# Parameter-Efficient Adaptation of a Pretrained Language Model via Soft Prompt Tuning Enables Hit-Enriched Conditional Cell Line-Specific Generation of CPPs

**Adelina Latypova** [1] **Evgeniy V. Nam** [1] **Timofei Ryko** [1] **Ekaterina V. Timofeeva** [2] **Violetta Isakova** [3]
**Nikita Serov** [1]

## Abstract

Cell-penetrating peptides (CPPs) offer a promising route for intracellular delivery, yet their design remains constrained by scarce, heterogeneous, and weakly standardized experimental data. Existing computational work has largely focused on supervised modeling, while the more challenging problem of generating functional, cell line–specific CPPs under limited supervision remains underexplored. Here, we introduce a conditional generative framework for cell line–aware CPP design by fine-tuning a pretrained protein language model (pLM) using parameter-efficient adaptation strategies. To support reliable post-generation selection, we developed independent predictive model for CPP activity, including a calibrated classifier achieving F1 = 0.847 and precision = 0.952. We further enforced strict dataset separation and active sampling to mitigate information leakage and improve robustness in low-data regimes. Among the evaluated conditioning strategies, soft-prompt tuning provided the best balance between functional enrichment, diversity, and sequence-level novelty, while maintaining favorable diversity and similarity constraints. Furthermore, external validation with PMIPred was conducted, which showed strong enrichment toward membrane-active peptides. Overall, our results demonstrate that conditional generation of CPPs is feasible even under stringent data limitations, and provide a scalable blueprint for moving beyond predictive peptide modeling toward controllable, function-oriented molecular design.

[1]Center for AI in Chemistry, ITMO University, Saint Petersburg, Russia [2]Department of Bioengineering and Bioinformatics, Institute of Pharmacy, Sechenov First Moscow State Medical University, Moscow, Russia [3]Department of Biological and Medical Physics, Moscow Institute of Physics and Technology, Moscow, Russia. Correspondence to: Nikita Serov <serov@pish.itmo.ru>.

Accepted at the AI for Science workshop (ICML 2026).

## 1. Introduction

Cell-penetrating peptides (CPPs) are short amino acid sequences, typically 5–30 residues in length, capable of traversing the cell membrane and delivering diverse therapeutic cargoes, including nucleic acids, proteins, and small molecules (Martin & Rice, 2007; Ruseska & Zimmer, 2020; Guo et al., 2016; Du et al., 2025). They have attracted considerable interest in oncology, gene therapy, and overcoming the blood–brain barrier (Bottens & Yamada, 2022; Borrelli et al., 2018; Stalmans et al., 2015; Ghorai et al., 2023; Du et al., 2025). Traditional CPP discovery relies on high-throughput screening or rational design based on physicochemical rules (Carney et al., 2017; Oikawa et al., 2018; Marks et al., 2011; Numata et al., 2018; Schwaar et al., 2019; Liu et al., 2024), but experimental approaches are costly and labor-intensive, while rational design remains unreliable due to the complex, nonlinear nature of membrane penetration (Borrelli et al., 2018).

Computational methods offer a scalable alternative, yet remain constrained by limited experimental data (Shi et al., 2024). Existing discriminative models (Holton et al., 2013; Gautam et al., 2013; 2015; Manavalan et al., 2018; Manavalan & Patra, 2022; Preto et al., 2024) enable screening of candidate sequences but cannot generate novel designs. Recent generative approaches (Tran et al., 2021; Fukunaga et al., 2024) address this gap by learning sequence distributions, yet none account for cell line specificity—a critical factor, as CPP uptake efficiency varies substantially across cell types due to differences in membrane composition and receptor profiles (Mueller et al., 2008; Patel et al., 2019; Numata et al., 2018; Tyagi et al., 2001; Milletti, 2012).

We present a conditional generative framework that incorporates cell line identity as an explicit control variable, enabling targeted CPP design under data-constrained settings. The approach combines parameter-efficient fine-tuning (PEFT) of the pretrained protein language model ProtGPT2 with independent classifier-based filtering to enrich for functional candidates (Figure 3). Our key contributions include: (1) a conditional generation strategy achieving substantial CPP enrichment while maintaining sequence diversity, (2) development and rigorous validation

of predictive models under strict dataset separation, and (3) demonstration of cell line–specific sequence distributions with strong membrane-binding enrichment via external validation using PMIpred (van Hilten et al., 2024).

## 2. Methods

### 2.1. Data Processing

For generative purposes, a comprehensive dataset of experimentally validated CPPs and confirmed non-CPPs was collected. The primary data source was the POSEIDON database (Preto et al., 2024). We additionally retrieved data from published ML studies on CPP prediction: CellPPD (Gautam et al., 2013; 2015), AiCPP (Park et al., 2023), KELM-CPPpred (Pandey et al., 2018), TargetCPP (Arif et al., 2020), an SVM-based predictive model (Sanders et al., 2011), and a predictive model for phosphorodiamidate morpholino oligomer (PMO)-peptide conjugates (Wolfe et al., 2018). Part of the sequences was manually curated from review articles (Bechara & Sagan, 2013; Milletti, 2012; Heitz et al., 2009; Järver & Langel, 2006; Hansen et al., 2008; Gori et al., 2023) and experimental studies (Hällbrink et al., 2005; Numata et al., 2018; Oikawa et al., 2018; Lee et al., 2012; Karagiannis et al., 2013; Gautam et al., 2015). After initial aggregation, the combined dataset comprised 5,211 CPP entries and 3,793 non-CPP entries prior to deduplication and quality control procedures.

To prepare sequences for ML model training, we implemented a standardization pipeline to extract chemical modification information, validate AA composition, and generate canonical sequence representations. Raw peptide sequences were classified into 13 distinct categories using regular expression pattern matching. Categories encompassed standard L-amino acid sequences, stereoisomeric variants containing D-amino acids, sequences with chemical modifications (N-terminal acetylation or acylation, biotinylation, C-terminal amidation, cyclization, phosphorylation), affinity-tagged peptides (hexahistidine), and sequences with complex or non-standard notation. For each sequence, modification flags were extracted through rule-based pattern matching against the raw sequence strings. Eight modification features were encoded into the binary vector. Additionally, D-amino acid positions were recorded for sequences using lowercase notation to indicate D-stereochemistry.

Upon detection of modification patterns, the corresponding notation symbols (prefixes, suffixes, parenthetical annotations) were removed from the raw sequence string to yield the underlying peptide backbone sequence. This cleaning step enabled downstream feature extraction from canonical AA representations while preserving modification information as separate binary features. Cleaned sequences were validated to contain only the 20 standard proteinogenic AAs. Sequences containing non-canonical residues, ambiguous position codes, or residual non-alphabetic characters after cleaning failed validation. Sequences failing validation were flagged and excluded from feature extraction. This standardization process yielded two complementary outputs for each valid sequence: (1) a canonical AA string, and (2) a set of binary modification flags preserving information about chemical derivatization.

Finally, deduplication was performed separately for CPP and non-CPP datasets using a sequence-centric approach. After removing exact duplicates based on sequence identity, 1,781 unique CPP and 1,321 unique non-CPP sequences remained. Case-sensitive sequence variants (e.g., "RKKRRQRRR" vs. "rkkrrqrrr"), representing L- and D-amino acid forms modifications, were retained as distinct entities. Sequence-based comparison revealed 180 sequences annotated as CPP in some sources and non-CPP in others. To handle these ambiguous cases, we implemented a three-class labeling system for downstream analysis: sequences consistently classified as CPP ("yes", n=1,601), consistently classified as non-CPP ("no", n=1,141), and conflicting sequences ("both", n=180). For binary classification and generation tasks requiring unambiguous labels, the "both" category was assigned to non-CPP.

### 2.2. Feature Engineering

Four complementary feature sets spanning two distinct abstraction levels were engineered: (1) chemical structure-based representations emphasizing molecular topology and physicochemical properties, and (2) sequence-based representations leveraging evolutionary and biochemical information encoded in AA order.

#### 2.2.1. Chemical Structure-Based Representations

All standardized peptide sequences were converted to canonical Simplified Molecular Input Line Entry System (SMILES) notation (Weininger, 1988). Standard unmodified sequences were processed using RDKit's (Landrum et al., 2024) MolFromSequence function. Sequences containing chemical modifications or D-amino acids were handled through a custom conversion pipeline: chemical modification notations (e.g., Ac-, (biotin)-, Stearyl-) were substituted with corresponding SMILES fragments, both L-amino acids (uppercase) and D-amino acids (lowercase) were mapped to stereochemically correct representations. The resulting SMILES strings were canonicalized using the RDKit CanonSmiles function with chirality preservation. Of 2,922 sequences, 2,915 were successfully converted to valid SMILES. The seven failed conversions were assigned zero-valued fingerprint vectors to maintain consistent dimensionality across the dataset.

**Morgan Fingerprints** These are equivalent to Extended Connectivity Fingerprints (Rogers & Hahn, 2010) and encode molecular structure through iterative identification of circular atom neighborhoods. Fingerprints were generated using RDKit's GetMorganFingerprintAsBitVect function with the following parameters: radius - 2 bonds, vector length - 1024 bits, standard atom-type hashing, enabled stereochemistry.

**Interpretable Feature Subset** Interpretable feature subset was constructed as a minimal feature set (19 features) comprising sequence length, extracted during sequence standardization binary modification flags and BioPython global descriptors. We used features characterizing the properties considered to influence the efficiency of peptide penetration into cells: the balance between cationicity (total positive charge), hydrophobicity, and helicity of peptides. (Jiang et al., 2008; Epand et al., 2003; Béven et al., 2003; Song et al., 2005).

#### 2.2.2. Sequence-Based Representations

**ProtBERT Embeddings** We used the ProtBERT-BFD model (Elnaggar et al., 2021) . The sequences preprocessed according to ProtBERT input requirements were tokenized and passed through the frozen ProtBERT model. The final hidden states from the last transformer layer were extracted and aggregated via mean pooling across all AA positions (excluding padding tokens) to produce a single 1024-dimensional vector per sequence.

