
ITERATIVE PEPTIDE MODELING WITH ACTIVE LEARNING AND META-LEARNING

A PREPRINT

Rainier Barrett

Department of Chemical Engineering
University of Rochester

Andrew D White*

Department of Chemical Engineering
University of Rochester

September 2019

ABSTRACT

Often the development of novel materials is not amenable to high-throughput or purely computational screening methods. Instead, materials must be synthesized one at a time in a process that does not generate significant amounts of data. One way this method can be improved is by ensuring that each experiment provides the best improvement in both material properties and predictive modeling accuracy. In this work, we study the effectiveness of active learning, which optimizes the order of experiments, and meta learning, which transfers knowledge from one context to another, to reduce the number of experiments necessary to build a predictive model. We present a novel multi-task benchmark database of peptides designed to advance active, few-shot, and meta-learning methods for experimental design. Each task is binary classification of peptides represented as a sequence string. We show results of standard active learning and meta-learning methods across these datasets to assess their ability to improve predictive models with the fewest number of experiments. We find the ensemble query by committee active learning method to be effective. The meta-learning method Reptile was found to improve accuracy. The robustness of these conclusions were tested across multiple model choices.

Keywords Active Learning · Machine Learning · Meta-Learning · Peptide Modeling

1 Introduction

Although great strides have been made in predictive computational modeling where large datasets exist, the use of computational modeling where data is scarce and/or expensive has been limited. Data expense and scarcity is the norm in materials design.^[1] Computer-aided design of materials often relies on physics-based predictive methods,^[2,3] which require no data, but cannot predict complex properties like activity of a drug or refractive index of a thin film. Even if physics-based modeling is used, there is no clear mechanism to improve predictions as data is gathered in the course of testing materials. Thus, human intuition is often the state-of-the-art for choosing which new materials to test when there are small amounts of data.

Here we apply active learning to binary classification of peptides across a variety of binary tasks like predicting solubility or activity against bacteria. We examine two standard active learning methods: query by committee (QBC) and uncertainty minimization with supervised learning of a deep convolutional neural network. We also examine meta-learning to see if there is gain in transferring knowledge from one task to another. To apply these methods to past data, we only allow our active learners to choose from the dataset of labeled peptides. The methods are evaluated not based on the peptide chosen, but the accuracy of the resulting trained model. The rationale is that there are many competing design constraints in peptide design (e.g., synthetic feasibility, cost, bioavailability, etc) and thus it is better to have an accurate model than a finite set of examples proposed to be active. The goal of this work is to assess active learning and meta-learning as potential ways to improve iterative discovery of peptides in this setting.

* andrew.white@rochester.edu

Active learning has a long history as an extension to design of experiments, which is about choosing the optimal experiments to do with limited resources. Our concern is a sequence of experiments where the results of the previous experiments influence our decision of the next, whereas optimal design of experiments is about choosing the best experiment prior to beginning, and assumes a linear model. Active learning is this process of choosing the next experiment optimally.^[4] It is sometimes called optimal experimental design,^[5] targeted experiment design,^[6] sequential design of hypotheses,^[7] optimal learning,^[8] and artificial intelligence scientific discovery^[9] depending on the goal and problem context.

Active learning is typically formulated as an optimization problem.^[4] Consider N observation pairs x_i, y_i of features and labels, respectively, with $i \in [0, 1, \dots, N]$ indicating the order of observation. Assume that y_i is a class label and x_i is a vector of reals. We have a *task model*, $P_{\theta_i}(y|x)$, that assigns a probability to each class label for a feature x and is defined by parameters θ_i , which are updated after each new observation. In this work P_{θ} is a deep-learning convolutional neural network. θ_i is updated according to some training procedure after a new x_i, y_i pair is observed. In active learning, we choose x_{i+1} from our fixed dataset of x, y pairs according to

$$x_{i+1} = \operatorname{argmax}_x A_{\psi}[x, P_{\theta_i}(y|x)] \quad (1)$$

where $A_{\psi}(\cdot)$ is a functional of the task model and possibly x . $A_{\psi}(\cdot)$ can be defined by parameters ψ , although it is normally fixed. For example, $A_{\psi}(\cdot)$ could be the most uncertain point, which gives

$$x_{i+1} = \operatorname{argmax}_x [1 - P_{\theta_i}(\hat{y}|x)] \quad (2)$$

where \hat{y} is the most likely class label for x .^[10] $A_{\psi}(\cdot)$ is called the acquisition function or utility function depending on the problem setting.^[4]

Within this framework, there are a variety of approaches depending on the form of the task model and utility function. If the task model is probabilistic, like above, utility functions which maximize information gain,^[5,11] reduce model uncertainty,^[12] maximize expected model change,^[4] or reduce model variance^[13] can be chosen. Within variance reduction methods, there are so-called A-optimal,^[13] D-optimal,^[13,14] and E-optimal^[15] approaches which minimize the covariance matrix according to different assumptions. If the task model is non-parametric, Bayesian approaches are well-suited.^[6] One still has a choice of utility function and can, for example, maximize the expected model improvement at each experiment.^[14,16] Bayesian approaches can work with recent deep learning methods through Bayesian convolutional neural networks^[17] or through the hypothesized connection between dropout and neural network uncertainty.^[18]

If the task model is not stochastic but there is flexibility in parameter choice or multiple models to choose from, then there are a variety of pooled or consensus active learning approaches. QBC is an active learning approach that maintains a committee of models and chooses the next experiment based on where the committee disagrees most.^[19] There are also active learning pool methods for specific model types. For example, k-nearest neighbor,^[20] logistic regression,^[21,22] and linear regression with noise.^[23] One can also treat the choice of model as a probability distribution and then the pool of models can be viewed as a stochastic or Bayesian model to use any of the previous utility functions.

There are active learning methods that are independent of the task model and instead find a characteristic set of data.^[24] This can be done by clustering the data, optimizing a function which describes local variance,^[25] finding regions of high uncertainty,^[26] or by finding a “coreset” characteristic subset of data via submodular function optimization.^[20,27] These methods are closely related to semi-supervised learning, which tries to use unlabeled data to improve a model when labeled data is sparse.^[28,29]

Another closely related topic to active learning is Bayesian optimization or global optimization of black box functions. There the goal is to optimize a function while evaluating it a minimum number of times. To connect this to active learning described above, view the “expensive function” as the experiment. A surrogate stochastic model is constructed, often through bootstrapping or non-parametric models, and that surrogate model is used within an active learning framework to reduce the number of the function evaluations.^[30] Then active learning is applied so that each function evaluation improves the maximum function value and/or understanding of the model. This method can be equivalent to the variance reduction techniques discussed above if the same utility functions are chosen.^[31] More sophisticated active learning methods can be used with the surrogate model, including Gaussian process regression and complex portfolios of acquisition (utility) functions.^[32,33] A common critique of Bayesian optimization is that it typically scales between $O(n^2)$ to $O(n^3)$, depending on approximations made, and struggles with high-dimensional surrogate models. However, this is not applicable here, since our goal is learning in systems with dozens of experiments, not thousands. A recent overview on the application of Bayesian optimization to materials design can be found in Frazier and Wang.^[34]

