Finding Short Peptide Substrates using Bayesian Active Learning

Jialei Wang[1], Michael D. Burkart[2], Peter I. Frazier[1], Nathan Gianneschi[2], Michael K. Gilson[3], Nicholas Kosa[4], Lorillee Tallorin[2], and Pu Yang[*1]

[1]School of Operations Research & Information Engineering, Cornell University
[2]Department of Chemistry and Biochemistry, UC San Diego
[3]Skaggs School of Pharmacy and Pharmaceutical Sciences, UC San D
[4]Bioo Scientific Corp, Austin TX

Abstract

We consider a Bayesian active learning problem arising in biochemistry, in which we wish to find a peptide that (1) has a certain expensive-to-ascertain biochemical property (it is a substrate for two protein-modifying enzymes, phosphopantetheinyltransferase and ACP hydrolase); and (2) is as short as possible. Finding such a peptide would allow tracking protein interactions with great ease, and would support a number of innovations in medicine, biochemistry, and materials science. However, such peptides are difficult to find, because the set of peptides is large, only a small fraction have the desired property, and ascertaining whether or not a peptide has the desired property requires performing a time-consuming laboratory experiment. We present a machine learning method for choosing which peptides to test to find such a peptide as quickly as possible. We prove theoretical bounds on its solution quality, demonstrate in simulation that it outperforms two natural benchmark methods, and then describe how it was used in practice to find a peptide with the desired property that is shorter than the shortest previously known. While our method was developed for this specific application in biochemistry, it can also be used in other Bayesian active learning problems in which we wish to find an exemplar whose expensive-to-obtain binary label is positive, and for which a secondary easy-to-evaluate cost objective is as small as possible.

1 Introduction

In this paper, we use machine learning to address a specific instance of a problem common in medicine, materials science, chemistry, and other physical sciences. In this class of problems, we wish to find a molecule with a particular biological or chemical property. While we have historical data on related molecules or related properties that can support predictive modeling, the only way that we can determine with certainty whether a particular molecule has the property of interest (in colloquial language, ``is a hit'') is to synthesize and test it in a

[*]Author order is alphabetical, except for first author Jialei Wang.

time-consuming laboratory experiment. Success is hampered by the enormous search space, and the limits imposed by experimental cost on the number of molecules that can be tested. Within this context, we wish to use mathematical methods to recommend which experiments to perform, so as to reliably find a hit within a given experimental budget. Common examples include drug discovery, in which we wish to find a molecule that can be effective in treating a particular disease, and materials discovery, in which we wish to find a molecule that has a particular material property.

The instance of this problem that we consider in detail in this paper comes from biochemistry: we wish to find a peptide that is (1) as short as possible; and (2) is a substrate for two protein-modifying enzymes, phosphopantetheinyltransferase (PPTase) and ACP hydrolase (AcpH). These protein-modifying enzymes can be used to attach and detach arbitrary molecules to their substrates. In particular, we can use PPTase to attach a fluorescent ``tag" or ``label" to peptides that we would discover that are substrates for both enzymes, allowing us to track the peptide, starting from a time of our choosing. We can then use AcpH to remove the fluorescent tag at a time of our choosing. If this peptide were short, then it could be inserted into a protein or peptide-based catalytic complex of our choosing without disrupting its activity, allowing monitoring of biological and catalytic systems. This monitoring tool would have a number of applications in medicine, biology, and materials science (Kosa et al., 2012).

Finding such a peptide is challenging, because the number of peptides is large, and only a small fraction of them are substrates for both enzymes (i.e., are ``hits"). A peptide is a string of amino acids, of which 20 occur in nature, so there are 20^n peptides of length n. The typical length of the peptides under consideration is 15 amino acids long, and the number of peptides with length 15 is $20^{15} \approx 3.3 \times 10^{19}$. Of these, we estimate that approximately 1 in 10^4 is a substrate. We can test peptides in batches of size approximately 300, and each batch requires approximately one week of work, several thousand dollars in materials, and access to an expensive shared machine on which time can be reserved for a batch of experiments approximately once every two months. If we were to test peptides of length 15 uniformly at random, then we would need to test $n = \log(1/2)/\log(1 - 10^{-4}) \approx 6931$ peptides to ensure a probability of $1/2$ of finding at least one hit, which would in turn require 23 batches of experiments, and roughly four years' time.