**BLOMAP** Each AA residue was encoded as an 11-dimensional numerical vector. The first five components were derived from multidimensional scaling of the BLOSUM62 substitution matrix (Maetschke et al., 2005). The remaining six components captured explicit physicochemical properties, including molecular weight, isoelectric point, normalized hydrophobicity, polarity and other properties. Sequences were padded to the length of the longest sequence in the dataset using a special padding token "X", encoded as a zero vector. The resulting per-residue vectors were then concatenated to produce a single flat feature vector. Ambiguous AA codes (B, Z, J) were mapped to averaged properties of their constituent possibilities. Selenocysteine (U) and pyrrolysine (O) were mapped to zero vectors.

### 2.3. Classification Task

#### 2.3.1. Classifier Baseline

To distinguish between CPPs and non-CPPs, we employed a set of supervised learning algorithms. The models selected were Logistic Regression (LR), Support Vector Machines (SVM), Decision Trees (DT), Random Forest (RF), and Gradient Boosting (CatBoost). Implementation was performed using scikit-learn (ver. 1.7.0) (Pedregosa et al., 2011) and CatBoost (ver. 1.2.8) (Prokhorenkova et al., 2019). Model performance was evaluated using standard binary classification metrics (Accuracy, Precision, Recall, F1-score).

**Classifier Baseline's Train Datasets** All five algorithms were trained on an Interpretable Feature Subset. CatBoost classifiers were additionally trained on Morgan Fingerprints, BLOMAP embeddings and ProtBERT embeddings. For LR and SVM models, features were standardized using z-score normalization. Tree-based models (DT, RF, CatBoost) were trained on raw feature values.

The dataset exhibited approximate class balance (55:45 ratio), excluding the need for resampling strategies. The dataset was partitioned into training (80%) and test (20%) sets using stratified random sampling to preserve class proportions (seed = 42). For the precision-optimized Random Forest, an additional 20% validation split was reserved from the training data for threshold calibration.

**Hyperparameter Optimization of Classifier** Hyperparameter optimization was performed using Optuna (50 trials, TPE sampler) with 5-fold stratified cross-validation, maximizing average precision score. The search space included: number of estimators (200–2000), maximum depth (2–50), minimum samples per split (2–20), minimum samples per leaf (1–20), maximum features (sqrt, log2, none), bootstrap sampling, and class weight balancing strategies. The optimal model was subsequently calibrated using isotonic regression with 5-fold cross-validation to improve probability estimates. A classification threshold was selected on a held-out validation set to achieve the target precision of $\geq$0.90, and final performance was evaluated on the test set at this fixed threshold.

#### 2.3.2. Active Learning-Based Classification

Since the main aim of the trained model is unbiased evaluation of generated sequences, we trained a classifier and a generator on non-overlapping datasets. To achieve this we employed active sampling with a minimal informative subset for the classifier while preserving the majority of data for the generator. The architecture of the classifier is based on the Random Forest model that was selected and optimized in the previous step. The initial dataset of 1,601 CPP and 1,321 non-CPP sequences was randomly split into training pool (60%), validation (20%), and test (20%) sets using stratified sampling (random seed = 42). The validation set was used for threshold calibration, and the test set was reserved for final evaluation. Next, we implemented a hybrid active sampling strategy with three components: (1) Core-set sampling (70%): K-means clustering (k = 10) was applied separately to CPP and non-CPP sequences using standardized feature space. For each cluster, the sample closest to the

centroid was selected as a prototype, ensuring coverage of the feature space. (2) Hard negative sampling (20%): Negative samples (non-CPP) with the highest out-of-fold (OOF) predicted probabilities were prioritized. These represent samples near the decision boundary that the model tends to misclassify as positive. OOF probabilities were obtained via 5-fold stratified cross-validation using a Random Forest classifier (n_estimators = 400). (3) Margin sampling (10%): Samples with predicted probabilities closest to 0.5 were selected to refine the decision boundary. Within each sampling category, final selection was performed using farthest-first traversal (k-center greedy algorithm) to maximize diversity. Distances were calculated as Euclidean distances between StandardScaler-normalized features. Budget sizes ranging from 100 to 1,000 were evaluated. For each one, the classification threshold was selected to maximize F1-score while maintaining precision $\geq 0.90$ on the validation set.

## 2.4. CPP Generation

### 2.4.1. Base Model

The ProtGPT2 model (Ferruz et al., 2022) was used as the foundational model for CPP generation. Model weights were initialized from the pre-trained ProtGPT2 checkpoint available on HuggingFace.

### 2.4.2. Conditional Generation and Training Objective

Three post-training strategies were evaluated: full fine-tuning, soft-prompt tuning, and prefix tuning. The first served as an unconditional baseline, while the others incorporated conditional generation to enable cell-line-specific peptide design.

For conditional generation, a 13-dimensional binary condition vector **c** was constructed for each training example. The first element indicates CPP class membership (1 for CPP, 0 for non-CPP), and the remaining 12 elements encode the target cell line using one-hot encoding. For instance, a CPP active against the third cell line is represented as [1, 0, 0, 1, 0, . . . , 0]. During training, only CPP-positive examples were used, so the first element was set to 1 for all samples. This condition vector was passed as input to a multilayer perceptron (MLP) that projected it into the model's representation space; the architecture of this MLP differed between soft-prompt and prefix tuning (see **sections 2.4.4 and 2.4.5**).

For both PEFT methods, the pre-trained ProtGPT2 parameters $\phi$ were frozen, and only the MLP generator parameters $\theta$ were updated to minimize the negative log-likelihood. The loss functions for Prefix Tuning (PT) and Soft-Prompt Tuning (SPT) are defined in Eq.(1) and Eq.(2), respectively:

$$L_{PT} = -\sum_{k=1}^{D}\sum_{i=1}^{n_k} \log p_{\phi,\theta}(w_i^k \mid w_{<i}^k, KV_\theta^{(1:L)}(c_k)) \quad (1)$$

$$L_{SPT} = -\sum_{k=1}^{D}\sum_{i=1}^{n_k} \log p_{\phi,\theta}(w_i^k \mid [P_\theta(c_k); w_{<i}^k]) \quad (2)$$

where $D$ is the number of training sequences,$n_k$ is the length of the $k$-th sequence in tokens, $w_i^k$ is the $i$-th token, $w_{<i}^k$ denotes all preceding tokens, $c_k$ is the condition vector, $P_\theta$ is the soft-prompt generator, and $KV_\theta^{(1:L)}$ is the key-value prefix generator across all $L$ layers.

### 2.4.3. Fine-Tuning

As an unconditional baseline, the entire set of model parameters $\phi$ was updated without the condition vector. Fine-tuning was performed using the run_clm.py script from the HuggingFace Transformers library (Wolf et al., 2020), following the procedure described on the ProtGPT2 HuggingFace page.

### 2.4.4. Soft-Prompt Tuning

For soft-prompt tuning (Lester et al., 2021), a prompt generator MLP was trained while the base model weights remained frozen. The MLP takes the 13-dimensional condition vector as input and produces 20 continuous embedding vectors, each of dimension 1,280, matching the hidden dimensionality of ProtGPT2. These embeddings are prepended to the input token embeddings, so the frozen model processes the combined sequence as if the prompt vectors were part of the input. The MLP consists of a single hidden layer with 512 neurons, ReLU activation, and dropout (p = 0.1), followed by Layer Normalization on the output embeddings. The generator was trained for 3 epochs using the AdamW optimizer with a learning rate of $5 \times 10^{-4}$ and a batch size of 8. The maximum sequence length was set to 256 tokens. No validation split was used; PEFT variants were selected using post-generation distributional diagnostics rather than validation loss.

### 2.4.5. Prefix Tuning

For prefix tuning (Li & Liang, 2021), a key-value (KV) prefix generator MLP was trained while the base model weights remained frozen. The MLP takes the 13-dimensional condition vector as input and produces key-value prefix pairs that are prepended to the attention key and value matrices at every transformer layer. The MLP output is reshaped into prefixes of length 8 for each of the 20 attention heads across all 36 layers. The MLP consists of a single hidden

layer with 1,024 neurons and ReLU activation. Unlike the original prefix tuning method, no reparameterization was applied, as the primary objective was to compare tuning strategies rather than to optimize computational efficiency. The generator was trained for 3 epochs using the AdamW optimizer with a learning rate of $5 \times 10^{-4}$ and a batch size of 4. The maximum sequence length was set to 256 tokens. No validation split was used; PEFT variants were selected using post-generation distributional diagnostics rather than validation loss.

#### 2.4.6. Sequence Generation

Sequences were generated using top-k sampling (k = 50) with a temperature value of 1.0. Repetition penalty values ranging from 1.0 to 2.0 were tested to mitigate amino acid repetition in generated sequences. The maximum number of generated tokens was set to 50.

All models were trained and evaluated on an NVIDIA RTX A6000 GPU using PyTorch 2.6.0 and HuggingFace Transformers 4.52.0.dev0.

#### 2.4.7. Sequence Validation

To evaluate the performance of the proposed framework, the generated peptide sequences were validated using the PMIPred web server (van Hilten et al., 2024). PMIPred was employed as a proxy model to assess the membrane-binding propensity of the generated sequences. The validation pipeline consisted of the following steps: (i) peptide sequence generation; (ii) length-based filtering to satisfy PMIPred input constraints (minimum and maximum sequence lengths of 7 and 24 residues, respectively); and (iii) preliminary filtering using a classifier. This procedure was iteratively repeated until a total of 1,200 sequences passed the minimum selection criteria. All retained sequences were subsequently evaluated using the PMIPred web server. The predicted membrane-binding free energy for a vesicle of 50 nm radius ($\Delta F_{sm}(R = 50)$) was used as the primary criterion for classification, following the thresholds defined by the PMIPred authors. Sequences classified by PMIPred as binders were operationally defined as CPP candidates, under the assumption that stable membrane association is a necessary (though not sufficient) condition for cellular penetration. All remaining sequences were classified as non-CPPs. In addition to binders and non-binders, PMIPred defines an intermediate class of sensor peptides, characterized by moderate membrane-binding free energies. These peptides are predicted to sense membrane curvature without forming stable membrane-bound states and were therefore not considered CPP candidates in our analysis.

