Predicting Sequence-Activity Relationships among Antimicrobial Peptides

Deepti Kannan
dkannan@stanford.edu

Vinh Nguyen
vnguyen5@stanford.edu

***Abstract*—Antimicrobial peptides (AMPs) are membrane-active peptides that serve as promising therapeutic alternatives to antibiotics. Due to their diverse sequences, AMPs are active against a variety of microbes including bacteria, fungi, viruses, and cancer cells. Here, we present the use of machine learning techniques to classify antifungal AMPs from antibacterial AMPs using sequence-derived physico-chemical features. Although our best models performed moderately well with 70-75% accuracy, we attempt to rationalize the high variance using feature reduction techniques.**

***Index Terms*—chemoinformatics, machine learning, protein-function prediction, antimicrobial peptides**

I. Introduction

Rates of antibiotic resistance among dangerous pathogens have been on the rise due to the indiscriminate use of conventional antibiotics. As a result, researchers are interested in studying and designing therapeutic alternatives for treating infectious diseases. One such alternative are antimicrobial peptides (AMPs), a class of membrane-active peptides that are crucial to innate host defense. AMPs are relatively short with less than 50 amino acids, net cationic charge, and amphiphilicity (segregated hydrophobic and hydrophilic residues) [1]. Due to their diverse sequences and secondary structures, these ancient peptides demonstrate activity against gram-negative and gram-positive bacteria, viruses, fungi, some cancer cells, and other microbes. However, distinguishing among AMP activities can provide novel insights into biophysical mechanisms of action against specific microbes (i.e. bacteria vs. fungi). In addition, screening for activity is essential for designing targeted therapies. For example, AMPs potent against pathogenic bacteria should not interfere with commensal fungi in the human microbiome. With the rise of precision medicine, classifying anti-fungal versus. antibacterial AMPs can have long-standing consequences in developing cutting-edge therapies.

Previous machine learning AMP studies have focused on empirical quantitative structure activity relationship (QSAR) models, which relate quantitative properties of a compound (known as descriptors) to its biological activity. QSAR models for AMP discovery include linear regression, principle component analysis (PCA), support vector classifiers (SVC), and artificial neural networks (ANN) [2,3,4]. For example, Lata et. al analyzed C and N-terminal residues of 486 AMPs to develop ANN and SVC models [5]. Fjell et al. developed a QSAR-descriptor based ANN model to screen 100,000 candidates with 94% accuracy in identifying peptides active against drug-resistant bacteria [6]. Cherkasov et al. iteratively synthesized thousands of nine-residue peptides and trained ANN models to discover peptides with activities against drug-resistant superbugs [7].

The above studies rely on high-throughput screens of candidate AMPs to experimentally measure descriptors and predict activity. Our work attempts to find a more efficient way to predict AMP activity by computationally generating descriptors from primary amino acid sequences alone and leveraging online databases. In principal, it is possible to build models based on sequence homology to a known class of AMPs; however the variable length of peptides along with the incredible sequence diversity within a given class of AMPs makes this approach ineffective [8]. As a result, we instead compute physico-chemical features from primary amino-acid sequences as the inputs to our algorithms.

This approach proved effective in a recent paper by Lee, et. al (2017), in which UCLA researchers developed a SVM that was able to predict with over 90% accuracy whether an alpha-helical peptide had membrane-active behavior [9, 10]. Their screen was designed to distinguish AMPs from membrane-inactive decoy peptides; in contrast, we hope to expand on their research by further parsing the activity of known AMPs – in particular, classifying antifungal AMPs (class 1) from solely antibacterial AMPs (class 0). To our knowledge, we are the first to design such a classifier using physico-chemical features, in the absence of experimental data. In this paper, we present a variety of supervised learning algorithms, including four SVCs and logistic regression, trained to perform the binary classification task. We demonstrate the performance of these algorithms on different feature subsets and rationalize the results using principal components analysis (PCA) and feature reduction.