An early example of active learning in chemistry can be found with van de Walle et al.,^[35] where a phase diagram was explored using variance minimization active learning on cluster expansions. Active learning works well in general with choosing cluster expansions via variance minimization.^[36,37] More recently, Lookman et al.^[38] explored elastic properties with ab initio calculations using a Bayesian optimization technique (Efficient Global Optimization) to optimize the ratio of three metals. The authors applied the same method to design new piezoelectrics^[39] with a four dimensional design space. Gopakumar et al.^[8] also showed how active learning methods that balance exploration and exploitation can work well on 2 and 3 dimensional materials systems. This active learning approach to finding compositions with Bayesian optimization is quickly gaining popularity in the material informatics community for low-dimensional systems.^[40-42]

Kim et al.^[43] explored active learning methods to find polymers with a specific glass transition temperature. This is a high-dimensional system because the polymers are represented with a variety of descriptors. It was made tractable by treating a list of 731 possible polymers as known a priori. At each step the model evaluates all 731 possible polymers. An earlier example using a fixed molecule set can also be found in Warmuth et al.^[44] where candidate drug molecules were selected from a vendor catalogue with a variance minimization active learning algorithm.

To avoid this difficulty of high-dimensionality and ambiguity of representing molecular structures as vectors (although see new work from Sanchez-Lengeling and Aspuru-Guzik^[45], Blaschke *et al.*^[46], Gómez-Bombarelli *et al.*^[47]), this work focuses on peptides. Peptides are a natural fit as targets for experimental design driven by machine learning, because they draw from the finite alphabet of 20 natural amino acids. They can be readily encoded as discrete vectors, as opposed to the less well-defined space of all possible valid chemical structures for other purposes, for which more specialized methods like autoencoders can be necessary. The easily-encoded amino acid sequence representation of peptides also has a convenient parallel in existing machine learning methods for natural language processing, which is already a developed sub-field of machine learning from which modeling techniques can be leveraged.^[48]

Recently Tallorin *et al.*^[49] explored the use of Bayesian optimization for *de novo* design of peptide substrates. Their work is similar in that both this work and theirs used a sequence model with the goal of minimizing the number of experiments required to optimize a peptide. The differences are that they were modeling with regression, allowed for complete choice in sequence space, did not have a goal of creating accurate models, and did not use a deep learning model but a Naïve Bayes classifier. Their work is an experimentally validated demonstration that intelligently designing experiments with predictive models can reduce the required number peptides that need to be synthesized and tested.

Another topic explored in this work is meta-learning. Meta-learning is a technique for improving few-shot learning across multiple tasks. The goal, in our nomenclature above, is to make the task model depend on hyperparameters ξ that are trained to work well across multiple tasks. Then on a new task, using $\hat{\xi}$, as few new examples are required to improve performance. Active learning and meta-learning are connected because both are concerned with maximizing the value of data. As the goal is to minimize the number of experiments required for new systems, we find it natural to consider this method on our dataset. Examples of meta-learning for accelerating task models can be found in transfer learning,^[50] few-shot learning,^[51-53] and automated machine learning.^[54] One application that has recently connected active learning and meta-learning is in model free reinforcement learning.^[55,56] Pang *et al.*^[57] and Fang *et al.*^[58] have also explored the connection between active learning and meta-reinforcement learning.

2 Methods

Five past published databases were used to generate training datasets in this work: the Antimicrobial Peptide Database (APD),^[59] the collection of protein surface fragments from Barrett *et al.*^[60], the PROSO II database,^[61] library hits with activity against TULA-2 protein,^[62] and library hits with activity against SHP-2.^[63] The APD contains peptides flagged with a variety of activities, such as antibacterial, antifungal, antiviral, anticancer, and hemolytic activities. The PROSO II database contains peptides and proteins categorized as soluble or insoluble. The TULA-2 and SHP-2 libraries are fixed-width peptides optimized for binding to a specific target. Eight datasets were chosen from the APD: 1. “antibacterial,” 2. “anticancer,” 3. “antifungal,” 4. “antiHIV,” 5. “antiMRSA,” 6. “antiparasital,” 7. “antiviral,” and 8. “hemolytic.” All sequences above length 200 were excluded. These wide variety of tasks represent a range of modeling goals in peptide research.

The task model is a deep convolutional neural network classifier, necessitating negative training examples as well as positive. One corresponding negative training data set was generated for each positive data set. Two types of negative data were generated: fake scrambled data with amino acid distributions identical to the positive dataset and samples from datasets which are expected to have no intersection with positive examples. These are expected to be rather challenging negative examples, since scrambled peptides likely have many physical properties of the positive examples. The non-intersecting datasets are also, generally, naturally occurring peptides that likely are biologically

Positive Dataset	Size	Negative Datasets
antibacterial	2079	shp2 ¹ , tula2, insoluble ² , antifungal ³ , antiHIV, anticancer, fake
anticancer	183	shp2, tula2, insoluble, antifungal, antiHIV, antiparasital, antibacterial, fake
antifungal	891	shp2, tula2, insoluble, antiHIV, anticancer, antibacterial, fake
antiHIV	87	shp2, tula2, insoluble, antifungal, anticancer, antiparasital, antibacterial, fake
antiMRSA	119	shp2, tula2, insoluble, antiHIV, anticancer, antiparasital, fake
antiparasital	90	shp2, tula2, insoluble, antiHIV, anticancer, fake
antiviral	150	shp2, tula2, insoluble, antifungal, anticancer, antiparasital, antibacterial, fake
hemolytic	253	shp2, tula2, insoluble, human ⁴ , fake
soluble	7028	insoluble ⁵
shp2	120	fake ⁶
tula2	65	fake ⁶
human	2880	insoluble, hemolytic, fake

Table 1: The positive and negative examples chosen for training the classifiers. Negative datasets were sampled to be the same size as the positive datasets. ¹ It is exceedingly rare to find antibacterial peptides, so it is assumed that the SHP-2/Tula-2 binding peptides are not good examples of antibacterial peptides. ² Insoluble peptides cannot be successful antibacterial peptides. ³ Antifungal and antibacterial activity are different mechanisms, so it is unlikely that a given peptide is both antifungal and antibacterial. ⁴ it is assumed that fragments of proteins found on surfaces of proteins found in humans are not hemolytic (kill red blood cells). ⁵ No fake scrambled dataset is necessary because there are known negative examples. ⁶ classifying SHP-2 and Tula-2 is in the context of fixed-length peptides so only scrambled peptides with the same length are used for classification.