## 3. Results

### 3.1. Cell-Line Specific CPP Generator

Due to the limited available data and the need to train an independent classifier, the existing dataset was divided into non-overlapping parts (see **Section 2.3.2**). After preparation, 559 CPP sequences remained for generator fine-tuning (see Table 1). The pretrained ProtGPT2 model was adapted using this data, incorporating cell line information as a binary conditional vector (see **Section 2.4.2**). Cell lines represented by fewer than ten sequences were grouped into an "others" category. To address class imbalance, random oversampling was applied to match the largest class (Others, 165 peptides), resulting in 12 cell line classes.

For each tuning variant, a total 500 sequences were generated (temperature = 1; top_k = 50).

*Table 1.* Comparative analysis of tuning strategies.

| Model | Mean loss | Perplex. | Similar. | MMD | % cpp |
|---|---|---|---|---|---|
| Base | 5.824 | 373.46 | 19.90 | 0.393 | 3.42 |
| Fine-tune | 5.769 | 356.23 | 19.81 | 0.402 | 2.22 |
| Soft-prompt | 5.684 | 442.65 | **9.75** | 0.241 | 46.0 |
| Prefix | **2.405** | **93.53** | 17.27 | **0.141** | **47.2** |

The obtained results did not allow us to identify a more suitable strategy, necessitating a detailed analysis. For this purpose, the following metrics were used: the percentage of uniqueness, AA composition, median internal pairwise Levenshtein distance, normalized Shannon entropy for AA composition, Jensen-Shannon divergence by length (relative to real CPPs), Jensen-Shannon divergence by AA (relative to real CPPs), coverage $\leq 0.2$ (relative to real CPPs), and median Levenshtein distance to the nearest neighbor from the set of real CPPs. Coverage is the proportion of sequences that are at a normalized Levenshtein distance $< 0.2$ from the nearest natural sequence. All listed metrics for the tested hyperparameter configurations are summarized in Figure 4 and Table 3 in the supplementary materials. Following hyperparameter optimization, the best configurations were: Soft-prompt: temperature = 1, repetition penalty = 1.1; Prefix: temperature = 1, repetition penalty = 1.2

Based on the distributional analysis of the generated sequences we selected Soft-prompt as the final model (see details in **Section 4.1**).

### 3.2. CPP Generation Framework

The developed model, despite promising metrics, still generates a relatively low percentage of CPPs. This complicates its direct use. To mitigate this, we used an independent classifier as an initial filter. It was trained on a non-overlapping

data subset obtained using active sampling (see **Section 2.3.2**).

A systematic screening across five classification algorithms paired with multiple molecular representations (see **Section 2.3.1**) was performed, the top-10 configurations are listed in Table 2 (complete screening results are provided in Table 4 in the supplementary materials).

*Table 2.* Comparison of classifier performance.

| Model | Feature set | Acc. | Prec. | Recall | F1 |
|---|---|---|---|---|---|
| CatBoost | Morgan fingerprints | 0.836 | 0.847 | 0.852 | **0.850** |
| Random Forest | Interpretable Feature Subset | 0.840 | 0.839 | 0.854 | 0.846 |
| CatBoost | Interpretable Feature Subset | 0.835 | 0.817 | **0.875** | 0.845 |
| CatBoost | Blomap | **0.842** | **0.863** | 0.813 | 0.837 |
| Random Forest | Blomap | 0.840 | 0.884 | 0.783 | 0.830 |
| SVM | ProtBERT | 0.824 | 0.808 | 0.849 | 0.828 |
| SVM | Morgan fingerprints | 0.809 | 0.814 | 0.840 | 0.827 |
| Random Forest | Morgan fingerprints | 0.812 | 0.831 | 0.821 | 0.826 |
| CatBoost | ProtBERT | 0.811 | 0.805 | 0.820 | 0.812 |
| Logistic Regression | Morgan fingerprints | 0.788 | 0.798 | 0.818 | 0.807 |

Since the classifier serves as a filter for generated candidates prior to experimental testing, minimizing false positives is of critical importance. Accordingly, the best-performing Random Forest model was further optimized to prioritize precision without substantial degradation of overall discriminative performance. Isotonic probability calibration was applied and the decision threshold was adjusted to 0.825, yielding a test-set precision of **0.952** with recall reduced to **0.636** (F1 = 0.762). This trade-off was considered acceptable given the disproportionate cost of false positives. Full details of the selected hyperparameter configuration and calibration procedure are provided in Table 5.

### 3.3. External Validation

Following internal validation, generated sequences were further evaluated using PMIPred (see **Section 2.4.7**), which is a physics-based predictive model for estimating peptide-membrane binding free energies.

During validation, 191 of 1,200 sequences (15.9%) could not be evaluated due to repeated HTTP 502 server errors and were excluded. Reported class proportions are based solely on successfully evaluated sequences.

Among these, 51.8% were classified as binders (CPP candidates), 20.1% as sensor peptides, and 28.1% as non-binders. In addition to the aggregated analysis on all cell lines, cell line–specific analyzes were performed, results are shown in Figure 1.

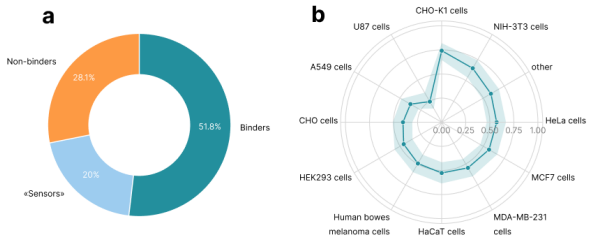


*Figure 1.* Cell line-specific analysis of generated sequences. A - The proportion of sequences classified by PMIPred as binders, sensor peptides, and non-binders. B - The proportion of sequences classified as binders for each target cell line.

## 4. Discussion

### 4.1. CPP Generator Metrics

Both PEFT approaches dramatically improved CPP generation capacity (see Table 1). Soft-prompt tuning increased CPP content to 46.0%, accompanied by substantially reduced pairwise similarity among generated peptides (9.75%) and improved MMD (0.241), indicating effective generation of novel CPP-like sequences. Prefix tuning achieved comparable CPP yield (47.2%) with the lowest training loss (2.41), but exhibited higher pairwise similarity among generated peptides (17.27%) and superior MMD (0.141), suggesting closer adherence to the learned CPP distribution at the cost of reduced exploration. To determine the optimal strategy, we conducted a distributional analysis of the generated sequences (see **Section 3.1**). Length and AA distributions were compared with real CPPs (Figures 2**a** and 2**c**). Prefix tuning demonstrated closer alignment with the native CPP distribution across multiple metrics (Figures 2**b** and 2**d**). The model achieved lower divergence in amino acid composition ($\sqrt{\text{JSD}} = 0.164$ vs. 0.243 for Soft-prompt), but higher divergence in length distribution ($\sqrt{\text{JSD}} = 0.411$ vs. 0.258). This was further reflected in superior MMD scores (0.134 vs. 0.247) and tighter coverage of the real CPP space (17.65% sequences with coverage $\leq 0.2$ vs. 4.33%). The median minimal Levenshtein distance to nearest real CPPs was lower for Prefix-generated sequences (0.38 vs. 0.50), indicating closer resemblance to training data.

In contrast, Soft-prompt excelled in generating diverse and novel sequences. The model produced 99.86% unique sequences compared to 67.03% from Prefix tuning. Soft-prompt also exhibited greater internal diversity, achieving a higher median pairwise Levenshtein distance (0.895 vs. 0.824). Critically, Soft-prompt demonstrated substantially higher novelty relative to the training set (99.93% vs. 93.91%), suggesting greater capacity for generating

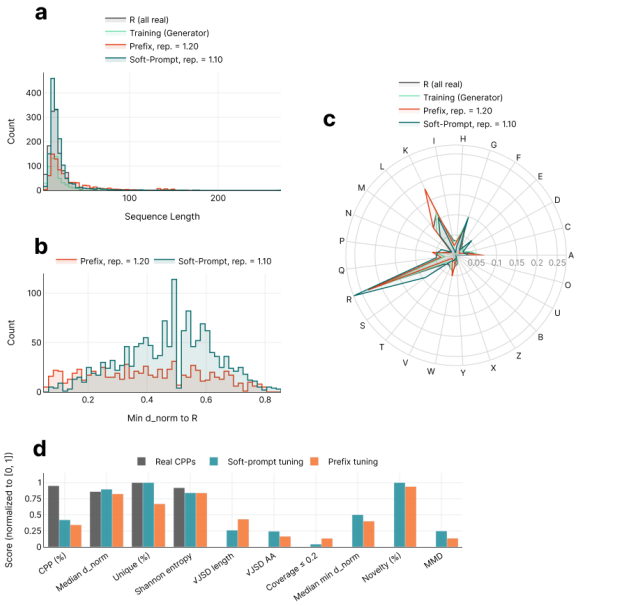


*Figure 2.* Distributional analysis results. A, B, C - comparison of the length, minimal distance to the nearest neighbour among real CPPs, and AA ratio respectively between generated subsamples, D - multi-metric evaluation of generated sequence sets relative to real CPPs, comprising CPP yield, diversity, distributional similarity, and novelty indicators.

unexplored chemical space while maintaining CPP characteristics.

Both models exhibited enrichment in cationic residues (Lysine and Arginine) relative to real CPPs (normalized Shannon entropy: 0.841 for Soft-prompt vs. 0.839 for Prefix), consistent with preferential learning of the dominant cationic CPP subtype. This bias likely reflects the compositional skew in training data, potentially underrepresenting amphipathic or hydrophobic CPP variants.

Despite Prefix tuning's closer adherence to real CPP distributions, we selected Soft-prompt as the final model based on its diversity, novelty, and generation stability, which are critical for exploring beyond known CPP space in subsequent peptide design applications.

### 4.2. Classifier-Based Filtering

As part of the CPP design framework the classifier was needed to filter generated sequences. To ensure unbiased estimation, the classifier and generator training sets should not overlap. At the same time, it was important to preserve the maximum amount of data for generator tuning and achieve optimal classifier performance under limited data constraints. For this purpose we employed a two stage strategy: a classifier was first trained on the full labeled dataset to establish an upper performance bound, after which a hybrid active sampling procedure was applied to identify a minimal yet representative training subset that approaches this bound. The first stage involved screening various models and descriptor sets, as well as tuning the optimal variant to minimize false positive results (see **Section 2.3.1**). This determined the upper bound on the model's possible performance given the full data.

