XUAMP (right) datasets. The results correspond to the average performance over five runs, each initialized with a different random seed. For comparison, the ROC curve of the SVM classifier, trained on the embeddings from the largest ESM2 model (ESM2t48), is also included. For the DRAMP dataset, the highest AUC value is achieved by fine-tuning ESM2t33 with LoRA, followed by ProtT5xl also fine-tuned with LoRA. Interestingly, the largest ESM2 model, fine-tuned with QLoRA, did not yield the highest AUC value, which may be attributed to the larger number of introduced trainable parameters leading to poor fitting, given the relatively small size of the training dataset. Furthermore, AUC values suggest that fine-tuning ESM2t12 offers classification performance comparable to the embedding-based transfer learning approach with the larger ESM2t48 model. ROC curve plots for the remaining datasets are available in the Supplementary Figure S12.

Table 4 reports a detailed comparison of average AUC values (five runs with different random seeds) for different PLMs fine-tuned with LoRA or QLoRA as well as stand-alone SOTA models. Standard deviations, being of the order of the third decimal, are omitted for clarity and are available in Supplementary Table S13. The results clearly demonstrate that fine-tuning consistently outperforms the embedding-based transfer learning approach in the AMP classification task, given the same model and dataset. The improvements observed in the DRAMP dataset, and to some extent in the XUAMP dataset, are particularly noteworthy. Furthermore, both LoRA and QLoRA show comparable performance in terms of AUC across all datasets, indicating that parameter quantization does not deteriorate performance. Apart from the AUC, the accuracy values of the fine-tuned PLMs have consistently improved, further demonstrating the enhanced predictive capabilities of these models. In particular, the highest accuracy achieved on the XUAMP dataset is 75.3% with the ESM2t36 model when fine-tuned using LoRA, while the highest accuracy on the DRAMP dataset reaches 79.4% with the same ESM2t36 model, but fine-tuned using QLoRA. For a detailed presentation of evaluation metrics, we refer to the Supplementary Tables S13–S18.