To address this challenge, we formulate this problem as a Bayesian active learning problem. We first develop a Bayesian classification method, which combines Naive Bayes, a prior distribution encoding knowledge from domain experts, and training data from naturally occurring and longer peptide substrates, to predict whether a short peptide is a hit. We then use the joint probability distributed over unobserved labels (``hit" or ``miss") of all peptides created by this classification method within a combinatorial optimization framework to find a *set* of peptides to test that, when taken together, are most likely to produce a hit. We provide a theoretical performance guarantee on the quality of the resulting set of peptides to test, and using simulation, we show that this combination of prediction and optimization provides robust results that are better than a more naive use of the predictive model that considers only marginal probabilities, and a benchmark method of truncating and mutating known hits at random. We then describe laboratory results in which this method was used to discover peptide substrates that are shorter than the shortest previously known: the ybbR sequence (Zhou et al., 2007).

While the methods we develop were designed to address a particular problem arising in biochemistry, the formal model we propose is general, and can be used whenever we have a large collection of exemplars with expensive-to-obtain labels, and we wish to choose exemplars to evaluate so as to find one with both a positive label and a small secondary cost objective,

within a limited budget.

This paper builds on a number of recently developed methods for optimal search, all of which aim to effectively collect information so as to make the best decisions under uncertainty. In this setting, they need to trade off the reward by sampling (i.e. exploitation) and the cost by acquiring this information (i.e. exploration). For example, in drug discovery, people search for a chemical derivative of the base molecule that best treats disease. To achieve this goal, they choose molecules to test to maximize the expected quality of the best compound discovered (Negoescu et al., 2011). Since the budget for testing is limited, they need to test the most informative and high quality molecules. To address this problem, Jones & Schonlau proposed Expected Improvement algorithm to sample points sequentially (Jones et al., 1998). Ginsbourger used constant liar heuristic to extend Expected Improvement algorithm to parallel setting (Ginsbourger et al., 2008). There are quite a few papers about parallel sampling in active learning research community (Chen et al., 2013; Hoi et al., 2006b,a) , but they only aim to maximize information gain (i.e. pure exploration).

In section 2, we provide a formal model of our problem. In section 3, we state the Bayesian classification model used for the problem. In section 4, we provide the formulation of the proposed optimization algorithm, prove its performance guarantee, and summarize the algorithm in pseudo code. In section 5, we use available real data to validate our statistical model, compare the performance of our proposed optimization method against other methods, and show progress of this biochemical application using our method.

2 Problem Formulation

We now formulate our applied problem as an active learning problem. Let E be a generic space of exemplars. In our application, E is the space of peptides. Each element $x \in E$ has an unknown binary label $y(x) = \{0, 1\}$. In our application, $y(x)$ is 1 if x is a hit, i.e., is a substrate for both enzymes, and 0 if not. A known deterministic function $f(x)$ measures the cost or disutility associated with x. In our application, $f(x)$ is the length of the peptide, as longer peptides interfere more with the system being monitored. Our goal is to perform experiments so as to find x with a positive label and a cost $f(x)$ that is as small as possible.

To obtain labels of exemplars, we can do experiments in batches, which evaluate the labels for a subset $S \subseteq E$. The cardinality of S is constrained by some K (i.e. $|S| \leq K$), modeling our budget for performing experiments. We measure the quality of S by

$$f^*(S) = \min_{x \in S: y(x)=1} f(x), \tag{1}$$

where we assume $\min \emptyset = \infty$. $f^*(S)$ measures the smallest cost function for positive labeled elements in the set S, and in our application, that means the shortest length of desired peptides in the set of peptides to test.