With the upper bound established, a hybrid active sampling strategy was employed to identify the smallest training subset that preserves classification quality. Single-criterion sampling approaches (e.g., diversity-only or uncertainty-only) risk either underrepresenting the decision boundary or inadequately covering the feature space (Settles, 2009); a composite strategy combining complementary selection principles was therefore adopted. The sampling budget was allocated as follows: 70% to core-set (centroid-based) sampling for broad, representative coverage of the feature space; 20% to hard negative mining to reinforce the decision boundary in the most error-prone region; and 10% to margin-based sampling for additional boundary refinement (see **Section 2.3.2**). This allocation reflects the design priority of ensuring representative coverage as the foundation, supplemented by targeted refinement near ambiguous regions.

A budget of 500 samples (204 CPP and 296 non-CPP sequences) achieved a precision of 0.946 (F1 of 0.651, recall of 0.496). A comparison of the metrics with the baseline model is shown in Figure 5b. This classifier was adopted as the high-precision filtering model for evaluating generated peptide sequences in the CPP design framework.

To quantify the separation between the two training subsets (classifier vs generator), we analyzed their inter-set Levenshtein distances. Internal sequence diversity, measured by median pairwise normalized Levenshtein distance, was slightly lower in the classifier set (0.85 vs. 0.88 for generator). A minimum normalized distance of 0.0278 confirmed the absence of identical sequences, though it also indicates the presence of highly similar pairs differing by as little as a single amino acid. We quantified this boundary region and found that 24.9% of sequences in the generator set and 26.6% in the classifier set had a close neighbor (normalized Levenshtein distance $\leq 0.2$) in the opposing set.

However, the majority of sequences were well-separated: the mean minimal Levenshtein distance from generator to classifier was 0.403, with the reverse direction yielding 0.423. Critically, the 5th percentile of cross-dataset distances was 0.667, demonstrating that $\geq 95\%$ of sequences differ by more than two-thirds of their length between datasets.

Normalized Shannon entropy, computed over per-sequence amino acid frequency distributions, was comparable between datasets (0.904 for classifier vs. 0.918 for generator),

indicating equivalent sequence-level compositional complexity. Global amino acid profiles were largely concordant between training sets Figure 5e). The generator set exhibited slight enrichment in cationic residues (Lysine and Arginine) relative to the classifier set, consistent with preservation of canonical CPP characteristics. No major compositional shifts were observed that would compromise the representativeness of either dataset.

Analysis of physicochemical properties from the interpretable feature subset revealed statistically significant but practically negligible differences: while several features showed $p < 0.001$, all effect sizes were minimal ($\epsilon < 0.01$), confirming distributional similarity in biologically relevant characteristics.

### 4.3. CPP Generation Framework Results

After developing the generator and classifier, we evaluated the CPP design framework using PMIPred (Figure 1). Overall, the framework produced a substantial fraction of CPP candidates, while notable variation in binder fraction was observed across cell lines. The proportion of sequences classified as binders ranged from 25% (U87 cells) to 74% (CHO-K1 cells), with most cell lines falling between 40% and 70%.

A $\chi^2$ test of independence confirmed that these differences are unlikely to arise from sampling variability alone ($p << 0.001$), indicating a statistically significant association between cell line and binder classification. Because PMIPred operates independently of cellular context and relies solely on physicochemical properties, this heterogeneity reflects differences in the distributions of generated sequences conditioned on each cell line.

Analysis of average net charge across cell lines revealed substantial variation; however, no clear monotonic relationship between charge and binder fraction was observed. For example, some cell lines with relatively low average charge exhibited moderate binder fraction (e.g. MDA-MB-231 cells), whereas others with higher charge did not consistently yield more binders (e.g. HaCaT cells). This suggests that membrane-binding propensity cannot be explained by charge alone, but instead emerges from a combination of sequence-level properties.

To quantify this effect, we fitted a logistic regression model using five physicochemical descriptors (length, net charge, GRAVY index, aromatic fraction, and aliphatic index). The model achieved a pseudo-$R^2$ of approximately 0.56, indicating that these features explain the majority of variance in PMIPred classification. Net charge, hydrophobicity (GRAVY), aromatic fraction, and aliphatic index were all positively associated with binding probability, while peptide length showed a negative association.

Extending the model to include cell line as an additional predictor resulted in only marginal improvement (pseudo-$R^2 \approx 0.57$), with most cell line coefficients not statistically significant. This indicates that the observed differences in binder fraction across cell lines can largely be explained by shifts in physicochemical feature distributions, rather than by additional cell line–specific effects.

Taken together, these results suggest that conditional generation produces distinct regions of sequence space characterized by different physicochemical profiles. These shifts, in turn, largely determine membrane-binding propensity as estimated by PMIPred. The findings therefore support the interpretation that the generative framework successfully controls global sequence properties, while remaining grounded in physically interpretable determinants of CPP activity.

## 5. Conclusion

In this study, we present a generative framework for the design of CPPs conditioned on specific cell lines. By combining PEFT of a pretrained language model with active sampling strategy, we successfully trained a generator in a low-data regime, preserving both sequence diversity and key physicochemical characteristics of known CPPs. An independent classification predictive model was integrated into the pipeline to evaluate generated sequences, ensuring high precision and biologically interpretable predictions.

Our analysis demonstrates that soft-prompt tuning provides an optimal balance between CPP content, diversity, and similarity to training sequences, while the generator produces peptides that largely recapitulate the length distributions and AA composition of real CPPs, particularly the enrichment in positively charged residues characteristic of membrane-active peptides. Importantly, conditional generation resulted in distinct physicochemical distributions across target cell lines, with statistically significant differences in predicted membrane-binding propensity, as assessed by PMIPred.

Taken together, these findings establish that the proposed framework can generate diverse, functional CPPs tailored to individual cellular contexts. The approach provides a practical and interpretable tool for rational peptide design, demonstrating the potential of combining generative modeling with active learning and predictive filtering to explore sequence-function relationships in peptide therapeutics.

## Software and Data

https://github.com/GenerativeMolMachines/CPPGeneration

## Acknowledgements

Authors thank ITMO University Research Projects in AI Initiative (RPAII) (project #640100) for infrastructure support.

## Limitations

First, the generative model relies on a limited (559 sequences) and heterogeneous training dataset. Although mitigated by parameter-efficient fine-tuning and active sampling, data scarcity combined with oversampling strategies may bias the model toward overrepresented structural motifs, restricting the exploration of rare sequence variants (e.g., hydrophobic CPPs). Consequently, the observed cell line–specific differences in generated sequences primarily reflect learned physicochemical associations rather than mechanistic biological pathways. Since the conditioning vector encodes cell lines categorically, this heterogeneity may partially capture underlying dataset biases rather than genuine biological specificity.

Furthermore, the current methodological pipeline does not incorporate explicit filtering for critical drug-like properties. While the framework enriches for membrane-active physicochemical traits, it lacks integrated ADMET predictors for proteolytic stability, toxicity, or immunogenicity, which are essential for prioritizing therapeutically viable candidates.

Finally, all evaluations were performed strictly *in silico*. For instance, external validation using PMIPred estimates thermodynamic membrane binding; however, while stable membrane association is a prerequisite for CPP activity, it does not guarantee actual cellular internalization. Therefore, the predicted membrane activity and cell line specificity remain computational proxies. Rigorous experimental validation, including head-to-head in vitro cellular assays, is required to confirm the true biological activity and context-dependent efficacy of the generated peptides.

## Impact Statement

This work aims to advance data-driven approaches for peptide design, with a particular focus on generating CPPs. Potential positive impacts include accelerating the development of peptide-based delivery systems and enabling more efficient exploration of sequence space in drug discovery.

At the same time, the generative nature of the proposed framework raises general considerations common to computational design of bioactive molecules. In particular, membrane-active peptides could, in principle, be misused in unintended contexts, for example as carriers for undesired molecular cargo.

Importantly, the results should not be interpreted as evidence of true cell line specificity at the biological level, but rather as differences in physicochemical properties of generated sequences. Experimental validation would be required before any practical application.

Overall, we believe the primary impact of this work is methodological, contributing to the development of interpretable and controllable generative models for peptide design, while remaining subject to standard limitations and considerations of computational biology approaches.

## Contributions

N.S. conceived the study. A.L. and E.N. developed the methodology. E.N., R.T., E.V.T., V.I., and N.S. contributed to dataset collection. R.T. developed the classification model, performed data preprocessing, and conducted feature engineering. A.L. implemented the generative model, active sampling strategy, internal filtering pipeline, and performed the experiments. E.N. conducted external validation. A.L. and E.N. wrote the original draft of the manuscript. E.N. and N.S. contributed to result interpretation and manuscript revision. All authors approved the final manuscript.

## References

Arif, M., Ahmad, S., Ali, F., Fang, G., Li, M., and Yu, D.-J. TargetCPP: accurate prediction of cell-penetrating peptides from optimized multi-scale features using gradient boost decision tree. *Journal of Computer-Aided Molecular Design*, 34(8):841–856, August 2020. ISSN 0920-654X, 1573-4951. doi: 10.1007/s10822-020-00307-z. URL http://link.springer.com/10.1007/s10822-020-00307-z.

Bechara, C. and Sagan, S. Cell-penetrating peptides: 20 years later, where do we stand? *FEBS Letters*, 587(12), 2013. ISSN 1873-3468. doi: 10.1016/j.febslet.2013.04.031. URL https://onlinelibrary.wiley.com/doi/abs/10.1016/j.febslet.2013.04.031.

Borrelli, A., Tornesello, A., Tornesello, M., and Buonaguro, F. Cell Penetrating Peptides as Molecular Carriers for Anti-Cancer Agents. *Molecules*, 23(2):295, January 2018. ISSN 1420-3049. doi: 10.3390/molecules23020295. URL https://www.mdpi.com/1420-3049/23/2/295.

Bottens, R. A. and Yamada, T. Cell-Penetrating Peptides (CPPs) as Therapeutic and Diagnostic Agents for Cancer. *Cancers*, 14(22):5546, November 2022. ISSN 2072-6694. doi: 10.3390/cancers14225546. URL https://www.mdpi.com/2072-6694/14/22/5546.