II. Methods

A. Data and Features

A total of 511 unique, alpha-helical AMP sequences with activity labels were pooled from three publicly-available online databases: the Collection of Antimicrobial Peptides (CAMP), the Antimicrobial Peptide Database (APD), and the Data Repository of Antimicrobial Peptides (DRAMP) [11, 12, 13]. Among these peptides, 203 are antifungal; these constitute the positive class (+1). In order to maintain a balanced training set, the remaining 308 antibacterial sequences were randomly subsampled down to 203, resulting in a mini data set of 406 peptides. We set aside 40 of these sequences (10% of the mini data set) for the test set. At each fold of cross-validation, 42 sequences (10% of the mini data set) were randomly chosen for the development set. We used the remaining 324 sequences for training, maintaining balanced classes at each split. The training/development data was used to perform feature selection and model testing, whereas the test data is only used to evaluate the final models on a blind dataset. The breakdown of the dataset is summarized in Table 1.

Each sequence in the databases consists of a string of letters, each representing an amino acid. We used the python propy package in order to compute 1411 physicochemical descriptors for all 406 sequences in the mini data set [14].These features fall into 5 categories: basic features (offset, peptide length, net charge), residue composition (420 descriptors), autocorrelation (720 descriptors), physicochemical composition (147 descriptors), and sequence order features (121 descriptors). The feature categories are described in Table 2. The full list of features can be found in the propy package user guide [14].

TABLE I: Data Breakdown

	Antifungal	**Antibacterial**
Training Set	162	162
Development Set	21	21
Test Set	20	20
Total	203	308

TABLE II: Feature Breakdown

Category	**Number of Features**
charge, length, offset	3
residue composition	420
autocorrelation	720
physiochemical composition	147
sequence order	121
Total	1411

B. Feature Selection

Because the number of descriptors exceeds the number of examples in our data set, feature selection techniques were employed to reduce the dimensionality of the data. Due to the large size of the feature space, exhaustively searching all subsets of 1411 features is computationally intractable. Instead, we opted to perform embedded feature selection by training a l1-penalized linear Support Vector Classifier (SVC), the approach described in Bi et. al [15]. The l1-penalty causes the weights of irrelevant features to go to zero when maximizing the objective function. We then used the optimal subset of features to train a variety of other models, including logistic regression and the more conventional l2-norm linear and nonlinear SVCs.

When training the l1-norm linear SVC, we use the squared hinge loss function to perform optimization and feature selection using the LinearSVC package in scikit-learn [16]. In order to determine the hyperparameter C, which controls the sparsity of the weight vector and therefore the number of selected features, we performed grid search using n=15 rounds of stratified shuffle cross-validation, and chose the value of C that maximized dev set accuracy (C_{opt} = 1.0). Running the l1-norm linear SVC on all 1411 features yielded a set of 256 features.

To further reduce the feature space, we also ran the l1 SVC over n=5 rounds of stratified shuffled cross-validation, selecting for features with non-zero weights upon each round. On average, 224 (+/- 14.15) features were selected each round of cross-validation. To account for the high variability in selected features due to the models sensitivity to the train/dev split, we only chose the features that were consistently selected in all 5 rounds, resulting in a bagged subset of 63 features. This bagging procedure is described in Breiman et. al [17]. We proceed to train our models on all 3 feature dimensions (N=1411, N=256, and N=63).

C. Learning Algorithms

We built five classifiers on each of the three feature subsets: Four support vector machines using various kernels (linear kernel with l1 or l2 regularization, RBF, and polynomial), and logistic regression. In total, we tested 15 models.

Support vector machines find an optimally-separating hyperplane in N-dimensional space, where N is the number of features. This is accomplished by maximizing the geometric margin, the minimum distance between the the separating hyperplane and the data set. Using kernels, we can map the feature set onto an higher, or even infinite-dimensional space to find the separating plane that may not exist in the lower dimensional space. The regularization parameter, C, controls how strongly large weights are penalized in the algorithm, and is tuned to achieve the best development set performance.