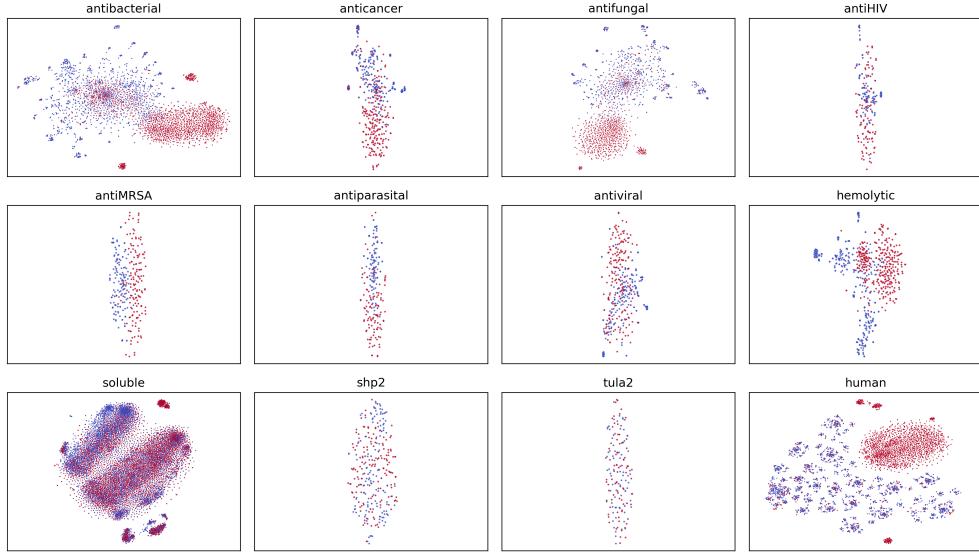


Figure 1: t-SNE projection of the dataset used in this work. The sequences were padded with zeros to be all length 200 and flattened to be a 200×20 dimensional point, which was projected with PCA to 40 dimensions and then embedded with t-SNE. The figure shows that the fake/scrambled datasets show significant overlap with their counterparts. Some of the datasets are seen to be easily separable as well, like the insoluble from the antibacterial, for example.

relevant. To generate the fake scrambled negative data set, a number of peptides were generated randomly, with lengths sampled from the same length range, and residues sampled from the same frequency distribution as their respective positive set. The non-intersecting examples are shown in Table 1 along with rationale in the caption. To balance classes, the negative datasets were sampled down to be the same size as the positive example. A t-SNE projection of the peptides in this work are shown in Figure 1.

Peptide sequences are encoded as one-hot vectors of dimension $N \times 20$ (with N the length of the peptide), where each index in the second dimension corresponds to the index of the amino acid in the n th position in the alphabet of amino acids: [A, R, N, D, C, Q, E, G, H, I, L, K, M, F, P, S, T, W, Y, V]. Activity was encoded as a one-hot label vector of length 2, where [1, 0] indicates a positive label and [0, 1] indicates a negative label.

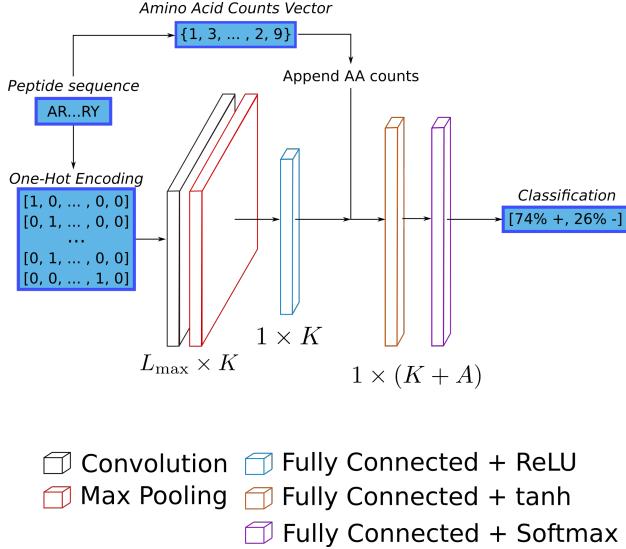


Figure 2: Neural network structure for active learning. Here, L_{\max} is the maximum width of a peptide in the dataset, K is the number of motif classes, A is the length of the amino acid alphabet, and N_{desc} is the number of chemical descriptors used. Peptides are first translated to a one-hot encoding ($L_{\max} \times A$) and a descriptors vector ($1 \times N_{\text{desc}}$), and the total number of occurrences of each amino acid is counted ($1 \times A$). The output of the max pool layer is passed through one fully connected layer with ReLU activation, then descriptors and amino acid counts are appended to the output. This is then passed into two more fully connected layers with a final output dimension of 2, for positive and negative class labels. Labels below neural network layers indicate the dimensionality of the data as it passes through the layer.

The task model structure is a convolutional neural network partially inspired by past work in peptide modeling.^[64] The model structure is shown schematically in Figure 2. The first layer of the neural network is a convolutional layer with a weight matrix of dimension $[W \times A \times K]$, where W and K conceptually represent peptide “motif widths” and number of “motif classes,” respectively, and A is the length of the amino acid alphabet considered (20 for the naturally-occurring peptides used here). The next layer is a max pool layer, which captures which “motif class” is most likely in the input peptide by pooling across the peptide’s length dimension, leaving a $1 \times K$ vector.

The output of the max pool layer is concatenated with the relative frequency of each amino acid in the input peptide (1×20 vector) and standardized (0-1) sequence length. This is then fed to three fully-connected (FC) layers with ReLU activation function and then to one FC layer with softmax activation for classification, ensuring the outgoing vector of size 2 adds up to 1, since this vector is meant to represent the likelihood of assignment to the positive or negative classes. The final output is compared with the true label vector (classification) or the true activity (regression), and loss is calculated as the cross-entropy between the two. The minimization algorithm used during training was TensorFlow’s^[65] Adam optimizer with parameters recommended in Kingma and Ba^[66] and a learning rate of 0.001.

The method of uncertainty minimization is designed to favor exploration to maximize information gained per training iteration of a model. This is achieved by choosing a new training point based on some measure of the uncertainty of the machine learning model used. In this sense, it is well-aligned with an eventual goal of automated experiment selection, because it can minimize the number of necessary experiments to characterize a property space well. This, in turn, would lead to a reduction in operating costs and time spent performing experiments.

The training procedure for this active learning method is shown schematically in Figure 3. The model described previously was used with W and K chosen as 5 and 6, respectively. Model weights are randomly initialized and then the model uncertainty is calculated as the variance of the output vector of the neural network for each peptide. One peptide is then sampled, with probabilities of selection for each peptide weighted by their respective variances under the current model parameters, i.e. $p(1 - p)$. This is different than Equation 1, where argmax is used instead of sampling. The chosen peptide is then used to train the neural network for one training iteration, then an additional 5 training iterations are performed with batch size 5, sampling uniformly from all previously-observed peptides for training input. Thus, in the first iteration the first point is used for 25 training steps, then in the second, the first and second points are used an expected value of 12.5 times each, etc. This process is repeated for 25 training epochs for one training run. To gather statistics, 30 training runs of 25 epochs each were performed for each dataset.