Béven, L., Castano, S., Dufourcq, J., Wieslander, , and Wróblewski, H. The antibiotic activity of cationic

linear amphipathic peptides: lessons from the action of leucine/lysine copolymers on bacteria of the class *Mollicutes*. *European Journal of Biochemistry*, 270 (10):2207–2217, May 2003. ISSN 0014-2956, 1432-1033. doi: 10.1046/j.1432-1033.2003.03587.x. URL https://febs.onlinelibrary.wiley.com/doi/10.1046/j.1432-1033.2003.03587.x.

Carney, R. P., Thillier, Y., Kiss, Z., Sahabi, A., Heleno Campos, J. C., Knudson, A., Liu, R., Olivos, D., Saunders, M., Tian, L., and Lam, K. S. Combinatorial Library Screening with Liposomes for Discovery of Membrane Active Peptides. *ACS Combinatorial Science*, 19(5):299–307, May 2017. ISSN 2156-8952, 2156-8944. doi: 10.1021/acscombsci.6b00182. URL https://pubs.acs.org/doi/10.1021/acscombsci.6b00182.

Du, J.-J., Zhang, R.-Y., Jiang, S., Xiao, S., Liu, Y., Niu, Y., Zhao, W.-X., Wang, D., and Ma, X. Applications of cell penetrating peptide-based drug delivery system in immunotherapy. *Frontiers in Immunology*, 16:1540192, January 2025. ISSN 1664-3224. doi: 10.3389/fimmu.2025.1540192. URL https://www.frontiersin.org/articles/10.3389/fimmu.2025.1540192/full.

Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., Bhowmik, D., and Rost, B. ProtTrans: Towards Cracking the Language of Life's Code Through Self-Supervised Learning, May 2021. URL https://www.biorxiv.org/content/10.1101/2020.07.12.199554v3. Pages: 2020.07.12.199554 Section: New Results.

Epand, R. F., Lehrer, R. I., Waring, A., Wang, W., Maget-Dana, R., Lelièvre, D., and Epand, R. M. Direct comparison of membrane interactions of model peptides composed of only Leu and Lys residues. *Peptide Science*, 71(1):2–16, January 2003. ISSN 0006-3525. doi: 10.1002/bip.10372. URL https://onlinelibrary.wiley.com/doi/10.1002/bip.10372.

Ferruz, N., Schmidt, S., and Höcker, B. ProtGPT2 is a deep unsupervised language model for protein design. *Nature Communications*, 13(1):4348, July 2022. ISSN 2041-1723. doi: 10.1038/s41467-022-32007-7. URL https://www.nature.com/articles/s41467-022-32007-7.

Fukunaga, I., Matsukiyo, Y., Kaitoh, K., and Yamanishi, Y. Automatic generation of functional peptides with desired bioactivity and membrane permeability using Bayesian optimization. *Molecular Informatics*, 43(4), 2024. ISSN 1868-1751. doi: 10.1002/minf.202300148. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/minf.202300148.

Gautam, A., Chaudhary, K., Kumar, R., Sharma, A., Kapoor, P., Tyagi, A., Raghava, G. P. S., and Open source drug discovery consortium. In silico approaches for designing highly effective cell penetrating peptides. *Journal of Translational Medicine*, 11 (1):74, March 2013. ISSN 1479-5876. doi: 10.1186/1479-5876-11-74. URL https://doi.org/10.1186/1479-5876-11-74.

Gautam, A., Sharma, M., Vir, P., Chaudhary, K., Kapoor, P., Kumar, R., Nath, S. K., and Raghava, G. P. Identification and characterization of novel protein-derived arginine-rich cell-penetrating peptides. *European Journal of Pharmaceutics and Biopharmaceutics*, 89:93–106, January 2015. ISSN 09396411. doi: 10.1016/j.ejpb.2014.11.020. URL https://linkinghub.elsevier.com/retrieve/pii/S0939641114003476.

Ghorai, S. M., Deep, A., Magoo, D., Gupta, C., and Gupta, N. Cell-Penetrating and Targeted Peptides Delivery Systems as Potential Pharmaceutical Carriers for Enhanced Delivery across the Blood–Brain Barrier (BBB). *Pharmaceutics*, 15(7):1999, July 2023. ISSN 1999-4923. doi: 10.3390/pharmaceutics15071999. URL https://www.mdpi.com/1999-4923/15/7/1999.

Gori, A., Lodigiani, G., Colombarolli, S. G., Bergamaschi, G., and Vitali, A. Cell Penetrating Peptides: Classification, Mechanisms, Methods of Study, and Applications. *ChemMedChem*, 18(17), 2023. ISSN 1860-7187. doi: 10.1002/cmdc.202300236. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cmdc.202300236.

Guo, Z., Peng, H., Kang, J., and Sun, D. Cell-penetrating peptides: Possible transduction mechanisms and therapeutic applications. *Biomedical Reports*, 4(5):528–534, May 2016. ISSN 2049-9434, 2049-9442. doi: 10.3892/br.2016.639. URL https://www.spandidos-publications.com/10.3892/br.2016.639.

Hansen, M., Kilk, K., and Langel, Predicting cell-penetrating peptides. *Advanced Drug Delivery Reviews*, 60(4):572–579, March 2008. ISSN 0169-409X. doi: 10.1016/j.addr.2007.09.003. URL https://www.sciencedirect.com/science/article/pii/S0169409X07002918.

Heitz, F., Morris, M. C., and Divita, G. Twenty years of cell-penetrating peptides: from molecular mechanisms to therapeutics. *British Journal of Pharmacology*, 157(2), 2009. ISSN 1476-5381. doi: 10.1111/j.1476-5381.2009.00057.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1476-5381.2009.00057.x.

Holton, T. A., Pollastri, G., Shields, D. C., and Mooney, C. CPPpred: prediction of cell penetrating peptides. *Bioinformatics*, 29(23):3094–3096, December 2013. ISSN 1367-4811, 1367-4803. doi: 10.1093/bioinformatics/btt518. URL https://academic.oup.com/bioinformatics/article/29/23/3094/246449.

Hällbrink, M., Kilk, K., Elmquist, A., Lundberg, P., Lindgren, M., Jiang, Y., Pooga, M., Soomets, U., and Langel, Prediction of Cell-Penetrating Peptides. *International Journal of Peptide Research and Therapeutics*, 11(4):249–259, December 2005. ISSN 1573-3149, 1573-3904. doi: 10.1007/s10989-005-9393-1. URL https://link.springer.com/10.1007/s10989-005-9393-1.

Jiang, Z., Vasil, A. I., Hale, J. D., Hancock, R. E. W., Vasil, M. L., and Hodges, R. S. Effects of net charge and the number of positively charged residues on the biological activity of amphipathic $\alpha$-helical cationic antimicrobial peptides. *Peptide Science*, 90(3):369–383, January 2008. ISSN 0006-3525. doi: 10.1002/bip.20911. URL https://onlinelibrary.wiley.com/doi/10.1002/bip.20911.

Järver, P. and Langel, Cell-penetrating peptides—A brief introduction. *Biochimica et Biophysica Acta (BBA) - Biomembranes*, 1758(3):260–263, March 2006. ISSN 0005-2736. doi: 10.1016/j.bbamem.2006.02.012. URL https://www.sciencedirect.com/science/article/pii/S0005273606000642.

Karagiannis, E. D., Urbanska, A. M., Sahay, G., Pelet, J. M., Jhunjhunwala, S., Langer, R., and Anderson, D. G. Rational Design of a Biomimetic Cell Penetrating Peptide Library. *ACS Nano*, 7(10):8616–8626, October 2013. ISSN 1936-0851, 1936-086X. doi: 10.1021/nn4027382. URL https://pubs.acs.org/doi/10.1021/nn4027382.

Landrum, G., Tosco, P., Kelley, B., Rodriguez, R., Cosgrove, D., Vianello, R., sriniker, gedeck, Jones, G., NadineSchneider, Kawashima, E., Nealschneider, D., Dalke, A., Swain, M., Cole, B., Turk, S., Savelev, A., Vaucher, A., Wójcikowski, M., Take, I., Scalfani, V. F., Probst, D., Ujihara, K., Walker, R., Pahl, A., godin, g., tadhurst cdd, Lehtivarjo, J., Bérenger, F., and Bisson, J. rdkit/rdkit: 2024_03_5 (Q1 2024) Release, July 2024. URL https://zenodo.org/records/12782092.

Lee, J., Song, H. S., Lee, S., Park, T. H., and Kim, B. Screening of cell-penetrating peptides using mRNA display. *Biotechnology Journal*, 7(3):387–396, March 2012. ISSN 1860-6768, 1860-7314. doi: 10.1002/biot.201100220. URL https://analyticalsciencejournals.onlinelibrary.wiley.com/doi/10.1002/biot.201100220.

Lester, B., Al-Rfou, R., and Constant, N. The Power of Scale for Parameter-Efficient Prompt Tuning, 2021. URL https://arxiv.org/abs/2104.08691. Version Number: 2.

Li, X. L. and Liang, P. Prefix-Tuning: Optimizing Continuous Prompts for Generation, 2021. URL https://arxiv.org/abs/2101.00190. Version Number: 1.

Liu, J., Heddleston, J., Perkins, D. R., Chen, J. J. H., Ghanbarpour, A., Smith, B. W., Miles, R., Aihara, E., and Afshar, S. Discovery of a new class of cell-penetrating peptides by novel phage display platform. *Scientific Reports*, 14(1):13437, June 2024. ISSN 2045-2322. doi: 10.1038/s41598-024-64405-w. URL https://www.nature.com/articles/s41598-024-64405-w.

Maetschke, S., Towsey, M., and Bodén, M. BLOMAP: AN ENCODING OF AMINO ACIDS WHICH IMPROVES SIGNAL PEPTIDE CLEAVAGE SITE PREDICTION. In *Proceedings of the 3rd Asia-Pacific Bioinformatics Conference*, pp. 141–150, Institute for Infocomm Research (Singapore), January 2005. PUBLISHED BY IMPERIAL COLLEGE PRESS AND DISTRIBUTED BY WORLD SCIENTIFIC PUBLISHING CO. ISBN 978-1-86094-477-2 978-1-86094-732-2. doi: 10.1142/9781860947322_0014. URL http://www.worldscientific.com/doi/abs/10.1142/9781860947322_0014.