Let b be a target value and we wish to find $S \subseteq E$ such that $f^*(S)$ is better than b. In our application, we let b be the length of the shortest previously known desired peptide. $f^*(S)$ is unknown because we do not know the label $y(x)$ for $x \in S$ yet. We assume a joint probability distribution over $y(x) : x \in E$, which is determined by the underlying statistical model for a particular problem, and for our application, the model is described in Section 3. Based on the probability distribution we consider the following two quality measures for S:

$$\begin{aligned} &\text{Probability of Improvement:} && \mathrm{P}^*(S) = \mathbb{P}(f^*(S) < b) \\ &\text{Expected Improvement:} && \mathrm{EI}(S) = \mathbb{E}[(b - f^*(S))^+] \end{aligned} \tag{2}$$

Given constraint on the cardinality of S, we wish to find S that maximizes one of these two measures. Let $g(S)$ be either $\mathrm{P}^*(S)$ or $\mathrm{EI}(S)$, and we can write our goal as

$$\max_{S \subseteq E : |S| \leq K} g(S). \tag{3}$$

3 Statistical model

To construct the joint probability distribution over $y(x) : x \in E$, which is needed to calculate (2) and to choose which peptides to test, we develop a Bayesian classification method for our application, which combines Naive Bayes and a prior distribution encoding knowledge from domain experts. Let $X = (X_1, \ldots, X_n)$ be an instance with n features and Y be its label. Using Bayes's Rule, we have:

$$\mathbb{P}(Y = y|X = x) = \frac{\mathbb{P}(X = x|Y = y)\mathbb{P}(Y = y)}{\mathbb{P}(X = x)} = \frac{\mathbb{P}(X = x|Y = y)\mathbb{P}(Y = y)}{\sum_{y'} \mathbb{P}(X = x|Y = y')\mathbb{P}(Y = y')}$$

The Naive Bayes classifier assumes that the presence or absence of a particular feature is unrelated to the presence or absence of any other feature, given the class variable, i.e.

$$\mathbb{P}(Y = y|X = x) = \frac{\prod_{j=1}^{n} \mathbb{P}(X_j = x_j|Y = y)\mathbb{P}(Y = y)}{\sum_{y'} \prod_{j=1}^{n} \mathbb{P}(X_j = x_j|Y = y)\mathbb{P}(Y = y)}$$

In our application, we have a set of peptides, each with length less than or equal to L. Each peptide is a sequence of amino acids. We use a reduced alphabet for amino-acids, i.e., we group them into K groups. For each peptide, let A_i be the amino acid on position j, and let X_i be the class of this amino acid. For a specific enzyme, let $Y(x) = 1$ if peptide x is a substrate for that enzyme and 0 if not.

We let $\theta_{y,j}(k) = \mathbb{P}(X_i = k|Y(X) = y)$, for each $j = 1, \ldots, L$, $k = 1, \ldots, K$ and $y \in \{0, 1\}$. We further assume some known prior distribution $\mathbb{P}(Y(x) = y)$, $y \in \{0, 1\}$. Let θ be the full set of parameters $\theta_{y,j}(k)$, for $j = 1, \ldots, L$, $k = 1, \ldots, K$ and $y \in \{0, 1\}$. Then, given an unlabeled peptide, we can calculate its probability of being a substrate as:

$$\mathbb{P}\left(Y(x) = 1|\theta\right) = \frac{\mathbb{P}(Y(x) = 1) \prod_j \theta_{1,j}(x_j)}{\left[\mathbb{P}(Y(x) = 1) \prod_j \theta_{1,j}(x_j)\right] + \left[\mathbb{P}(Y(x) = 0) \prod_j \theta_{0,j}(x_j)\right]} \tag{4}$$

We estimate the parameters $\theta_{y,j}(k)$ using Bayesian inference. We assume for each $j = 1, \ldots, L$, $y \in \{0, 1\}$, the vector $\theta_{y,j} \sim \text{Dirichlet}(\alpha_{y,j}(1), \ldots, \alpha_{y,j}(K))$. A good initial choice for the parameter vector $\alpha_{y,j} = (\alpha_{y,j}(1), \ldots, \alpha_{y,j}(6))$ can be choosing $\alpha_{y,j}(k)$ to be constant across k, and y, and to only depend upon j. Since amino acids further from the serine are less likely to have a strong influence on its activity, we choose this value to be 1 in the positions next to the serine and to increase as j moves further.