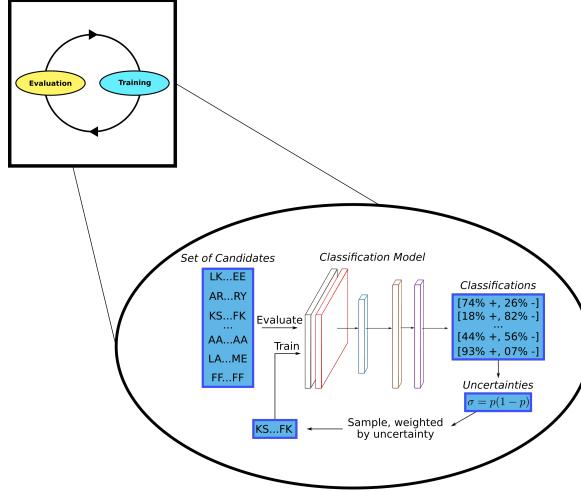


Figure 3: Training procedure for uncertainty minimization active learning method. All candidate peptides are evaluated under the current model parameters, and their uncertainty is calculated as the variance, $p(1 - p)$, of the resultant output vector. These uncertainties are then normalized and used as probabilities to randomly select one peptide for training.

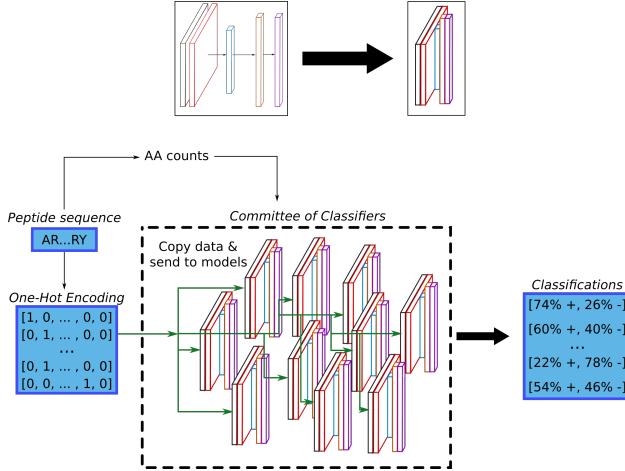


Figure 4: Diagram of the query by committee model structure. The model from Figure 2 is duplicated with 10 different convolution weight matrix dimensions and trained in the same way. Average uncertainty among all committee models is used to sample new training points.

The QBC method employed in this work uses the same approach as uncertainty minimization, but instead of a single task model that is trained and used for selection, a committee of 10 models is used. The 9 additional models are all structured in the same way as the model used in uncertainty minimization, but use different hyperparameters. They differ in the dimensions of the weights matrix used in the convolution layer, having all combinations of $3 \leq w \leq 5$ and $3 \leq k \leq 5$ along with the $w = 5, k = 6$ model used in uncertainty minimization. In QBC, input data is passed to each committee member (task model), and each one produces a prediction. The average variance among all models is used as sampling weight for selecting a new training point, and training is performed in the same way as for uncertainty minimization, with the same number of epochs and training runs.

To compare these active learning methods against a base case, we use two control training methods with the same model as used for the uncertainty minimization method. The first control is a “baseline” Adam training, where the model trains on all peptides for 5,000 steps with a batch size of 32. The second is a more direct comparison to the active learning methods called “random” where peptides are chosen randomly the model is trained in the same way as in the active learning methods (batch size 5, 5 iterations, 25 epochs).

The meta-learning method used in this work is Reptile from Nichol *et al.*^[67]. It is related to the the work by Finn *et al.*^[68], called model agnostic meta-learning. In our notation, the goal of meta-learning is to optimize the initial parameters of the task model, θ_0 , to work well given a sampled task τ . $P(\tau)$ is taken to be uniform across our datasets. θ_0 is optimized by Adam optimization of a meta-objective function:

$$\mathbb{E}_{P(\tau)} [\mathcal{L} [U(\psi, \theta_0, \tau)]] \quad (3)$$

where τ is the dataset corresponding to a set (x, y) , ψ are the parameters defining the active learning method $A_\psi(\cdot)$, and θ_0 is the given initial task model parameters. \mathcal{L} is the usual loss function and U is a stand-in for doing J steps of active learning training with $A_\psi(\cdot)$. The gradient of this meta-objective requires a Hessian, but Reptile approximates this with a Taylor expansion. In this work $J = 5$, meaning we train with active learning on 5 peptides each time with a batch size of 5. 2,500 meta-learning iterations were done and then early stopping was used to prevent overfitting. This was done on an 80% split of the left-out dataset and then results were reported on the 20% remaining sequences of the left-out dataset.

AUC values and accuracies are reported on withheld data which was 20% of the dataset size. In the meta-learning results, the accuracies and AUCs reported had the dataset withheld from training. During meta-learning, the location of active/inactive labels (first or second index) was swapped between tasks to prevent over-fitting to active being in one position or another. This gives minimal zero-shot accuracy. The minimized loss was cross entropy between label probabilities and true labels. Error bars and individual traces are different due to split differences of data and random number initial parameter seeds.

3 Results and Discussion

Figure 5 shows the active learning, meta-learning, and baseline results across the datasets for the uncertainty minimization active learning strategy. The baseline results (gray) show that the convolutional neural network provides reasonable results across the range of modeling tasks despite its simple architecture, with the exception of the soluble and Tula-2 tasks. The solubility task is challenging because many of the training examples are actually folded proteins, requiring long-range sequence correlations to model properly. The current state of the art on this dataset is accuracy of 0.77^[69] and the convolutional neural network here has an accuracy of 0.56 (barely above random). The Tula-2 peptide affinity task simply is difficult to predict due to the small data amount ($N = 65$) and diversity of sequences. Figure 5 also shows the comparison of choosing peptides randomly and choosing maximum uncertainty peptides (uncertainty minimization). Uncertainty minimization (blue line) is no better than choosing randomly (dashed green) in general. It is sometimes worse and sometimes better.

To assess the effect of meta-learning on reducing experiment number, it was evaluated both alone and in combination with active learning. Meta-learning results are shown in Figure 5. Meta-learning consistently improves accuracy. This can be seen in the red ML+Random line, which is consistently above the dashed green line (random alone). Combining uncertainty minimization with meta-learning provides no advantage, and mostly decreases accuracy. In many of the datasets, performance approaches baseline levels with only 25 examples, while the baseline is trained on all data. This demonstrates the advantage of using meta-learning.

Receiver operator characteristic (ROC) curves provide an accounting of the balance between type I and type II error. This is important for peptide activity because, due to the large design space, false positives are more detrimental to a model's usefulness. The area under curve (AUC) of a ROC curve gives a scalar number representing the quality of the ROCs. Although note that here we enforced balanced classes. The ROC AUCs are reported in Table 2. As observed in Figure 5, there is no gain by using the uncertainty minimization active learning. Also, 25 examples is not enough to match the baseline models without meta-learning. The source of variance in this work is because there are many ways to choose 25 peptides from the datasets and the datasets are small. Significant exceptions are the Tula-2, SHP-2, and soluble datasets, which show poor performance with limited examples relative to baseline models. In particular, SHP-2 seems to require the full dataset to achieve good accuracy. This may be due to the importance of motifs in this dataset.^[70]

The QBC results are reported in Table 2 and Figure 8. QBC provides consistently better performance over random choice and uncertainty minimization. QBC further improves with meta-learning. QBC seems to have better accuracy in general and the best performance when combined with meta-learning.