The optimization problem for the l1-norm SVM is as follows:

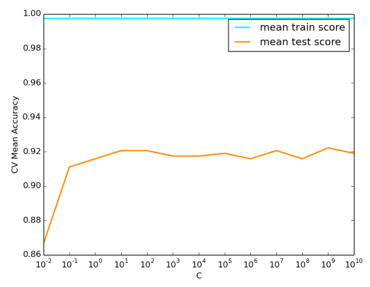
$$\underset{\mathbf{w},b}{\operatorname{argmin}}\left[\frac{1}{2}\left\|w\right\|_1+\frac{C}{m}\sum_{i=1}^{m}([1-y_i(\mathbf{w}\cdot\mathbf{x}_i+b)]_+)^2\right]. \quad (1)$$

where m is the number of training examples. The optimization problem for the l2-norm SVM is as follows:

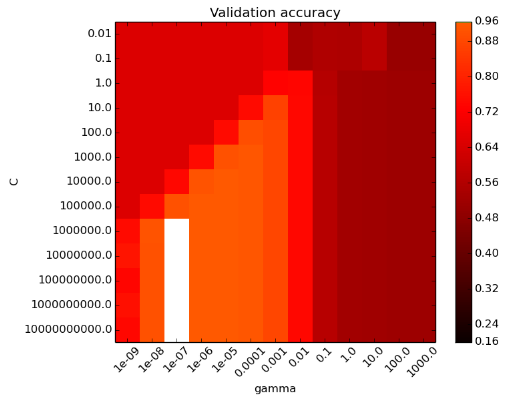
$$\underset{\mathbf{w},b}{\operatorname{argmin}}\left[\frac{1}{2}\left\|w\right\|_2+\frac{C}{m}\sum_{i=1}^{m}([1-y_i(\mathbf{w}\cdot\phi(\mathbf{x}_i)+b)]_+)^2\right]. \quad (2)$$

We used the following three kernels:

- linear:
$$\phi(\mathbf{x}_i,\mathbf{x}_j)=\mathbf{x}_i\cdot\mathbf{x}_j. \quad (3)$$
- polynomial:
$$\phi(\mathbf{x}_i,\mathbf{x}_j)=(\mathbf{x}_i\cdot\mathbf{x}_j)^d \mid d\in\mathbb{N}. \quad (4)$$
- RBF:
$$\phi(\mathbf{x}_i,\mathbf{x}_j)=\exp\big(-\gamma\left\|\mathbf{x}_i-\mathbf{x}_j)\right\|_2^2\big). \quad (5)$$



(a) 1



(b) 2

Fig. 1: Train and dev set accuracies over the range of hyperparameters during grid search for (a) linear l2 SVM, and (b) SVM with RBF kernel

Logistic regression attempts to fit an N-dimensional sigmoid function to the training examples, and classifies the training examples based on the output of the function being greater or less than 0.5. It does so by maximizing the likelihood of the parameters θ.

The optimization problem for regularized logistic regression is as follows:

$$\underset{\theta}{\operatorname{argmax}} -\lambda \|\theta\|_2^2 + \sum_{i=1}^{m} y^{(i)} \log(h(\mathbf{x}^{(i)})) + (1-y^{(i)}) \log(h(1-\mathbf{x}^{(i)})) \tag{6}$$

where

$$h(x^{(i)}) \equiv \frac{1}{1+\exp(\theta^T x^{(i)})}.$$

For each learning algorithm, the model hyperparameters, such as C for the SVMs, γ for the RBF kernal, and d for the polynomial kernel, were optimized via grid search to maximize the accuracy of the classifier over 15 rounds of stratified shuffled cross-validation. We plot the train set and dev set accuracies over the range of C and γ for the linear l2 SVM and the RBF kernel SVM in Fig. 1.