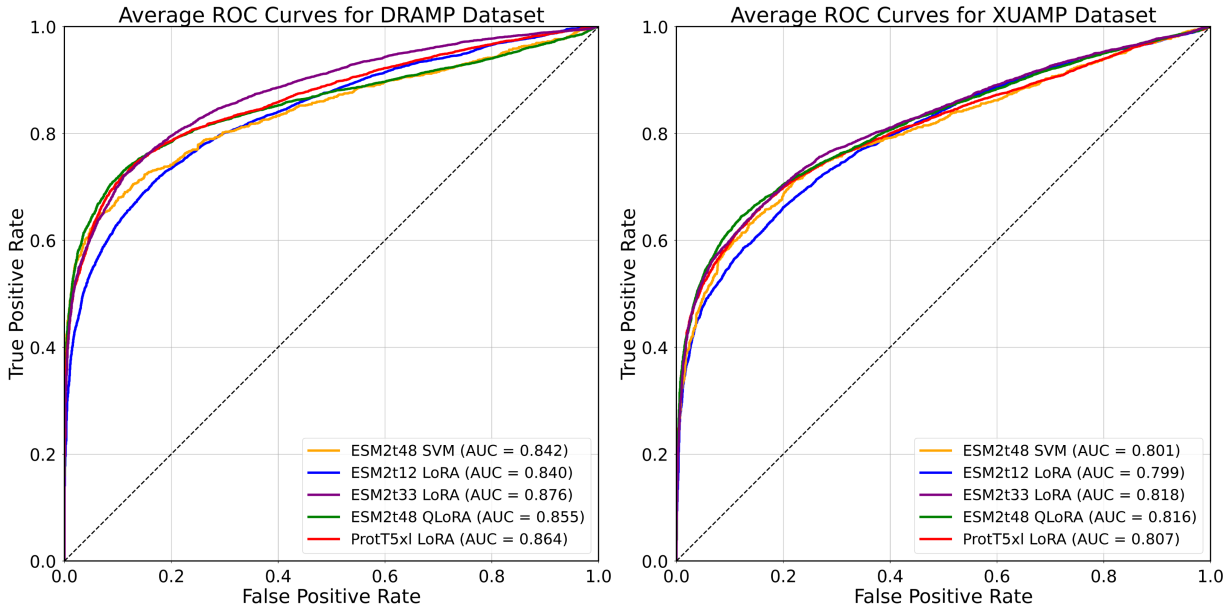

Figure 4. Average ROC curves for models evaluated on the DRAMP (left) and XUAMP (right) datasets. We compare the SVM classifier trained on the ESMt48 model's embeddings (orange) with ESM2t12 (blue), ESM2t33 (purple), ESM2t48 (green) as well as ProtT5xl (red) fine-tuned with LoRA or QLoRA. The best-performing model from the embedding-based approach achieves performance comparable to the smallest fine-tuned ESM2 model. In contrast, all other PLMs fine-tuned with (Q)LoRA exhibit superior performance.

Table 4. Average AUC values for various fine-tuned PLMs across different datasets.

PLM		Transfer Learning	Dataset						
			XUAMP	DRAMP	LAMP	dbAMP	APD3	YADAMP	CAMP
ESM2	t48	SVM	0.801	0.842	0.962	0.972	0.977	0.996	**1.000**
	t12	LoRA	0.799	0.840	0.951	0.968	0.970	0.997	**1.000**
	t33	LoRA	**0.818**	0.877	0.960	0.976	0.977	0.997	**1.000**
	t33	QLoRA	0.817	**0.882**	0.961	0.977	0.974	0.997	**1.000**
	t36	LoRA	0.810	0.861	**0.965**	0.978	0.978	0.997	**1.000**
	t36	QLoRA	0.806	0.861	**0.965**	**0.980**	0.979	0.997	**1.000**
	t48	QLoRA	0.816	0.856	0.964	0.976	0.978	**0.998**	**1.000**
ProtT5	xxl	SVM	0.805	0.857	0.956	0.971	0.977	0.997	0.999
	xl	LoRA	0.807	0.864	0.962	0.972	0.980	0.997	0.999
	xxl	LoRA	0.802	0.852	0.963	0.975	0.978	0.996	**1.000**
SOTA		sAMPpred-GAT	0.777	0.827	0.917	0.952	0.954	0.994	**1.000**
		amPEPpy	0.742	0.759	0.855	0.940	0.972	0.968	0.978
		AMPEP	0.727	0.773	0.818	0.933	**0.983**	0.992	0.994
		ADAM-HMM	0.684	0.736	0.872	0.886	0.886	0.927	0.869
		AMPFUN	0.735	0.810	0.852	0.930	0.972	0.997	**1.000**

Discussion

The unprecedented number of protein sequences available in public databases, coupled with the exponential growth of computational resources, presents a unique opportunity to deepen our knowledge of proteins, their structure and function through self-supervised learning. This work explores the transfer learning capabilities of various PLMs on the AMP classification task. The evaluation is conducted using real data from the literature and we aim to understand which characteristics of PLMs crucially contribute to efficiently capturing discriminative information between antimicrobial and non-antimicrobial peptide sequences. Our findings emphasize the *importance of scale* in achieving higher accuracy, with larger PLMs consistently outperforming their smaller counterparts. The results also highlight that PLMs can achieve SOTA results with minimal effort

via transfer learning making AI-driven protein engineering more reliable and accessible to a broader audience. Indeed, the embedding-based transfer learning pipeline employed in this study demonstrates that highly effective AMP classification can be achieved without relying on complex architectures or extensive fine-tuning. This simplicity ensures that the embedding-based transfer learning framework can be readily adopted by researchers with minimal machine learning expertise. Additionally, fine-tuning methods such as LoRA and QLoRA further enhance PLM performance by adapting the models to specific datasets efficiently, while mitigating risks such as overfitting and catastrophic forgetting. This observation aligns well with recent studies on fine-tuning approaches of PLMs which have demonstrated notable performance gains across a range of classification or regression tasks[32]. Effectively, these parameter-efficient techniques balance computational demands and performance improvements, making them valuable tools for maximizing the utility of PLMs. We also investigated potential performance differences between the LoRA and QLoRA approaches via repeated experiments which were conducted across multiple datasets using the same training configurations. No statistically significant differences were observed in performance between LoRA and QLoRA as it is evident from Table 4 and Supplementary Tables S13–S18. In contrast, there are significant differences in computational requirements with QLoRA significantly reducing GPU memory usage, requiring up to 10 times less memory compared to LoRA without affecting the inference time.