Overall, meta-learning improves learning across these data, especially when combined with QBC. Recent analysis of meta-learning has shown mixed results across tasks. Raghu *et al.*^[71] showed that model agnostic meta-learning methods like Reptile only learn to re-use features across tasks. To assess how feature re-use can be applicable in these dataset, we performed a basic sensitivity analysis in Figure 6 which gives insight into the various features used in the

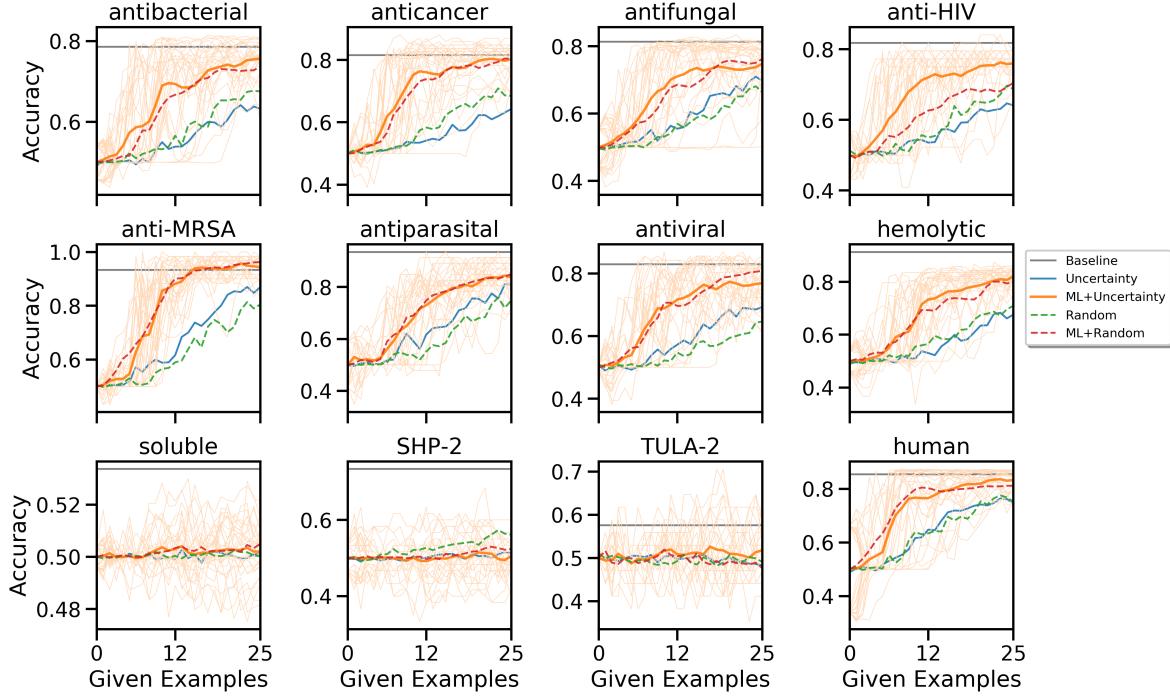


Figure 5: Training curves of uncertainty minimization active learning compared with baseline (gray) trained across all data points and randomly choosing examples. The y-axis is accuracy on withheld data. Light traces are individual 30 runs and the dark trace is median (only one set of traces is shown). Each run has a different train/withheld split and random number generator seeds. Each subplot is a different task.

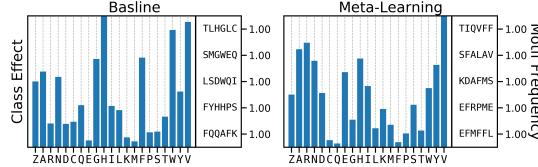


Figure 6: Features across tasks for baseline model. The barplot shows the partial derivative of probability of activity with respect to amino acid count averaged across training data. This gives importance for assigning label. The right side of the plots shows the maximum magnitude weight in the convolution, which roughly corresponds to most attended motif in the sequence. The y-axis label on the right side shows the frequency of this motif across 25 training.

modeling. Figure 6 shows the features for the baseline model on the antibacterial dataset. The zero-shot features for meta-learning are shown in Figure 6. Note that all motif frequencies are 1.0 since this is the meta-learned parameters, not a realization of training. The results show that meta-learning does not have the exact features found in the baseline model, although many of the important amino acids are shared (N, W, C, H). Some of the amino acids within the motifs are shared, but they are not identical.

To ensure our conclusions about meta-learning and QBC being preferred for peptide design are robust, we explored three different model choices. These are shown in Figure 7 for only the antibacterial task (although meta-learning traces were trained on all but antibacterial). The first subplot is ablation of the motif convolutions. Without the motifs, accuracy is reduced a small amount and the advantage of QBC and meta-learning disappears. Switching from a cross-entropy loss to absolute error loss reduces accuracy and the advantage of QBC over random choice. Meta-learning is still advantageous. The last plot shows if we remove the label swapping, which is used to reduce over-fitting to “active” peptides. Zero-shot performance is improved, due to good correlation of activity in peptides across tasks. The QBC method has an improved margin, likely because it has more learnable parameters. Meta-learning is preferred in this setting, when it is known that the class labels of active vs inactive can be re-used across tasks.

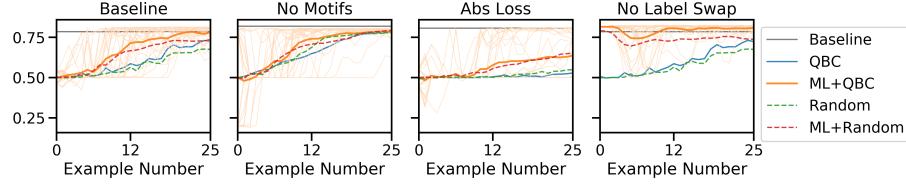


Figure 7: Training curves for different model choices. Baseline is the same in each panel and the baseline subplot is the model as presented in text. No motifs has the convolution layers removed, abs loss uses an absolute difference loss instead of cross-entropy, and no label swaps means that labels weren't swapped during meta-learning so that zero-shot accuracy is maximized. The results show that meta-learning is not always better, but is consistently a good choice.