III. Results and Discussion

A. *Performance of Learning Algorithms*

After applying the optimal hyperparameters, we trained the algorithms once more on the balanced training set using 15 rounds of cross-validation, and evaluated on the test set. We compute the following set of point metrics to evaluate the algorithm's performance on the dev set (mean cross-validation accuracy) and the test set:

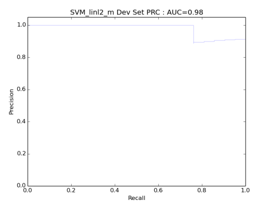
$$accuracy = (TP+TN)/(TP+FP+TN+FN) \tag{7}$$

$$precision = TP/(TP+FP) \tag{8}$$

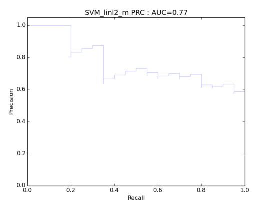
$$recall = TP/(TP+FN) \tag{9}$$

All 15 algorithms performed better than random chance (50% baseline). We found that the large feature set (N=1411) suffered from overfitting, but performed just as well on the dev set as it did on the test set. The smallest feature set (N=63) did not have sufficient information to explain the training set, but generalized to the test set with similar performance as in training. The best models in training were from the medium-sized feature set, but these did not generalize to the test set; they achieved similar accuracies as the models in the other feature set sizes. Table 3 shows the comparison of key point metrics among the 15 algorithms.

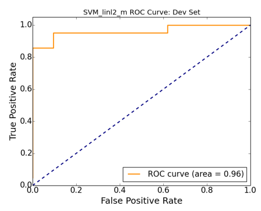
In Fig. 2, we also compare the ROC (Receiver Operating Characteristic) and Precision-Recall curves for the best model in training, the linear L2 SVM for the medium-sized feature set, which had a 94% dev set accuracy and a 70% test set accuracy.



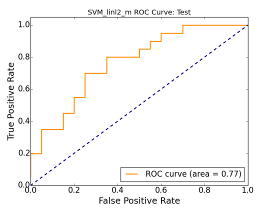
(a) 1



(b) 2



(c) 3



(d) 4

Fig. 2: PRC (a) and ROC (b) curves for linear l2-SVM

B. *Discussion*

Based on the hyperparameter search, we know that the models' performance is the best that can be achieved given our feature selection methodology. The performance, however, is still lower than desired, especially for the N=256 models, which performed well on the development set but did not generalize to the test set. One explanation for the high variance problem is that our linear l1-SVC feature selector chooses features to maximize cross-validated accuracy on the train-dev test. As a result, the input features are biased towards

TABLE III: Evaluation Metrics on All Models

Model	N	train accuracy	dev accuracy	dev precision	dev recall	test accuracy	test precision	test recall
SVM-rbf	1411	0.9512	0.6651	0.6756	0.6476	0.725	0.7917	0.5652
SVM-poly	1411	0.9132	0.5968	0.8353	0.2413	0.75	0.6833	0.6042
SVM-lin1	1411	0.9981	0.6032	0.6036	0.6127	0.625	0.6583	0.549
SVM-linl2	1411	0.9979	0.6159	0.6281	0.5714	0.65	0.6833	0.5556
logisticl2	1411	0.9981	0.6508	0.6523	0.6444	0.65	0.6833	0.5556
SVM-rbf-m	256	0.9835	0.7714	0.7533	0.819	0.675	0.7417	0.5556
SVM-poly-m	256	0.9302	0.5413	0.5643	0.8762	0.675	0.7417	0.5556
SVM-lin1-m	256	0.9981	0.8032	0.8091	0.7937	0.625	0.6917	0.5439
SVM-linl2-m	256	0.9977	0.9429	0.9361	0.9524	0.7	0.7333	0.5667
logsticl2-m	256	0.9973	0.9032	0.8919	0.9206	0.65	0.75	0.5455
SVM-rbf-s	63	0.7905	0.6476	0.6701	0.5905	0.575	0.675	0.5263
SVM-poly-s	63	0.7142	0.5825	0.8048	0.2286	0.65	0.65	0.5625
SVM-lin1-s	63	0.6949	0.654	0.6686	0.6286	0.625	0.6917	0.5439
SVM-linl2-s	63	0.694	0.6698	0.6798	0.6444	0.625	0.6917	0.5439
logisticl2-s	63	0.6984	0.6095	0.6362	0.5302	0.65	0.7167	0.55

sequences from the train-dev set, the same data used to cross-validate the final 15 models. This may explain the unusually high dev set accuracies for the medium-sized feature set models, as compared to the test set accuracies. In addition to the features not generalizing, we also acknowledge that our metrics are severely limited by the size of our data set. More sequences with known activity labels will be needed to improve model performance. Regardless, the fact that the three sets of models all performed similarly on the test set, despite having very different feature dimensions, suggests that the features we calculated are not very predictive of the activity that we are attempting to distinguish.