Although this study achieves SOTA performance in AMP classification, it likely represents a conservative estimate of PLMs' potential. For instance, the use of mean pooling for embedding aggregation, while effective, may not fully exploit the rich information within the embeddings. Future research will explore more advanced aggregation techniques, such as task-specific attention mechanisms or bottleneck autoencoders, to enhance performance further. Moreover, the potential for improvement is supported by ongoing advancements in PLM architecture and training methodologies, which continue to produce models with greater representational power. For example, recent developments like xTrimoPGLM[30] and ESM3[31], featuring PLMs with 100 billion parameters, have shown superior performance compared to the largest ESM2 model across various downstream tasks. While the weights for these models are not yet publicly available, their success underscores the rapid progress in the field and the immense potential for further innovation to advance AMP classification and related applications.

Data Availability

The datasets used in this study are publicly-available at: https://github.com/HongWuL/sAMPpred-GAT/tree/main/datasets.

References

1. Mahlapuu, M., Håkansson, J., Ringstad, L. & Björn, C. Antimicrobial peptides: an emerging category of therapeutic agents. *Front. cellular infection microbiology* **6**, 235805 (2016).
2. Cruz, J., Ortiz, C., Guzman, F., Fernández-Lafuente, R. & Torres, R. Antimicrobial peptides: promising compounds against pathogenic microorganisms. *Curr. medicinal chemistry* **21**, 2299–2321 (2014).
3. Zasloff, M. Antimicrobial peptides of multicellular organisms. *nature* **415**, 389–395 (2002).
4. Huan, Y., Kong, Q., Mou, H. & Yi, H. Antimicrobial peptides: classification, design, application and research progress in multiple fields. *Front. microbiology* **11**, 2559 (2020).
5. Xu, J. *et al.* Comprehensive assessment of machine learning-based methods for predicting antimicrobial peptides. *Briefings bioinformatics* **22**, bbab083 (2021).
6. Lin, T.-T. *et al.* AI4AMP: An antimicrobial peptide predictor using physicochemical property-based encoding method and deep learning. *Msystems* **6**, e00299–21 (2021).
7. Lawrence, T. J. *et al.* amPEPpy 1.0: A portable and accurate antimicrobial peptide prediction tool. *Bioinformatics* **37**, 2058–2060 (2021).
8. Chung, C.-R., Kuo, T.-R., Wu, L.-C., Lee, T.-Y. & Horng, J.-T. Characterization and identification of antimicrobial peptides with different functional activities. *Briefings bioinformatics* **21**, 1098–1114 (2020).
9. Wei, L., Zhou, C., Su, R. & Zou, Q. PEPred-Suite: Improved and robust prediction of therapeutic peptides using adaptive feature representation learning. *Bioinformatics* **35**, 4272–4280 (2019).
10. Bhadra, P., Yan, J., Li, J., Fong, S. & Siu, S. W. AmPEP: Sequence-based prediction of antimicrobial peptides using distribution patterns of amino acid properties and random forest. *Sci. reports* **8**, 1697 (2018).
11. Lee, H.-T., Lee, C.-C., Yang, J.-R., Lai, J. Z. & Chang, K. Y. A large-scale structural classification of antimicrobial peptides. *BioMed research international* **2015**, 475062 (2015).
12. Yan, K., Lv, H., Guo, Y., Peng, W. & Liu, B. sAMPpred-GAT: Prediction of antimicrobial peptide by graph attention network and predicted peptide structure. *Bioinformatics* **39**, btac715 (2023).

13. Lee, H., Lee, S., Lee, I. & Nam, H. AMP-BERT: Prediction of antimicrobial peptide function based on a BERT model. *Protein Sci.* **32**, e4529 (2023).

14. Zervou, M. A., Doutsi, E., Pantazis, Y. & Tsakalides, P. Multitask classification of antimicrobial peptides for simultaneous assessment of antimicrobial property and structural fold. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1836–1840 (IEEE, 2024).

15. Pang, Y., Yao, L., Xu, J., Wang, Z. & Lee, T.-Y. Integrating transformer and imbalanced multi-label learning to identify antimicrobial peptides and their functional activities. *Bioinformatics* **38**, 5368–5374 (2022).