We further assume two hyper parameters γ_0 and γ_1 that characterize the distribution for $y = 0$ and $y = 1$ respectively. Then, with the prior distribution and hyper parameters, our posterior distribution is also Dirichlet. In particular, it is $\text{Dirichlet}(\alpha_{y,j}(1) + \gamma_y N_{y,j}(1), \ldots, \alpha_{y,j}(K) + \gamma_y N_{y,j}(K))$, where $N_{y,j}(k)$ counts how many peptides x in the training data with $Y(x) = y$ had $x_j = k$. That is, it counts how many peptides had amino acid j in class j.

Since our training data is expensive and highly skewed, we use the leave-one-out cross validation procedure to choose the optimal hyper parameters. For each setting of the hyper parameters, we obtain an receiver operating characteristic(ROC) curve using the result of the leave-one out procedure and choose the setting with highest AUC(area under curve).

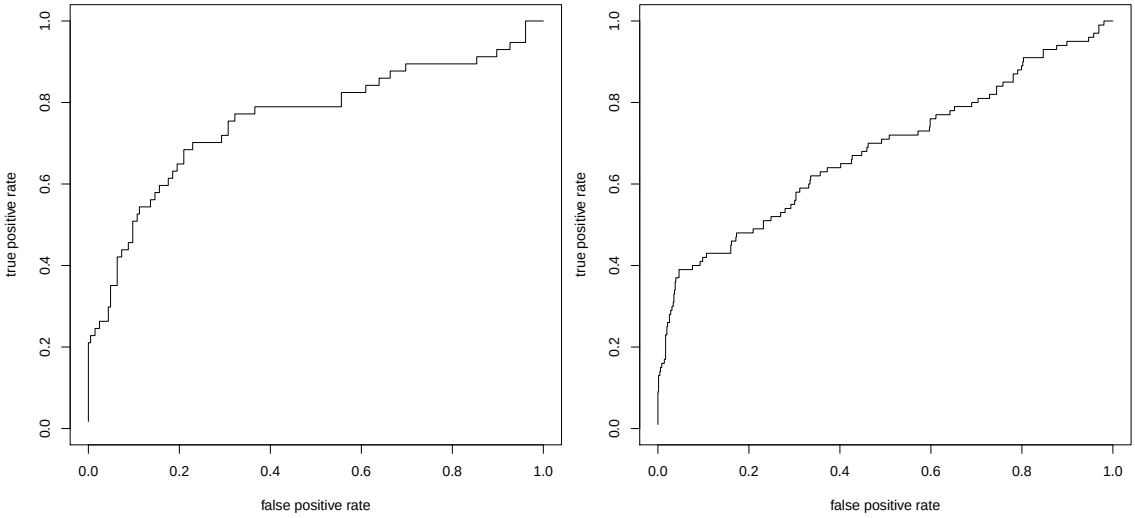

Figure 1: ROC curve using leave-one-out cross validation

We also use leave-one-out cross validation to plot ROC curve and validate our model. Currently we have three sets of data: data set #0 is the preliminary data obtained from biochemistry literature and has 36 data points; data set #1 contains 258 data points evaluated in the first round of experiments using our algorithm; data set #2 has 580 data points evaluated in the second round of experiments. In Figure 1, we compare the performance of our model using different combinations of data sets, and observe that ROC curve to the left is better than the one to the right. We think that this is because data set #2 was generated by our algorithm based on the previous two data sets, and due to the exploration manner of our algorithm, data set #2 should lie in the region that is more challenging for the classifier. Thus it is reasonable that our classifer performs worse using all three data sets.