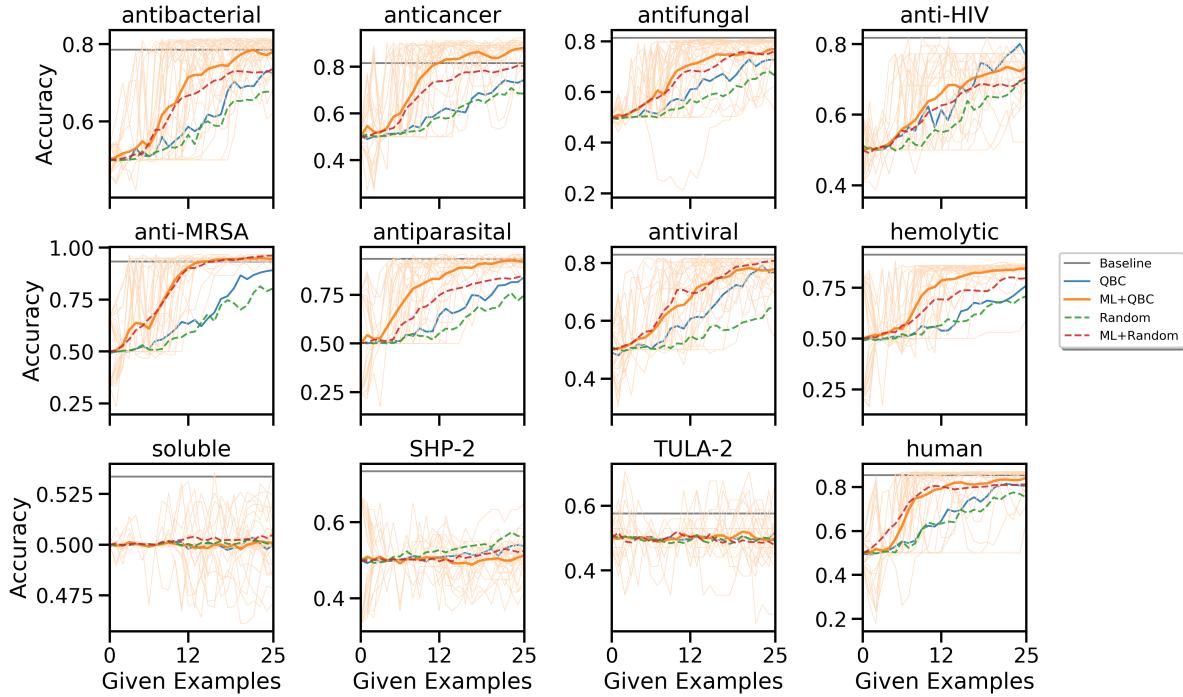


Figure 8: Training curves of uncertainty minimization active learning combined with Reptile meta-learning compared with baseline (gray) trained across all data points and meta-learning trained with randomly-chosen examples. The y-axis is accuracy on withheld data. Light traces are the 250 individual runs and the dark trace is the median. Each run has a different train/withheld split and random number generator seeds. Each subplot is a different task which was withheld during meta-learning. Meta-learning offers inconsistent improvements.

	Baseline	Random	QBC	Umin	ML+Random	ML+QBC	ML+Umin
Antibacterial	0.84	0.81 ± 0.02	0.82 ± 0.04	0.79 ± 0.07	0.81 ± 0.01	0.82 ± 0.01	0.83 ± 0.01
Anticancer	0.86	0.83 ± 0.04	0.83 ± 0.03	0.76 ± 0.04	0.88 ± 0.04	0.92 ± 0.02	0.86 ± 0.03
Antifungal	0.84	0.81 ± 0.04	0.83 ± 0.06	0.81 ± 0.07	0.84 ± 0.01	0.80 ± 0.02	0.83 ± 0.01
Anti-HIV	0.89	0.85 ± 0.03	0.89 ± 0.05	0.76 ± 0.07	0.76 ± 0.05	0.79 ± 0.04	0.84 ± 0.03
Anti-MRSA	0.97	0.90 ± 0.02	0.96 ± 0.03	0.97 ± 0.02	0.98 ± 0.01	0.96 ± 0.02	1.00 ± 0.01
Antiparasital	0.96	0.88 ± 0.04	0.95 ± 0.03	0.93 ± 0.03	0.90 ± 0.03	0.95 ± 0.02	0.91 ± 0.03
Antiviral	0.76	0.78 ± 0.08	0.86 ± 0.03	0.84 ± 0.07	0.89 ± 0.02	0.81 ± 0.03	0.84 ± 0.04
Hemolytic	0.98	0.82 ± 0.05	0.82 ± 0.04	0.81 ± 0.04	0.89 ± 0.05	0.85 ± 0.10	0.89 ± 0.03
Soluble	0.59	0.50 ± 0.01	0.50 ± 0.02	0.50 ± 0.02	0.51 ± 0.02	0.50 ± 0.02	0.51 ± 0.02
SHP-2	0.75	0.61 ± 0.07	0.55 ± 0.10	0.52 ± 0.07	0.54 ± 0.06	0.51 ± 0.07	0.52 ± 0.08
TULA-2	0.65	0.51 ± 0.12	0.51 ± 0.10	0.48 ± 0.09	0.48 ± 0.07	0.50 ± 0.10	0.55 ± 0.08
Human	0.87	0.86 ± 0.01	0.86 ± 0.01	0.85 ± 0.01	0.87 ± 0.01	0.86 ± 0.04	0.87 ± 0.01

Table 2: Area under curve (AUC) for receiver operator characteristic curves for classifiers trained with different active learning methods on different datasets. Baseline was trained on all data, whereas others saw 10 peptides according to their active learning strategy. Errors are computed from standard deviation across 30 trials on different data splits and random sampling in active learning strategy. Umin is uncertainty minimization, ML is meta-learning, and QBC is query by committee.

4 Conclusions

This work has explored active learning and meta-learning strategies for predicting peptide activity on a dataset of 12 different peptide activity tasks. The simple deep convolutional neural networks used here offer reasonable performance across the tasks. We expect more complex models with attention, more layers, and long-range interactions in sequence space could improve the accuracy. Active learning strategies provided improvements over sampling peptides randomly: around a 3-5% increase in accuracy after training the classifier on 25 examples. Meta-learning was found to improve accuracy. These conclusions were found to be robust across loss choice, model structure, and whether or not zero-shot learning is being optimized. This work provides a new peptide multi-task dataset and benchmark results for standard active learning and meta-learning methods.

5 Acknowledements

This material is based upon work supported by the National Science Foundation under grants 1764415 and 1751471.

References

- [1] Maxwell L. Hutchinson, Erin Antono, Brenna M. Gibbons, Sean Paradiso, Julia Ling, and Bryce Meredig. 2017. Overcoming data scarcity with transfer learning. In 31st Conference on Neural Information Processing Systems. Long Beach, CA, USA.
- [2] Xinjian Shi, Samira Siahrostami, Guo-Ling Li, Yirui Zhang, Pongkarn Chakthranont, Felix Studt, Thomas F Jaramillo, Xiaolin Zheng, and Jens K Nørskov. 2017. Understanding activity trends in electrochemical water oxidation to form hydrogen peroxide. *Nature Communications* 8:701.
- [3] Mohammad Atif Faiz Afzal, Mojtaba Haghhighatlari, Sai Prasad Ganesh, Chong Cheng, and Johannes Hachmann. 2019. Accelerated Discovery of High-Refractive-Index Polyimides via First-Principles Molecular Modeling, Virtual High-Throughput Screening, and Data Mining. *The Journal of Physical Chemistry C* 123:14610–14618.
- [4] Burr Settles. 2011. From Theories to Queries: Active Learning in Practice. In In Active Learning and Experimental Design workshop In conjunction with AISTATS.
- [5] Juliane Liepe, Sarah Filippi, Michał Komorowski, and Michael P H Stumpf. 2013. Maximizing the Information Content of Experiments in Systems Biology. *PLOS Comput. Biol.* 9:1–13.
- [6] Joep Vanlier, Christian A Tiemann, Peter J Hilbers, and Natan A W Van Riel. 2012. A Bayesian approach to targeted experiment design. *Bioinformatics* 28:1136–1142.
- [7] Shelemyahu Zacks. 1996. 5 Adaptive designs for parametric models, volume 13 of *Handbook of Statistics*. Elsevier.