To better understand how feature dimensionality affects the performance of our algorithms, we varied the number of rounds of cross validation used in bagging features and plotted train/dev set accuracy vs. number of features in Fig. 3 for the linear l2-SVM. We found that train and dev set accuracy slightly increased with the number of features included in the model, suggesting that the features are highly biased towards the training set. We also performed principal components analysis (PCA) to determine whether projecting the feature space down to a smaller dimension might reduce overfitting. We determined that 227 principal components explained 95% of the variance in the features, but as the number of principal components increased, the development set accuracy did not change (Fig. 4). From this, we can conclude that although the feature set contains redundancies, eliminating those redundancies does not significantly improve model performance. Both of these experiments support the hypothesis that model accuracies cannot be improved significantly by varying the number or type of physico-chemical descriptors alone. These results point to the need for more informative features to sub-classify antimicrobial peptides with different functions. Indeed, the literature suggests that AMPs are not particularly selective against their target microbes – in fact, many antifungal AMPs demonstrate activity against bacteria as well [1]. Perhaps additional information about the structure, mechanism of action, or binding pocket of the peptide is necessary to predict microbe-specific activity.

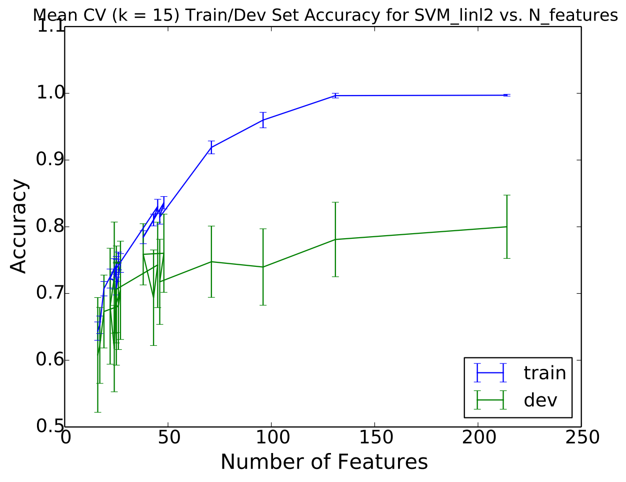


Fig. 3: Train and dev set accuracies vs. number of features selected during bagging for linear l2 SVM

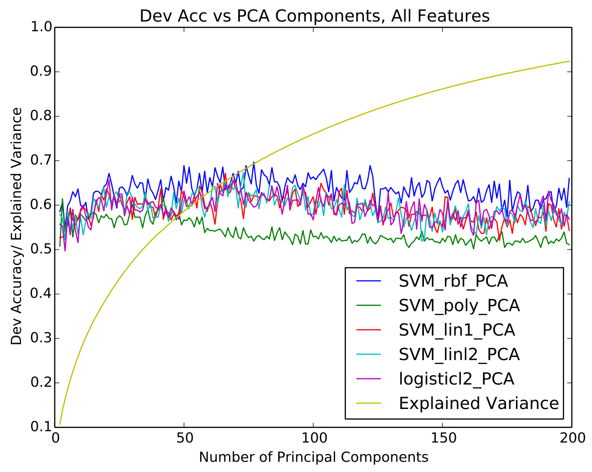


Fig. 4: Dev set accuracies for all 5 models and explained variance vs. number of principle components in PCA over all 1411 features.

IV. Conclusion and Future Work

In summary, we have described the use of supervised learning and feature selection techniques to attempt to classify antifungal AMPs from antibacterial AMPs. We found that four SVM-based models and logistic regression all performed the classification task with moderate success, with the best models achieving 70-75% accuracy on a blind test set. We attempted to rationalize model overfitting via feature reduction techniques (namely, PCA), but were unable to find an optimal feature subset of the 1411 physico-chemical descriptors that would significantly improve model accuracies.