4 Optimization Algorithm

(3) is a combinatorial optimization problem with huge feasible region. This problem is considered NP-hard and yet there is no known algorithm that solves it quickly. We propose an approximation algorithm, which solves (3) using greedy heuristic, that is, starting with the empty set $S = \emptyset$, iteratively find element e such that

$$\arg\max_{e \in E \setminus S} g(S \cup \{e\}), \tag{5}$$

and incorporate it into S until $|S| = K$ for some chosen K. This approach reduces the size of search space dramatically from $|E|^{|S|}$ down to $|E| \times |S|$, and we prove that if objective function g is either P* or EI, the proposed greedy algorithm is guaranteed to be near-optimal with a lower bound. Additionally, we show that under the statistical model described in section 3, we can formulate (5) as Mixed-Integer Nonlinear Programming (MINLP) and solve it efficiently using a off-the-shelf MINLP solver.

4.1 Performance guarantee for the greedy algorithm

In this subsection, we show that when the objective function g in (5) is either P* or EI, the proposed greedy algorithm has performance guarantee. The result is stated in the following:

Theorem 1. *If objective function is probability of improvement (i.e $P^*(S)$) or expected improvement (i.e* $\mathrm{EI}(S)$*), the greedy algorithm is guaranteed to achieve a factor* $(1-1/e)(\approx 63\%)$ *of the optimal value.*

We prove the theorem using the following three lemmas. Lemma 1 is a result from the analysis of greedy heuristic in combinatorial optimization by Nemhauser, which provides lower bound of greedy heuristic given that objective function satisfies certain conditions. Lemma 2 and 3 show P* and EI are the objective functions that satisfy conditions stated in Lemma 1, and thus greedy algorithm has a lower bound.

Lemma 1. *(Nemhauser et al., 1978) If $F(S)$ is submodular, nondecreasing and $F(\emptyset) = 0$, the greedy heuristic always produces a solution whose value is at least $1 - [(K-1)/K]^K$ times the optimal value, where $|S| \leq K$. This bound can be achieved for each K and has a limiting value of $1 - 1/e$, where e is the base of the natural logarithm.*

Lemma 2. *Probability of improvement $P^*(S)$ is submodular, nondecreasing and $P^*(\emptyset) = 0$.*

Lemma 3. *Expected improvement* $\mathrm{EI}(S)$ *is submodular, nondecreasing and* $\mathrm{EI}(\emptyset) = 0$.

Proof of Theorem 1. From Lemma 2 and 3, we know that P* and EI are submodular, nondecreasing and their measure of the empty set is 0. From Lemma 1, we conclude that if the objective function g in (5) is P* or EI, the greedy algorithm is guaranteed to achieve a factor of $(1 - 1/e)$ of the optimal value. □

4.2 Probability of Improvement

We first show in Proposition 1 that (5) can be written into the form in which the objective function can be easily calculated using any statitical classifier. This is the general form of greedy algorithm for optimization over probability of improvement and can be used with any underlying statistical model. In addition, we show that, using the Naive Bayes classifier proposed in section 3, we can further formulate (5) as a MINLP, which can be solved efficiently using an off-the-shelf MINLP solver.

Proposition 1. *If the objective function g is P*, we can write* (5) *as*

$$\underset{e \in E \setminus S, f(e) < b}{\arg\max} \quad \mathbb{P}(y(e) = 1 | y(x) = 0, \forall x \in S). \tag{6}$$

In our motivation application, using the Bayesian classification model described in section 3, we can formulate (6) as a MINLP. First we write equation (4) as

$$\mathbb{P}\left(Y(x) = 1 | \theta\right) = \frac{\prod_j \eta_j(x_j)}{\prod_j \eta_j(x_j) + \frac{\mathbb{P}(Y(x)=0)}{\mathbb{P}(Y(x)=1)}}, \tag{7}$$

where

$$\eta_j(x_j) = \frac{\theta_{1,j}(x_j)}{\theta_{0,j}(x_j)} \text{ for } \forall j \in \{1, \ldots, L\}.$$