- [8] Abhijith M. Gopakumar, Prasanna V. Balachandran, Dezhen Xue, James E. Gubernatis, and Turab Lookman. 2018. Multi-objective Optimization for Materials Discovery via Adaptive Design. *Sci. Rep.* 8:1–12.
- [9] Bruce Buchanan, Edward A Feigenbaum, and Joshua Lederberg. 1968. Heuristic DENDRAL: A program for generating explanatory hypotheses in organic chemistry. *Mach. Intell.* 4.
- [10] David D Lewis and William A Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIR’94*, pages 3–12. Springer, London.
- [11] Xin Li and Yuhong Guo. 2013. Adaptive Active Learning for Image Classification. In *2013 IEEE Conf. Comput. Vis. Pattern Recognit.*, pages 859–866. IEEE, Portland.
- [12] A J Joshiy, F Porikli, and N Papanikolopoulos. 2010. Multi-class batch-mode active learning for image classification. In *2010 IEEE Int. Conf. Robot. Autom.*, pages 1873–1878.
- [13] David J C Mackay. 1992. Information-Based Objective Functions for Active Data Selection. *Neural Comput.* 4:590–604.
- [14] Kathryn Chaloner and Isabella Verdinelli. 1993. Bayesian experimental design: A review. *Stat. Sci.* 7:473–511.
- [15] Patrick Flaherty, Adam P Arkin, and Michael Jordan. 2005. Robust design of biological experiments. In *Adv. Neural Inf. Process. Syst.* Vancouver, British Columbia, Canada.
- [16] Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. 2007. Active learning with gaussian processes for object categorization. In *2007 IEEE 11th Int. Conf. Comput. Vis.*, pages 1–8. IEEE.
- [17] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep Bayesian Active Learning with Image Data. In *ICML’17 Proc. 34th Int. Conf. Mach. Learn.* 70, pages 1183–1192. JMLR.org, Sydney, NSW, Australia.
- [18] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *ICML’16 Proc. 33rd Int. Conf. Int. Conf. Mach. Learn. - Vol. 48*, volume 48, pages 1050–1059. JMLR.org, New York, NY, USA.
- [19] H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. 1992. Query by committee. In *Proc. fifth Annu. Work. Comput. Learn. theory*, pages 287–294. ACM.
- [20] Kai Wei, Rishabh K Iyer, and Jeff A Bilmes. 2015. Submodularity in Data Subset Selection and Active Learning. In *Proc. 32nd Int. Conf. Mach. Learn.*, pages 1954–1963.
- [21] Steven C H Hoi, Rong Jin, Jianke Zhu, and Michael R Lyu. 2006. Batch mode active learning and its application to medical image classification. In *Proc. 23rd Int. Conf. Mach. Learn.*, pages 417–424.
- [22] Yuhong Guo and Dale Schuurmans. 2007. Discriminative Batch Mode Active Learning. In *Adv. Neural Inf. Process. Syst.* 20, pages 593–600.
- [23] Kai Yu, Jinbo Bi, and Volker Tresp. 2006. Active learning via transductive experimental design. In *Proc. 23rd Int. Conf. Mach. Learn.*, pages 1081–1088.
- [24] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. 1996. Active learning with statistical models. *J. Artif. Intell. Res.* 4:129–145.
- [25] Yi Yang, Zhigang Ma, Feiping Nie, Xiaojun Chang, and Alexander G. Hauptmann. 2015. Multi-Class Active Learning by Uncertainty Sampling with Diversity Maximization. *Int. J. Comput. Vis.* 113:113–127.
- [26] David Cohn, Les Atlas, and Richard Ladner. 1994. Improving generalization with active learning. *Mach. Learn.* 15:201–221.
- [27] Ozan Sener and Silvio Savarese. 2018. Active Learning for Convolutional Neural Networks: A Core-Set Approach. In *Sixth Int. Conf. Learn. Represent.*, pages 1–13. Vancouver, British Columbia, Canada.
- [28] Adrian Corduneanu and Tommi Jaakkola. 2002. On information regularization. In *Proc. Ninet. Conf. Uncertain. Artif. Intell.*, pages 151–158. Morgan Kaufmann Publishers Inc.
- [29] Martin Szummer and Tommi S Jaakkola. 2003. Information regularization with partially labeled data. In *Adv. Neural Inf. Process. Syst.*, pages 1049–1056.
- [30] Donald R Jones, Matthias Schonlau, and William J Welch. 1998. Efficient global optimization of expensive black-box functions. *J. Glob. Optim.* 13:455–492.
- [31] Kathryn Chaloner and Isabella Verdinelli. 1995. Bayesian experimental design: A review. *Stat. Sci.* pages 273–304.
- [32] Matthew D Hoffman, Eric Brochu, and Nando de Freitas. 2011. Portfolio allocation for Bayesian optimization. In *UAI’11 Proc. Twenty-Seventh Conf. Uncertain. Artif. Intell.*, pages 327–336.