As discussed above, the size of our data set poses a serious limitation to the success of our algorithms. For this work, we restricted our training set to alpha-helical sequences in order to account for secondary structure when making predictions (this selection criteria was also applied by other groups in similar work) [9]. However, provided more time and computational power, we would also incorporate beta-sheet peptides as well as sequences with unknown secondary structures to increase the size of our data set. In addition, we would perform a more thorough feature subset search to confirm our hypothesis that physico-chemical descriptors are insufficient to accurately perform this classification task. Finally, we would attempt different classes of learning algorithms, such as neural networks, to better assess the limitations of SVMs. Overall, though, these results provide novel insights into the predictive value of physico-chemical features in parsing out microbe-specific activity among antimicrobial peptides.

Acknowledgment

We would like to thank Ernest Lee, UCLA Dept. of Bioengineering, for his guidance at all stages of project development.

References

[1] Cruz, J., et al. Antimicrobial peptides: promising compounds against pathogenic microorganisms. Current medicinal chemistry 21.20 (2014): 2299-2321.

[2] Hilpert K, Fjell CD, Cherkasov A. Short linear cationic antimicrobial peptides: Screening, optimizing, and prediction. Methods Mol Biol. 2008;494(8):127159.

[3] Mitchell JBO. Machine learning methods in chemoinformatics. Wiley Interdiscip Rev Comput Mol Sci. 2014;4(5):468481.

[4] Yee LC, Wei YC. Current modeling methods used in QSAR/QSPR. In: Dehmer M, Varmuza K, Bonchev D, editors. Statistical Modelling of Molecular Descriptors in QSAR/QSPR. Vol 2. Wiley-VCH; Weinheim, Germany: 2012. pp. 131.

[5] Lata S, Sharma BK, Raghava G. Analysis and prediction of antibacterial peptides. BMC Bioinformatics. 2007;8:263.

[6] Fjell CD, et al. Identification of novel antibacterial peptides by chemoinformatics and machine learning. J Med Chem. 2009;52(7):20062015.

[7] Cherkasov A, et al. Use of artificial intelligence in the design of small peptide antibiotics effective against a broad spectrum of highly antibiotic-resistant superbugs. ACS Chem Biol. 2009;4(1):6574.

[8] Chou K-C (2009) Pseudo Amino Acid Composition and its Applications in Bioinformatics,Proteomics and System Biology. CP 6(4):262274.

[9] Lee, Ernest Y., Gerard CL Wong, and Andrew L. Ferguson. ”Machine learning-enabled discovery and design of membrane-active peptides.” Bioorganic & Medicinal Chemistry (2017).

[10] Lee EY, Lee MW, Fulan BM, Ferguson AL, Wong GCL. What can machine learning do for antimicrobial peptides,and what can antimicrobial peptides do for machine learning? Interface Focus 7: 20160153.(2017)

[11] Waghu FH, Barai RS, Gurung P, Idicula-Thomas S. CAMPR3: a database on sequences, structures and signatures of antimicrobial peptides. Nucleic Acids Research, Volume 44, Issue D1, 4 January 2016, Pages D1094D1097.

[12] Wang G, Li X, Wang Z. APD3: the antimicrobial peptide database as a tool for research and education. Nucleic Acids Research, Volume 44, Issue D1, 4 January 2016, Pages D1087D1093.

[13] Fan L, Sun J, Zhou M, Zhou J, Lao X, Zheng H, Xu H. DRAMP: a comprehensive data repository of antimicrobial peptides. Sci Rep, 2016, 6, 24482. PMID: 27075512

[14] Cao et al. propy: a tool to generate various modes of Chous PseAAC. Bioinformatics 29.7 (2013): 960-962.

[15] Bi J et al. Dimensionality reduction via sparse support vector machines. Journal of Machine Learning Research 3.Mar (2003): 1229-1243.

[16] Pedregosa F et al. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12.Oct (2011): 2825-2830.

[17] Breiman, Leo. Bagging predictors. Machine learning 24.2 (1996): 123-140.