Then we can write equation (6) as

$$\underset{e\in E\setminus S, f(e)<b}{\arg\max} \frac{\prod_j \eta_j(e_j)}{\prod_j \eta_j(e_j) + \frac{\mathbb{P}(Y(e)=0)}{\mathbb{P}(Y(e)=1)}}, \tag{8}$$

where

$$\eta_j(e_j) = \frac{\mathbb{P}(e_j|Y(e)=1, Y(x)=0, \forall x \in S)}{\mathbb{P}(e_j|Y(e)=0, Y(x)=0, \forall x \in S)}.$$

Now we can formulate equation (8) as a MINLP,

$$\begin{aligned} \max \quad & \frac{\prod_j \Sigma_k x_j(k)\eta_j(k)}{\prod_j \Sigma_k x_j(k)\eta_j(k) + \frac{\mathbb{P}(Y(x)=0)}{\mathbb{P}(Y(x)=1)}} \\ \text{s.t} \quad & k \in \{1, \ldots, K\} \\ & x_j(k) \in \{0,1\} \\ & \Sigma_k x_j(k) = 1, \end{aligned} \tag{9}$$

where

$$x_j(k) = \begin{cases} 1 & \text{if } e_j = k \\ 0 & \text{else.} \end{cases}$$

We summarize the algorithm in the appendix.

4.3 Expected Improvement

The formulation for expected improvement is similar to probability of improvement, and for our application, we can also formulate (5) as a MINLP.

Proposition 2. *If the objective function g is* EI*, we can write* (5) *as*

$$\underset{e\in E\setminus S}{\arg\max}\, c_0 \mathbb{P}_0(e)(b - f(e))^+ + \sum_{i=1}^{|S|} c_i \mathbb{P}_i(e)(f(x_i) - f(e))^+, \tag{10}$$

where

$$\begin{aligned} &\mathbb{P}_0(e) = \mathbb{P}(y(e)=1 | y(x)=0, \forall x \in S), \\ &\mathbb{P}_i(e) = \mathbb{P}(y(e)=1 | y(x_i)=1, y(x_j)=0, \forall j < i, x_i, x_j \in S), \end{aligned}$$

and $c_i (i = 0, \ldots, |S|)$ are known coefficients.

Note that each term in the summation of (10) has a similar structure as (6). Let $S = \{p^1, \ldots, p^{|S|}\}$, we can write (10) as a MINLP:

$$\begin{aligned} \max \quad & \Sigma_{i=0}^{|S|} c_i \frac{\prod_j \Sigma_k x_j(k)\eta_j^i(k)}{\prod_j \Sigma_k x_j(k)\eta_j^i(k) + \frac{\mathbb{P}(Y(x)=0)}{\mathbb{P}(Y(x)=1)}} (f_i - f(e))^+ \\ \text{s.t} \quad & k \in \{1, \ldots, K\} \\ & x_j(k) \in \{0,1\} \\ & \Sigma_k x_j(k) = 1, \end{aligned} \tag{11}$$

where

$$x_j(k) = \begin{cases} 1 & \text{if } e_j = k \\ 0 & \text{else,} \end{cases}$$

$$f_i = \begin{cases} b & \text{if } i = 0 \\ f(p^i) & \text{else,} \end{cases}$$

and c_i's are known coefficients. We summarize the algorithm in the appendix.

5 Model Validation and Performance

In this section we evaluate the performance of our proposed optimization methods by comparing it with two benchmark methods, in simulation, and by using it in practice. We show our algorithm helped find a peptide with the desired property that is shorter than the shortest previously known.

Comparison Methods We compare our proposed algorithm to two benchmark methods that our biochemist collaborators originally considered using: one is a typical biological evolution approach, that is, mutate known peptides with positive label, and take N peptide candidates with highest predicted probability being positive for testing. We call it mutation method. The other method is to use the trained classifier to predict probability being positive for all peptides in the library, and select N most probable ones for testing. We call it ranking method. We compare their performance below.