Manavalan, B. and Patra, M. C. MLCPP 2.0: An Updated Cell-penetrating Peptides and Their Uptake Efficiency Predictor. *Journal of Molecular Biology*, 434(11):167604, June 2022. ISSN 00222836. doi: 10.1016/j.jmb.2022.167604. URL https://linkinghub.elsevier.com/retrieve/pii/S002228362200184X.

Manavalan, B., Subramaniyam, S., Shin, T. H., Kim, M. O., and Lee, G. Machine-Learning-Based Prediction of Cell-Penetrating Peptides and Their Uptake Efficiency with Improved Accuracy. *Journal of Proteome Research*, 17(8):2715–2726, August 2018. ISSN 1535-3893, 1535-3907. doi: 10.1021/acs.jproteome.8b00148. URL https://pubs.acs.org/doi/10.1021/acs.jproteome.8b00148.

Marks, J. R., Placone, J., Hristova, K., and Wimley, W. C. Spontaneous Membrane-Translocating Peptides by Orthogonal High-Throughput Screening. *Journal of the American Chemical Society*, 133(23):8995–9004, June 2011. ISSN 0002-7863, 1520-5126. doi: 10.1021/ja2017416. URL https://pubs.acs.org/doi/10.1021/ja2017416.

Martin, M. E. and Rice, K. G. Peptide-guided gene delivery. *The AAPS Journal*, 9(1):E18–E29, March 2007. ISSN 1550-7416. doi: 10.1208/aapsj0901003. URL http://link.springer.com/10.1208/aapsj0901003.

Milletti, F. Cell-penetrating peptides: classes, origin, and current landscape. *Drug Discovery Today*, 17(15-16):850–860, August 2012. ISSN 13596446. doi: 10.1016/j.drudis.2012.03.002. URL https://linkinghub.elsevier.com/retrieve/pii/S1359644612000839.

Mueller, J., Kretzschmar, I., Volkmer, R., and Boisguerin, P. Comparison of Cellular Uptake Using 22 CPPs in 4 Different Cell Lines. *Bioconjugate Chemistry*, 19(12): 2363–2374, December 2008. ISSN 1043-1802, 1520-4812. doi: 10.1021/bc800194e. URL https://pubs.acs.org/doi/10.1021/bc800194e.

Numata, K., Horii, Y., Oikawa, K., Miyagi, Y., Demura, T., and Ohtani, M. Library screening of cell-penetrating peptide for BY-2 cells, leaves of Arabidopsis, tobacco, tomato, poplar, and rice callus. *Scientific Reports*, 8 (1):10966, July 2018. ISSN 2045-2322. doi: 10.1038/s41598-018-29298-6. URL https://www.nature.com/articles/s41598-018-29298-6.

Oikawa, K., Islam, M. M., Horii, Y., Yoshizumi, T., and Numata, K. Screening of a Cell-Penetrating Peptide Library in *Escherichia coli* : Relationship between Cell Penetration Efficiency and Cytotoxicity. *ACS Omega*, 3(12):16489–16499, December 2018. ISSN 2470-1343, 2470-1343. doi: 10.1021/acsomega.8b02348. URL https://pubs.acs.org/doi/10.1021/acsomega.8b02348.

Pandey, P., Patel, V., George, N. V., and Mallajosyula, S. S. KELM-CPPpred: Kernel Extreme Learning Machine Based Prediction Model for Cell-Penetrating Peptides. *Journal of Proteome Research*, 17(9):3214–3222, September 2018. ISSN 1535-3893, 1535-3907. doi: 10.1021/acs.jproteome.8b00322. URL https://pubs.acs.org/doi/10.1021/acs.jproteome.8b00322.

Park, H., Park, J.-H., Kim, M. S., Cho, K., and Shin, J.-M. In Silico Screening and Optimization of Cell-Penetrating Peptides Using Deep Learning Methods. *Biomolecules*, 13(3):522, March 2023. ISSN 2218-273X. doi: 10.3390/biom13030522. URL https://www.mdpi.com/2218-273X/13/3/522.

Patel, S. G., Sayers, E. J., He, L., Narayan, R., Williams, T. L., Mills, E. M., Allemann, R. K., Luk, L. Y. P., Jones, A. T., and Tsai, Y.-H. Cell-penetrating peptide sequence and modification dependent uptake and subcellular distribution of green florescent protein in different cell lines. *Scientific Reports*, 9(1): 6298, April 2019. ISSN 2045-2322. doi: 10.1038/s41598-019-42456-8. URL https://www.nature.com/articles/s41598-019-42456-8.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011. ISSN 1533-7928. URL http://jmlr.org/papers/v12/pedregosa11a.html.

Preto, A. J., Caniceiro, A. B., Duarte, F., Fernandes, H., Ferreira, L., Mourão, J., and Moreira, I. S. POSEIDON: Peptidic Objects SEquence-based Interaction with cellular DOmaiNs: a new database and predictor. *Journal of Cheminformatics*, 16(1):18, February 2024. ISSN 1758-2946. doi: 10.1186/s13321-024-00810-7. URL https://jcheminf.biomedcentral.com/articles/10.1186/s13321-024-00810-7.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. CatBoost: unbiased boosting with categorical features, January 2019. URL http://arxiv.org/abs/1706.09516. arXiv:1706.09516 [cs].

Rogers, D. and Hahn, M. Extended-Connectivity Fingerprints. *Journal of Chemical Information and Modeling*, 50(5):742–754, May 2010. ISSN 1549-9596, 1549-960X. doi: 10.1021/ci100050t. URL https://pubs.acs.org/doi/10.1021/ci100050t.

Ruseska, I. and Zimmer, A. Internalization mechanisms of cell-penetrating peptides. *Beilstein Journal of Nanotechnology*, 11:101–123, January 2020. ISSN 2190-4286. doi: 10.3762/bjnano.11.10. URL https://www.beilstein-journals.org/bjnano/articles/11/10.

Sanders, W. S., Johnston, C. I., Bridges, S. M., Burgess, S. C., and Willeford, K. O. Prediction of Cell Penetrating Peptides by Support Vector Machines. *PLoS Computational Biology*, 7(7):e1002101, July 2011. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1002101. URL https://dx.plos.org/10.1371/journal.pcbi.1002101.

Schwaar, T., Lettow, M., Remmler, D., Börner, H. G., and Weller, M. G. Efficient Screening of Combinatorial Peptide Libraries by Spatially Ordered Beads Immobilized on Conventional Glass Slides. *High-Throughput*, 8(2):11, April 2019. ISSN 2571-5135. doi: 10.3390/ht8020011. URL https://www.mdpi.com/2571-5135/8/2/11.

Settles, B. Active Learning Literature Survey. Technical Report, University of Wisconsin-Madison Department of Computer Sciences, 2009. URL https://minds.wisconsin.edu/handle/1793/60660. Accepted: 2012-03-15T17:23:56Z.

Shi, K., Xiong, Y., Wang, Y., Deng, Y., Wang, W., Jing, B., and Gao, X. PractiCPP: a deep learning approach tailored for extremely imbalanced datasets in cell-penetrating peptide prediction. *Bioinformatics*, 40(2):btae058, February 2024. ISSN 1367-4803, 1367-4811. doi: 10.1093/bioinformatics/btae058. URL https://academic.oup.com/bioinformatics/article/doi/10.1093/bioinformatics/btae058/7596624.

Song, Y. M., Park, Y., Lim, S. S., Yang, S.-T., Woo, E.-R., Park, I.-S., Lee, J. S., Kim, J. I., Hahm, K.-S., Kim, Y., and Shin, S. Y. Cell Selectivity and Mechanism of Action of Antimicrobial Model Peptides Containing Peptoid Residues. *Biochemistry*, 44(36):12094–12106, September 2005. ISSN 0006-2960, 1520-4995. doi: 10.1021/bi050765p. URL https://pubs.acs.org/doi/10.1021/bi050765p.

Stalmans, S., Bracke, N., Wynendaele, E., Gevaert, B., Peremans, K., Burvenich, C., Polis, I., and De Spiegeleer, B. Cell-Penetrating Peptides Selectively Cross the Blood-Brain Barrier In Vivo. *PLOS ONE*, 10(10):e0139652, October 2015. ISSN 1932-6203. doi: 10.1371/journal.pone.0139652. URL https://dx.plos.org/10.1371/journal.pone.0139652.

Tran, D. P., Tada, S., Yumoto, A., Kitao, A., Ito, Y., Uzawa, T., and Tsuda, K. Using molecular dynamics simulations to prioritize and understand AI-generated cell penetrating peptides. *Scientific Reports*, 11(1):10630, May 2021. ISSN 2045-2322. doi: 10.1038/s41598-021-90245-z. URL https://www.nature.com/articles/s41598-021-90245-z.

Tyagi, M., Rusnati, M., Presta, M., and Giacca, M. Internalization of HIV-1 Tat Requires Cell Surface Heparan Sulfate Proteoglycans. *Journal of Biological Chemistry*, 276(5):3254–3261, February 2001. ISSN 00219258. doi: 10.1074/jbc.M006701200. URL https://linkinghub.elsevier.com/retrieve/pii/S0021925818465027.

van Hilten, N., Verwei, N., Methorst, J., Nase, C., Bernatavicius, A., and Risselada, H. J. PMIpred: a physics-informed web server for quantitative protein-membrane interaction prediction. *Bioinformatics*, 40(2):btae069, February 2024. ISSN 1367-4811. doi: 10.1093/bioinformatics/btae069.

Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, February 1988. ISSN 0095-2338, 1520-5142. doi: 10.1021/ci00057a005. URL https://pubs.acs.org/doi/abs/10.1021/ci00057a005.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., Platen, P. v., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. HuggingFace's Transformers: State-of-the-art Natural Language Processing, July 2020. URL http://arxiv.org/abs/1910.03771. arXiv:1910.03771 [cs].