- [33] Bobak Shahriari, Ziyu Wang, Matthew W Hoffman, Alexandre Bouchard-Côté, and Nando de Freitas. 2014. An Entropy Search Portfolio for Bayesian Optimization. *arXiv Prepr. arXiv1406.4625* .
- [34] Peter I Frazier and Jialei Wang. 2016. Bayesian Optimization for Materials Design. *Inf. Sci. Mater. Discov. Des.* pages 45–75.
- [35] Axel van de Walle and Gerbrand Ceder. 2002. Automating first-principles phase diagram calculations. *J. Phase Equilibria* 23:348.
- [36] Tim Mueller and Gerbrand Ceder. 2010. Exact expressions for structure selection in cluster expansions. *Phys. Rev. B - Condens. Matter Mater. Phys.* 82:1–6.
- [37] Atsuto Seko and Isao Tanaka. 2011. Grouping of structures for cluster expansion of multicomponent systems with controlled accuracy. *Phys. Rev. B* 83:224111.
- [38] Turab Lookman, Prasanna V. Balachandran, Dezhen Xue, John Hogden, and James Theiler. 2017. Statistical inference and adaptive design for materials discovery. *Curr. Opin. Solid State Mater. Sci.* 21:121–128.
- [39] Ruihao Yuan, Zhen Liu, Prasanna V Balachandran, Deqing Xue, Yumei Zhou, Xiangdong Ding, Jun Sun, Dezhen Xue, and Turab Lookman. 2018. Accelerated Discovery of Large Electrostrains in BaTiO₃-Based Piezoelectrics Using Active Learning. *Adv. Mater.* 30:1702884.
- [40] Taro Fukazawa, Yosuke Harashima, Zhufeng Hou, and Takashi Miyake. 2019. Bayesian optimization of chemical composition: A comprehensive framework and its application to RFe₁₂-type magnet compounds. *Phys. Rev. Mater.* 3:1–15.
- [41] Cheng Wen, Yan Zhang, Changxin Wang, Dezhen Xue, Yang Bai, Stoichko Antonov, Lanhong Dai, Turab Lookman, and Yanjing Su. 2019. Machine learning assisted design of high entropy alloys with desired property. *Acta Mater.* 170:109–117.
- [42] Jeffrey M. Rickman, Turab Lookman, and Sergei V. Kalinin. 2019. Materials informatics: From the atomic-level to the continuum. *Acta Mater.* 168:473–510.
- [43] Chiho Kim, Anand Chandrasekaran, Anurag Jha, and Rampi Ramprasad. 2019. Active-learning and materials design: The example of high glass transition temperature polymers. *MRS Commun.* pages 1–7.
- [44] Manfred K Warmuth, Gunnar Rätsch, Michael Mathieson, Jun Liao, and Christian Lemmen. 2002. Active learning in the drug discovery process. In *Adv. Neural Inf. Process. Syst., Neurocolt II*, pages 1449–1456.
- [45] Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. 2018. Inverse molecular design using machine learning: Generative models for matter engineering. *Science (New York, N.Y.)* 361:360–365.
- [46] Thomas Blaschke, Marcus Olivecrona, Ola Engkvist, Jürgen Bajorath, and Hongming Chen. 2018. Application of Generative Autoencoder in De Novo Molecular Design. *Molecular Informatics* 37:1700123.
- [47] Rafael Gómez-Bombarelli, Jennifer N. Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, Ryan P. Adams, and Alán Aspuru-Guzik. 2018. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Central Science* 4:268–276.
- [48] Erik Cambria and Bebo White. 2014. Jumping nlp curves: A review of natural language processing research. *IEEE Computational intelligence magazine* 9:48–57.
- [49] Lorillee Tallorin, JiaLei Wang, Woojoo E Kim, Swagat Sahu, Nicolas M Kosa, Pu Yang, Matthew Thompson, Michael K Gilson, Peter I Frazier, Michael D Burkart, and Nathan C Gianneschi. 2018. Discovering de novo peptide substrates for enzymes using machine learning. *Nature Communications* 9:5253.
- [50] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22:1345–1359.
- [51] Han Altae-tran, Bharath Ramsundar, Aneesh S Pappu, and Vijay Pande. 2017. Low Data Drug Discovery with One-Shot Learning. *ACS Central Science* 3:283–293.
- [52] Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical Networks for Few-shot Learning. In *Advances in Neural Information Processing Systems*. Long Beach, CA, USA.
- [53] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems*. Barcelona, Spain.
- [54] Yao Quanming, Wang Mengshuo, Jair Escalante Hugo, Guyon Isabelle, Hu Yi-Qi, Li Yu-Feng, Tu Wei-Wei, Yang Qiang, and Yu Yang. 2018. Taking human out of learning applications: A survey on automated machine learning. *arXiv preprint arXiv:1810.13306* .

- [55] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. 2016. RL2: Fast Reinforcement Learning via Slow Reinforcement Learning. arXiv Preprint pages 1–14.
- [56] Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. 2018. Meta-Reinforcement Learning of Structured Exploration Strategies. In 32nd Conference on Neural Information Processing Systems, NeurIPS. Montreal, Canada.
- [57] Kunkun Pang, Mingzhi Dong, Yang Wu, and Timothy Hospedales. 2018. Meta-Learning Transferable Active Learning Policies by Deep Reinforcement Learning. arXiv Preprint pages 1–8.
- [58] Meng Fang, Yuan Li, and Trevor Cohn. 2017. Learning how to Active Learn: A Deep Reinforcement Learning Approach. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, page 595–605. Association for Computational Linguistics, Copenhagen, Denmark.
- [59] Guangshun Wang, Xia Li, and Zhe Wang. 2016. APD3: the antimicrobial peptide database as a tool for research and education. Nucleic acids research 44:D1087–93.
- [60] Rainier Barrett, Shaoyi Jiang, and Andrew D White. 2018. Classifying antimicrobial and multifunctional peptides with Bayesian network models. Peptide Science 110:e24079.
- [61] Paweł Smialowski, Gero Doose, Philipp Torkler, Stefanie Kaufmann, and Dmitrij Frishman. 2012. PROSO II—a new method for protein solubility prediction. The FEBS journal 279:2192–2200.
- [62] Gang Cheng, Hong Xue, Guozhu Li, and Shaoyi Jiang. 2010. Integrated antimicrobial and nonfouling hydrogels to inhibit the growth of planktonic bacterial cells and keep the surface clean. Langmuir : the ACS journal of surfaces and colloids 26:10425–8.
- [63] Michael C Sweeney, Anne-Sophie S Wavreille, Junguk Park, Jonathan P Butchar, Susheela Tridandapani, and Dehua Pei. 2005. Decoding protein-protein interactions through combinatorial chemistry: sequence specificity of SHP-1, SHP-2, and SHIP SH2 domains. Biochem 44:14932–14947.
- [64] Rainier Barrett, Shaoyi Jiang, and Andrew D. White. 2018. Classifying antimicrobial and multifunctional peptides with Bayesian network models. Peptide Science 110:e24079.
- [65] Martín Abadi, Ashish~Agarwal, Paul~Barham, Eugene~Brevdo, Zhifeng~Chen, Craig~Citro, Greg~S.~Corrado, Andy~Davis, Jeffrey~Dean, Matthieu~Devin, Sanjay~Ghemawat, Ian~Goodfellow, Andrew~Harp, Geoffrey~Irving, Michael~Isard, Yangqing Jia, Rafal~Jozefowicz, Lukasz~Kaiser, Manjunath~Kudlur, Josh~Levenberg, Dandelion~Mané, Rajat~Monga, Sherry~Moore, Derek~Murray, Chris~Olah, Mike~Schuster, Jonathon~Shlens, Benoit~Steiner, Ilya~Sutskever, Kunal~Talwar, Paul~Tucker, Vincent~Vanhoucke, Vijay~Vasudevan, Fernanda~Viégas, Oriol~Vinyals, Pete~Warden, Martin~Wattenberg, Martin~Wicke, Yuan~Yu, and Xiaoqiang~Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.
- [66] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. CoRR abs/1412.6980.
- [67] Alex Nichol, Joshua Achiam, and John Schulman. 2018. On First-Order Meta-Learning Algorithms. eprint arXiv pages 1–15.
- [68] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In Proceedings of the 34th International Conference on Machine Learning. Sydney, Australia.
- [69] Sameer Khurana, Reda Rawi, Khalid Kunji, Gwo-Yu Chuang, Halima Bensmail, and Raghvendra Mall. 2018. DeepSol: a deep learning framework for sequence-based protein solubility prediction. Bioinformatics (Oxford, England) 34:2605–2613.
- [70] Andrew D. White, Andrew J. Keefe, Ann K. Nowinski, Qing Shao, Kyle Caldwell, and Shaoyi Jiang. 2013. Standardizing and simplifying analysis of peptide library data. Journal of Chemical Information and Modeling 53:493–499.
- [71] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. 2019. Rapid learning or feature reuse? towards understanding the effectiveness of maml. arXiv preprint arXiv:1909.09157 .