Simulation Results with Historical Data The benchmark was performed on data from previously performed experiments. We use data set #0 and #1 to train the statistical model, and based on this model, we generate peptides for testing using our proposed algorithm and the other two comparison methods. Then we train another statistical model using data set #0, #1 and #2, and we evaluate the performance of the methods using both trained models.

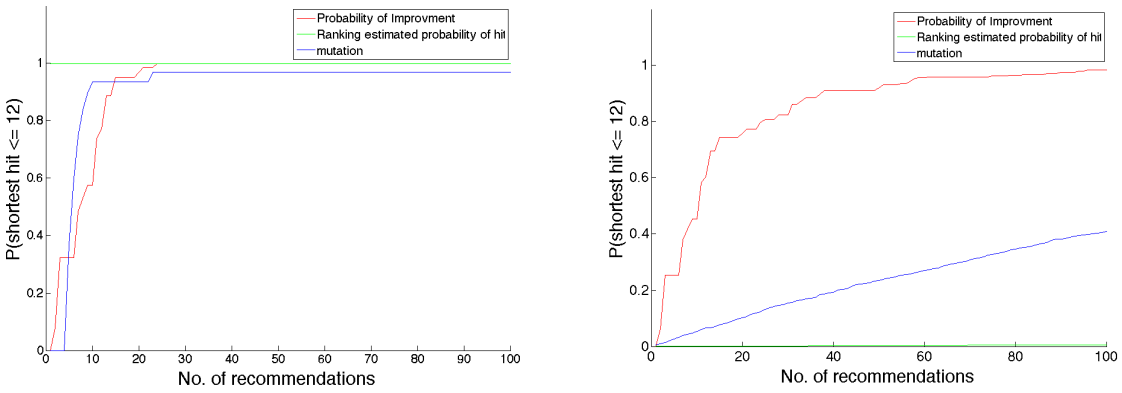

Figure 2: Benchmark of Probability of Improvement algorithm

In Figure 2 we show the benchmark using P* as performance metric. The plot to the left is calculated using the model trained on data set #0 and #1, and the plot to the right is obtained using the model trained on all three datasets. We notice that the performance of ranking method

changes drastically across the two plots.This is because the pure exploitation behavior of ranking method makes it heavily dependent on the accuracy of underlying model. If you recall the ROC curve in Figure 1, it shows that the statistical model is not very accurate considering limited size of training data. Since ranking method finds the ``best'' peptides based on the current model, which are likely to be very similar to each other, in reality, if one of these peptides is not hit, then all the others are likely to be nonhits, and this is the case shown in the plot to the right. On the other hand, Probability of Improvement algorithm performs consistently well, especially in the plot to the right, where the performance is calculated using the model that the algorithm has not ``seen'', it outperforms the other two methods significantly. The reason is that the Probability of Improvement algorithm is designed to make the best decisions given the model is not perfect and tends to explore the space while generating probable hits. The diversity of the recommendation set generated by this algorithm ensures its good performance even when the underlying model is not good.

We also plan to show the benchmark using EI as performance metric in the furture.

Performance in Practice Given the preliminary data, we have performed two rounds of experiments using recommendations generated by Probability of Improvement algorithm. We show the histogram of length distribution of peptide hits in Figure 3 and explain how the algorithm worked to find a shorter peptide hit than the shortest peptide previously known: Ybbr sequence. Observe that all the new peptide hits found have length less than 20, this is because we set the target length $b = 20$ for our algorithm, which made the algorithm only generate peptide candidates with length less than 20. We did not set target length as the shortest length previously known, because we only have 2 short peptides from the preliminary data set and all the others have length greater than 40. Our goal for the first two rounds of experiments was to find a number of reasonably short peptide hits. After one round of experiment, we found additional 38 peptide hits, with most of them having length 19 and a few short peptides. This is expected because we have more information of long peptides than that of short peptides from the preliminary data set. To generate the second round of recommendations, we trained the statistical model using both preliminary data set and data from the first round of experiment, which contain more information on the short peptides, and we hoped to get more short peptide hits. As expected, after the second round of experiment, we can see from the histogram that there are many more short peptide hits found in this round, and we even found a peptide hit with length 10, which is shorter than ybbR sequence! We conclude that Probability of Improvement algorithm did a great job in finding short desired peptides.