Wolfe, J. M., Fadzen, C. M., Choo, Z.-N., Holden, R. L., Yao, M., Hanson, G. J., and Pentelute, B. L. Machine Learning To Predict Cell-Penetrating Peptides for Antisense Delivery. *ACS Central Science*, 4(4):512–520, April 2018. ISSN 2374-7943, 2374-7951. doi: 10.1021/acscentsci.8b00098. URL https://pubs.acs.org/doi/10.1021/acscentsci.8b00098.

# A. Appendix

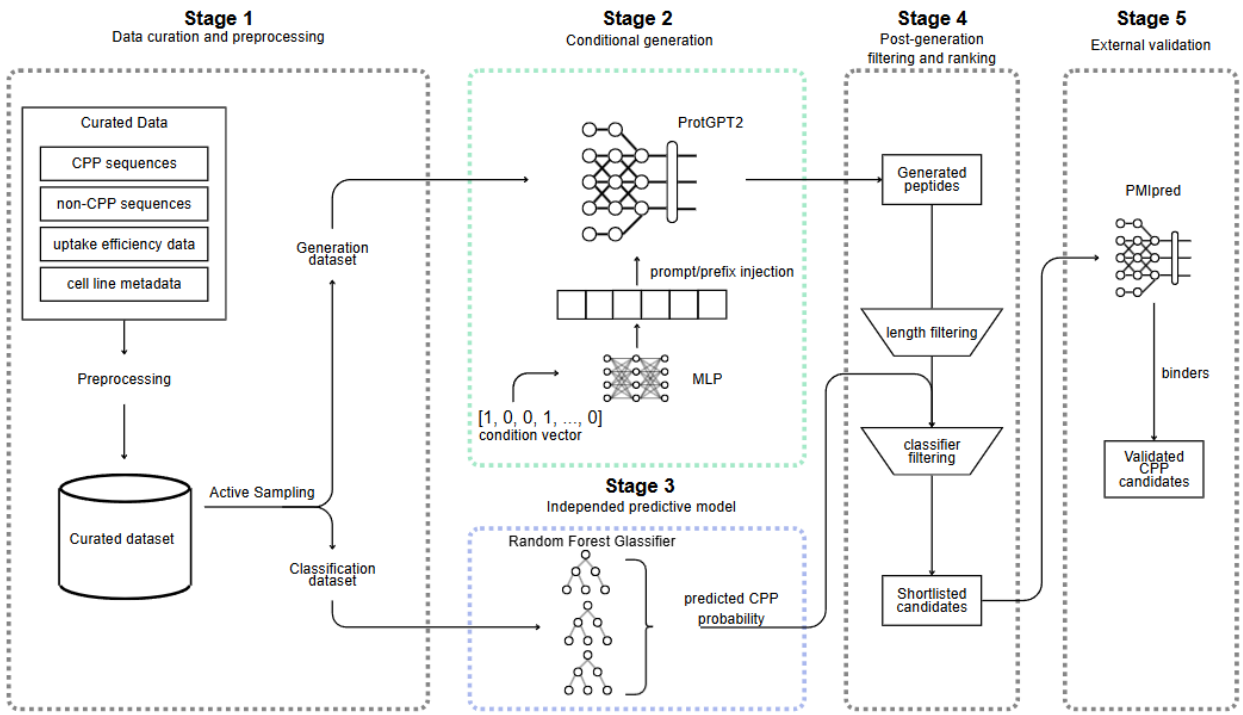


*Figure 3.* Overview of the framework.

*Table 3.* Results of hyperparameter optimization for Prefix and Soft-prompt tuning.

| RP | Valid AA (%) | CPP/Val AA (%) | Yield (%) | Set Size | Med $d_{\text{norm}}$ | Uniq. (%) | Entr. (AA) | $\sqrt{\text{JSD}}$ len | $\sqrt{\text{JSD}}$ AA | Cov $\leq 0.2$ | Med min $d_{\text{norm}}$ | Novel (%) | Loss | MMD |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Prefix tuning* | | | | | | | | | | | | | | |
| 1.0 | 88.71 | 47.09 | 41.77 | 824 | 0.815 | 56.36 | 0.797 | 0.320 | 0.239 | 0.21 | 0.31 | 91.26 | 2.38 | 0.138 |
| 1.1 | 85.43 | 50.13 | 42.83 | 909 | 0.828 | 60.64 | 0.816 | 0.374 | 0.220 | 0.21 | 0.35 | 92.08 | 2.75 | 0.123 |
| 1.15 | 79.34 | 49.37 | 39.17 | 884 | 0.829 | 64.48 | 0.831 | 0.404 | 0.181 | 0.20 | 0.37 | 93.10 | 3.13 | 0.115 |
| 1.2 | 69.14 | 49.63 | 34.31 | 805 | 0.824 | 67.03 | 0.839 | 0.411 | 0.164 | 0.18 | 0.38 | 93.91 | 3.66 | 0.134 |
| 1.25 | 59.80 | 49.26 | 29.46 | 685 | 0.828 | 66.44 | 0.848 | 0.470 | 0.147 | 0.17 | 0.43 | 94.31 | 4.26 | 0.151 |
| 1.3 | 48.46 | 46.82 | 22.69 | 533 | 0.833 | 67.13 | 0.859 | 0.515 | 0.144 | 0.15 | 0.46 | 92.50 | 4.98 | 0.179 |
| 1.4 | 32.03 | 47.73 | 15.29 | 368 | 0.839 | 68.79 | 0.869 | 0.616 | 0.128 | 0.10 | 0.57 | 94.84 | 6.18 | 0.242 |
| 1.5 | 19.80 | 49.64 | 9.83 | 254 | 0.842 | 73.84 | 0.874 | 0.721 | 0.121 | 0.07 | 0.69 | 95.67 | 7.11 | 0.296 |
| 1.6 | 11.31 | 61.11 | 6.91 | 216 | 0.824 | 89.26 | 0.869 | 0.847 | 0.127 | 0.08 | 0.74 | 95.37 | 7.84 | 0.336 |
| 1.7 | 8.49 | 70.03 | 5.94 | 200 | 0.811 | 96.15 | 0.864 | 0.937 | 0.129 | 0.04 | 0.75 | 97.50 | 8.10 | 0.353 |
| 1.8 | 6.66 | 77.68 | 5.17 | 171 | 0.812 | 94.48 | 0.875 | 0.980 | 0.123 | 0.01 | 0.76 | 98.83 | 8.20 | 0.364 |
| 1.9 | 6.09 | 82.16 | 5.00 | 172 | 0.810 | 98.29 | 0.868 | 0.988 | 0.127 | 0.01 | 0.76 | 99.42 | 8.30 | 0.364 |
| 2.0 | 6.29 | 83.18 | 5.23 | 183 | 0.813 | 100.00 | 0.875 | 0.988 | 0.120 | 0.01 | 0.76 | 99.45 | 8.31 | 0.368 |
| *Soft-prompt tuning* | | | | | | | | | | | | | | |
| 1.0 | 88.89 | 48.22 | 42.86 | 1485 | 0.917 | 99.00 | 0.854 | 0.254 | 0.232 | 0.05 | 0.47 | 99.87 | 5.74 | 0.241 |
| 1.1 | 80.34 | 52.45 | 42.14 | 1473 | 0.895 | 99.86 | 0.841 | 0.258 | 0.243 | 0.04 | 0.50 | 99.93 | 5.63 | 0.247 |
| 1.15 | 70.91 | 53.67 | 38.06 | 1330 | 0.895 | 99.85 | 0.851 | 0.302 | 0.250 | 0.03 | 0.50 | 99.92 | 5.51 | 0.266 |
| 1.2 | 60.09 | 54.40 | 32.69 | 1142 | 0.901 | 99.83 | 0.850 | 0.362 | 0.274 | 0.02 | 0.53 | 99.91 | 5.42 | 0.297 |
| 1.25 | 49.37 | 53.70 | 26.51 | 927 | 0.913 | 99.89 | 0.850 | 0.423 | 0.286 | 0.02 | 0.57 | 99.89 | 5.33 | 0.331 |
| 1.3 | 39.69 | 53.56 | 21.26 | 744 | 0.917 | 100.00 | 0.847 | 0.488 | 0.308 | 0.02 | 0.59 | 99.87 | 5.27 | 0.365 |
| 1.4 | 26.94 | 57.69 | 15.54 | 544 | 0.913 | 100.00 | 0.852 | 0.620 | 0.315 | 0.00 | 0.68 | 100.00 | 5.21 | 0.417 |
| 1.5 | 20.49 | 61.65 | 12.63 | 442 | 0.908 | 100.00 | 0.865 | 0.732 | 0.311 | 0.00 | 0.74 | 100.00 | 5.19 | 0.448 |
| 1.6 | 17.23 | 64.01 | 11.03 | 386 | 0.892 | 100.00 | 0.865 | 0.821 | 0.310 | 0.00 | 0.76 | 100.00 | 5.19 | 0.470 |
| 1.7 | 15.11 | 65.97 | 9.97 | 349 | 0.872 | 100.00 | 0.866 | 0.907 | 0.308 | 0.00 | 0.78 | 100.00 | 5.18 | 0.481 |
| 1.8 | 14.26 | 67.13 | 9.57 | 335 | 0.863 | 100.00 | 0.865 | 0.954 | 0.309 | 0.00 | 0.79 | 100.00 | 5.18 | 0.485 |
| 1.9 | 14.00 | 67.35 | 9.43 | 330 | 0.859 | 100.00 | 0.865 | 0.975 | 0.309 | 0.00 | 0.79 | 100.00 | 5.18 | 0.487 |
| 2.0 | 13.83 | 67.77 | 9.37 | 328 | 0.857 | 100.00 | 0.865 | 0.982 | 0.310 | 0.00 | 0.79 | 100.00 | 5.18 | 0.488 |