16. Veltri, D., Kamath, U. & Shehu, A. Deep learning improves antimicrobial peptide recognition. *Bioinformatics* **34**, 2740–2747 (2018).

17. Bongirwar, V. & Mokhade, A. Different methods, techniques and their limitations in protein structure prediction: A review. *Prog. Biophys. Mol. Biol.* **173**, 72–82 (2022).

18. Sun, R., Wu, L., Lin, H., Huang, Y. & Li, S. Z. Enhancing protein predictive models via proteins data augmentation: a benchmark and new directions. *arXiv preprint arXiv:2403.00875* (2024).

19. Brown, T. B. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165* (2020).

20. Radford, A. *et al.* Language models are unsupervised multitask learners. *OpenAI blog* **1**, 9 (2019).

21. Raffel, C. *et al.* Exploring the limits of transfer learning with a unified text-to-text transformer. *J. machine learning research* **21**, 1–67 (2020).

22. Wei, J. *et al.* Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652* (2021).

23. Touvron, H. *et al.* Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

24. Min, B. *et al.* Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Comput. Surv.* **56**, 1–40 (2023).

25. Achiam, J. *et al.* GPT-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

26. Rives, A. *et al.* Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proc. Natl. Acad. Sci.* **118**, e2016239118 (2021).

27. Elnaggar, A. *et al.* ProtTrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis Mach. Intell.* **44**, 7112–7127 (2021).

28. Shin, J.-E. *et al.* Protein design and variant prediction using autoregressive generative models. *Nat. communications* **12**, 2403 (2021).

29. Lin, Z. *et al.* Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science* **379**, 1123–1130 (2023).

30. Chen, B. *et al.* xTrimoPGLM: Unified 100B-scale pre-trained transformer for deciphering the language of protein. *arXiv preprint arXiv:2401.06199* (2024).

31. Hayes, T. *et al.* Simulating 500 million years of evolution with a language model. *bioRxiv* 2024–07 (2024).

32. Schmirler, R., Heinzinger, M. & Rost, B. Fine-tuning protein language models boosts predictions across diverse tasks. *Nat. Commun.* **15**, 7407 (2024).

33. Thumuluri, V., Almagro Armenteros, J. J., Johansen, A. R., Nielsen, H. & Winther, O. DeepLoc 2.0: Multi-label subcellular localization prediction using protein language models. *Nucleic Acids Res.* **50**, W228–W234 (2022).

34. Watson, J. L. *et al.* De novo design of protein structure and function with RFdiffusion. *Nature* **620**, 1089–1100 (2023).

35. Meier, J. *et al.* Language models enable zero-shot prediction of the effects of mutations on protein function. *Adv. neural information processing systems* **34**, 29287–29303 (2021).

36. 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).

37. Rao, R. *et al.* Evaluating protein transfer learning with TAPE. *Adv. neural information processing systems* **32** (2019).

38. Heinzinger, M. *et al.* Modeling aspects of the language of life through transfer-learning protein sequences. *BMC bioinformatics* **20**, 1–17 (2019).

39. Vaswani, A. *et al.* Attention is all you need. *Adv. neural information processing systems* **30** (2017).

40. 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).

41. Suzek, B. E., Huang, H., McGarvey, P., Mazumder, R. & Wu, C. H. UniRef: Comprehensive and non-redundant UniProt reference clusters. *Bioinformatics* **23**, 1282–1288 (2007).

42. Steinegger, M., Mirdita, M. & Söding, J. Protein-level assembly increases protein sequence recovery from metagenomic samples manyfold. *Nat. methods* **16**, 603–606 (2019).

43. Steinegger, M. & Söding, J. Clustering huge protein sequence sets in linear time. *Nat. communications* **9**, 2542 (2018).

44. Consortium, U. UniProt: A worldwide hub of protein knowledge. *Nucleic acids research* **47**, D506–D515 (2019).

45. Lan, Z. *et al.* ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942* (2019).

46. Yang, Z. XLNet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237* (2019).

47. El-Gebali, S. *et al.* The Pfam protein families database in 2019. *Nucleic acids research* **47**, D427–D432 (2019).

48. Peters, M. E. *et al.* Deep contextualized word representations. In Walker, M., Ji, H. & Stent, A. (eds.) *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2227–2237 (Association for Computational Linguistics, New Orleans, Louisiana, 2018).