6 Conclusion

We proposed an optimal search algorithm based on greedy heuristic for solving the active learning problem motivated by finding short peptide substrates problem, and proved that the proposed algorithm guarantees to achieve at least a factor (1-1/e) of the optimal value. From benchmark results, we further showed that the proposed algorithm outperformed the other two heuristic search methods. In addition to theoretical results, we demonstrated effectiveness of our method in practice: so far we have performed two rounds of evaluations by doing experiments. The first round of experiments tested 258 peptides recommended by our algorithm, which is based on the statistical classifier trained using 36 preliminary data points, and the experiments took about 2 weeks to complete. As demonstrated in Figure 3, we found 38 new peptide hits with shortest length 11. A month later, the second round of experiments tested 580

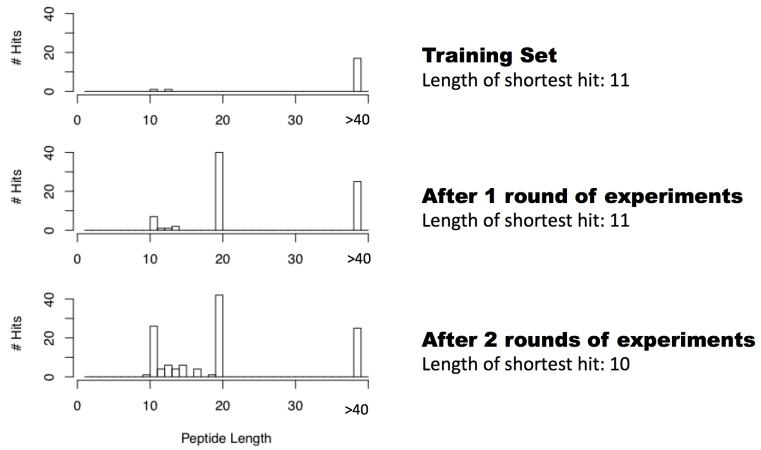

Figure 3: Length distribution of target peptides.

peptides and took another 2 weeks, in which we found 43 new hits and a peptide hit with length 10, which is shorter than the shortest previously known.

References

Chen, Y., Krause, A., and Zurich, E. T. H. (2013). Near-optimal Batch Mode Active Learning and Adaptive Submodular Optimization. 28.

Ginsbourger, D., Le Riche, R., and Carraro, L. (2008). A Multi-points Criterion for Deterministic Parallel Global Optimization based on Gaussian Processes. Technical report.

Hoi, S. C. H., Jin, R., and Lyu, M. R. (2006a). Large-scale text categorization by batch mode active learning. *Proceedings of the 15th international conference on World Wide Web - WWW '06*, page 633.

Hoi, S. C. H., Jin, R., Zhu, J., and Lyu, M. R. (2006b). Batch mode active learning and its application to medical image classification. *Proceedings of the 23rd international conference on Machine learning - ICML '06*, pages 417--424.

Jones, D., Schonlau, M., and Welch, W. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, pages 455--492.

Kosa, N. M., Haushalter, R. W., Smith, A. R., and Burkart, M. D. (2012). Reversible labeling of native and fusion-protein motifs. *Nature Methods*, 9(10):981--984.

Negoescu, D. M., Frazier, P. I., and Powell, W. B. (2011). The knowledge-gradient algorithm for sequencing experiments in drug discovery. *INFORMS Journal on Computing*, 23(3):346--363.

Nemhauser, G., Wolsey, L., and Fisher, M. (1978). An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265--294.

Zhou, Z., Cironi, P., Lin, A. J., Xu, Y., Hrvatin, S., Golan, D. E., Silver, P. A., Walsh, C. T., and Yin, J. (2007). Genetically encoded short peptide tags for orthogonal protein labeling by sfp and acps phosphopantetheinyl transferases. *ACS Chemical Biology*, 2(5):337--346.