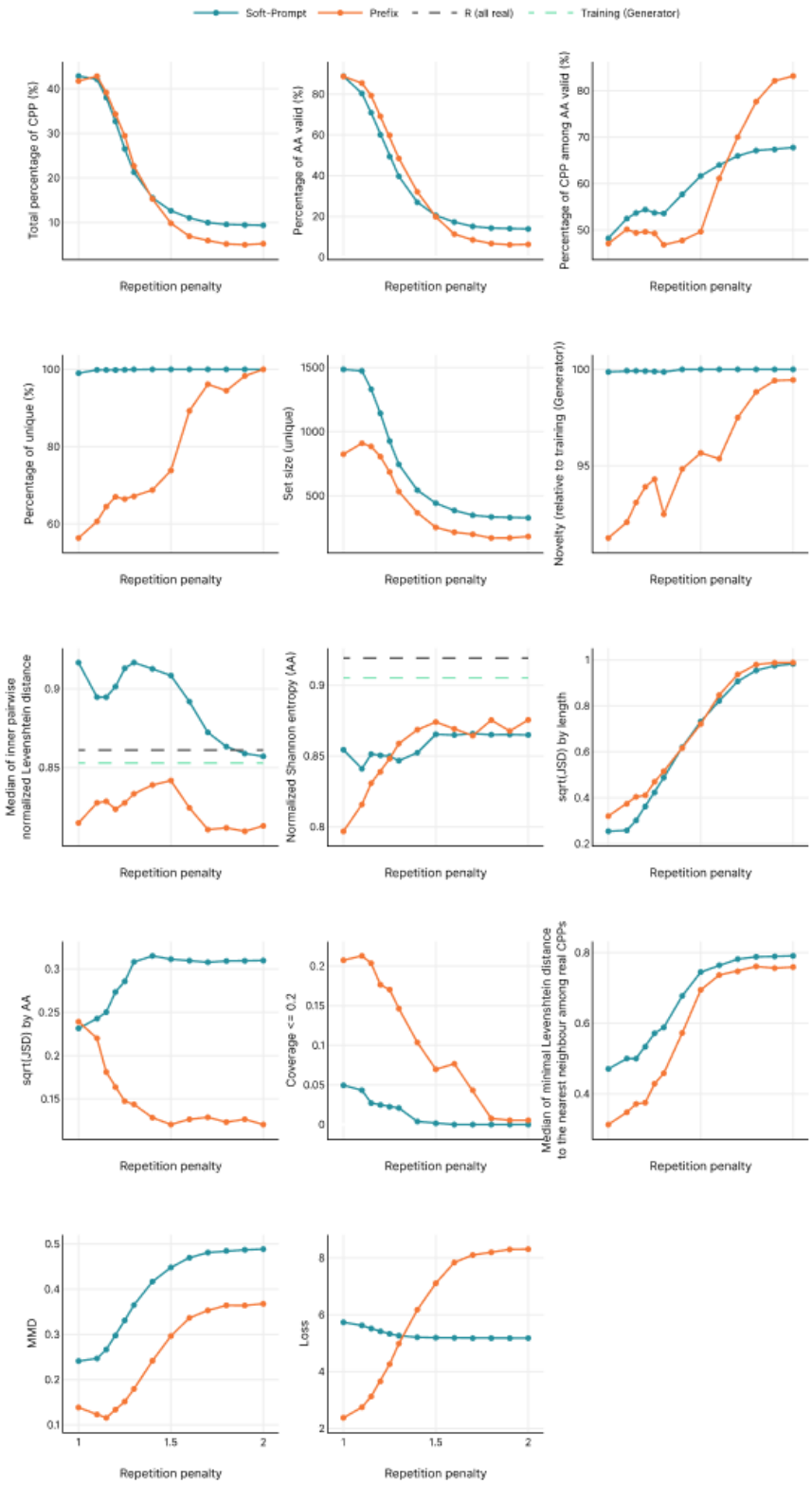


*Figure 4.* Results of hyper parameter optimization.

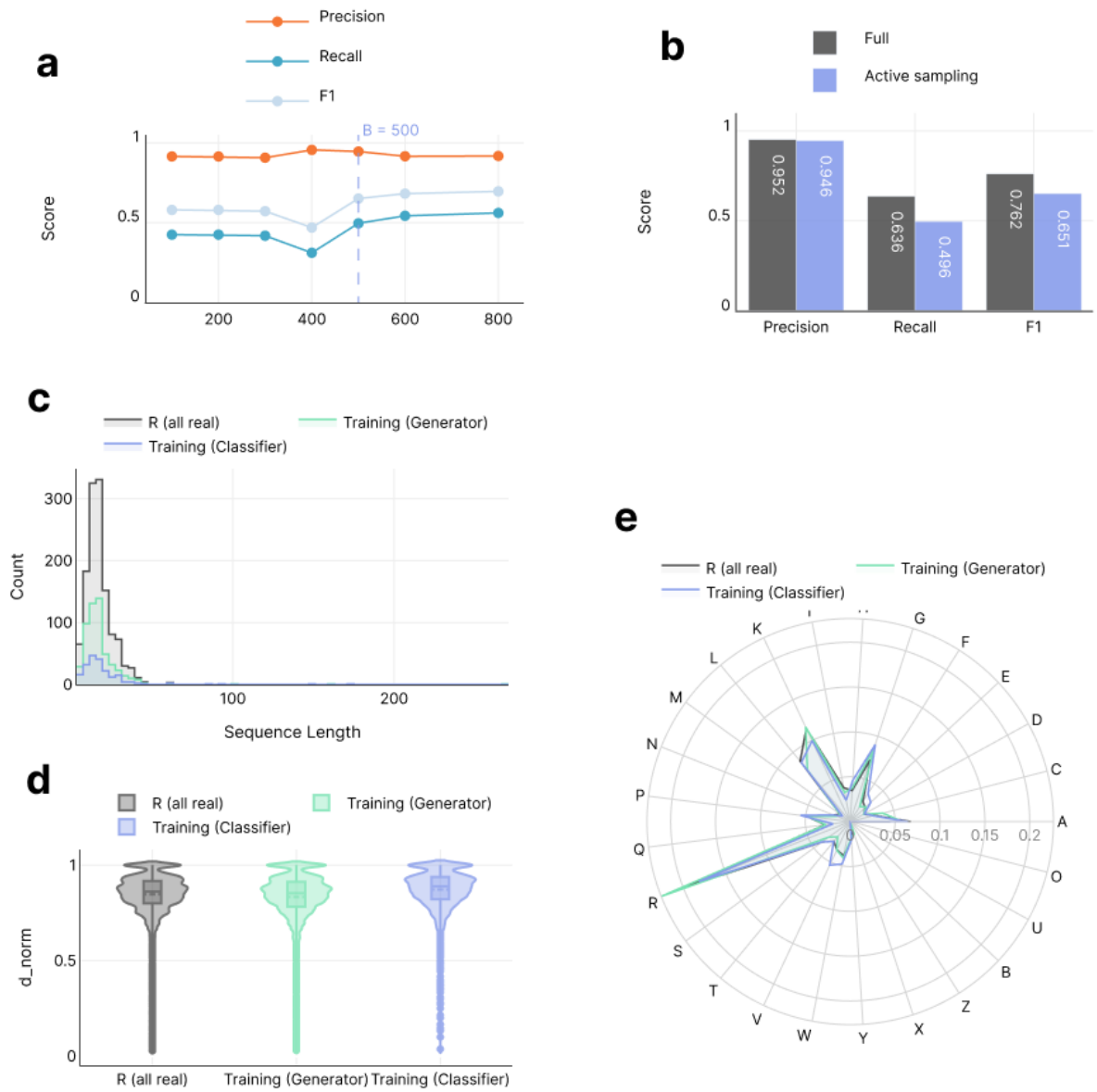


*Figure 5.* Active sampling implementation results. A - Performance metrics of the classifier trained on active sampling subsets, B - comparison of performance and baseline model, C, D, E - comparison of the length, internal distance, and AA ratio respectively between the resulting subsamples.

*Table 4.* Comparison of classifier performance.

| Model | Feature set | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|---|
| CatBoost | Morgan fingerprints | 0.836 | 0.847 | 0.852 | **0.850** |
| | Interpretable Feature Subset | 0.835 | 0.817 | **0.875** | 0.845 |
| | Blomap | **0.842** | **0.863** | 0.813 | 0.837 |
| | ProtBERT | 0.811 | 0.805 | 0.820 | 0.812 |
| Random Forest | Interpretable Feature Subset | **0.840** | 0.839 | **0.854** | **0.846** |
| | Blomap | **0.840** | **0.884** | 0.783 | 0.830 |
| | Morgan fingerprints | 0.812 | 0.831 | 0.821 | 0.826 |
| | ProtBERT | 0.798 | 0.793 | 0.805 | 0.799 |
| SVM | ProtBERT | **0.824** | 0.808 | **0.849** | **0.828** |
| | Morgan fingerprints | 0.809 | **0.814** | 0.840 | 0.827 |
| | Interpretable Feature Subset | 0.789 | 0.785 | 0.811 | 0.798 |
| | Blomap | 0.785 | **0.814** | 0.739 | 0.775 |
| Logistic Regression | Morgan fingerprints | **0.788** | **0.798** | **0.818** | **0.807** |
| | Blomap | 0.785 | 0.778 | 0.798 | 0.788 |
| | ProtBERT | 0.776 | 0.776 | 0.776 | 0.776 |
| | Interpretable Feature Subset | 0.759 | 0.758 | 0.782 | 0.770 |
| Decision Tree | Morgan fingerprints | **0.762** | **0.775** | **0.792** | **0.784** |
| | Interpretable Feature Subset | 0.748 | 0.747 | 0.771 | 0.759 |
| | ProtBERT | 0.711 | 0.714 | 0.706 | 0.710 |
| | Blomap | 0.706 | 0.701 | 0.717 | 0.709 |
| Final model — Tuned Random Forest | Interpretable Feature Subset | 0.796 | 0.952 | 0.636 | 0.762 |

*Table 5.* Hyperparameter configuration of the precision-optimized Random Forest classifier.

| Parameter | Search range | Selected value |
|---|---|---|
| Number of estimators | 200–2 000 | 1 200 |
| Maximum depth | 2–50 | 30 |
| Min. samples per split | 2–20 | 6 |
| Min. samples per leaf | 1–20 | 2 |
| Maximum features | sqrt / log2 / None | log2 |
| Bootstrap | True / False | False |
| Class weights | None / balanced / balanced_subsample | balanced_subsample |
| Probability calibration | — | Isotonic regression (5-fold CV) |
| Decision threshold | — | 0.825 |