49. Tian, Y., Wang, Y., Krishnan, D., Tenenbaum, J. B. & Isola, P. Rethinking few-shot image classification: a good embedding is all you need? In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, 266–282 (Springer, 2020).

50. Sledzieski, S. *et al.* Democratizing protein language models with parameter-efficient fine-tuning. *Proc. Natl. Acad. Sci.* **121**, e2405840121 (2024).

51. Hu, E. J. *et al.* LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685* (2021).

52. Liu, H. *et al.* Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Adv. Neural Inf. Process. Syst.* **35**, 1950–1965 (2022).

53. He, H., Cai, J., Zhang, J., Tao, D. & Zhuang, B. Sensitivity-aware visual parameter-efficient fine-tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 11825–11835 (2023).

54. Dettmers, T., Pagnoni, A., Holtzman, A. & Zettlemoyer, L. QLoRA: Efficient finetuning of quantized LLMs. *Adv. Neural Inf. Process. Syst.* **36** (2024).

55. Liu, S.-Y. *et al.* DoRA: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353* (2024).

56. Zhang, Q. *et al.* AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512* (2023).

57. Biderman, D. *et al.* LoRA learns less and forgets less. *arXiv preprint arXiv:2405.09673* (2024).

58. Kumar, A., Raghunathan, A., Jones, R., Ma, T. & Liang, P. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054* (2022).

59. Zhai, Y. *et al.* Investigating the catastrophic forgetting in multimodal large language model fine-tuning. In *Conference on Parsimony and Learning*, 202–227 (PMLR, 2024).

60. Wang, G., Li, X. & Wang, Z. APD3: The antimicrobial peptide database as a tool for research and education. *Nucleic acids research* **44**, D1087–D1093 (2016).

61. Thomas, S., Karnik, S., Barai, R. S., Jayaraman, V. K. & Idicula-Thomas, S. CAMP: A useful resource for research on antimicrobial peptides. *Nucleic acids research* **38**, D774–D780 (2010).

62. Waghu, F. H., Barai, R. S., Gurung, P. & Idicula-Thomas, S. CAMPR3: A database on sequences, structures and signatures of antimicrobial peptides. *Nucleic acids research* **44**, D1094–D1097 (2016).

63. Jhong, J.-H. *et al.* dbAMP: An integrated resource for exploring antimicrobial peptides with functional activities and physicochemical properties on transcriptome and proteome data. *Nucleic acids research* **47**, D285–D297 (2019).

64. Fan, L. *et al.* DRAMP: A comprehensive data repository of antimicrobial peptides. *Sci. reports* **6**, 24482 (2016).

65. Kang, X. *et al.* Dramp 2.0, an updated data repository of antimicrobial peptides. *Sci. data* **6**, 148 (2019).

66. Zhao, X., Wu, H., Lu, H., Li, G. & Huang, Q. LAMP: A database linking antimicrobial peptides. *PloS one* **8**, e66557 (2013).

67. Ye, G. *et al.* LAMP2: A major update of the database linking antimicrobial peptides. *Database* **2020**, baaa061 (2020).

68. Piotto, S. P., Sessa, L., Concilio, S. & Iannelli, P. YADAMP: Yet another database of antimicrobial peptides. *Int. journal antimicrobial agents* **39**, 346–351 (2012).

69. McInnes, L., Healy, J. & Melville, J. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018).

Acknowledgements

This work is partially funded by Theodore Papazoglou FORTH Synergy Grants under the project “FUSNING: Biophysical tools FUSed via integrative computational approaches to decode protein foldING”. We also acknowledge Project I.C.Humans, no 91, funded by the Hellenic Foundation for Research and Innovation (HFRI) under the "1st Call for HFRI Research Projects to support Faculty members and Researchers and the procurement of high-cost research equipment" for covering our needs on computational resources.

Author contributions statement

Y.P. and M.A.Z. conceived the study, all authors assisted in the design of the study and E.G. implemented the pipelines, conducted the computational experiments and performed the data analysis and visualization. E.G. and Y.P. analyzed the results. All authors contributed to the writing of the manuscript. All authors reviewed and approved the final version of the manuscript.

Corresponding Author

Correspondence to Yannis Pantazis (email: pantazis@iacm.forth.gr).

Competing Interests

The authors declare no competing interests.