Active Learning for High Throughput Screening

Kurt De Grave, Jan Ramon, and Luc De Raedt

Katholieke Universiteit Leuven, Celestijnenlaan 200A, 3001 Leuven, Belgium
{Kurt.DeGrave,Jan.Ramon,Luc.DeRaedt}@cs.kuleuven.be
http://www.cs.kuleuven.be/cwis/research/dtai/

Abstract. An important task in many scientific and engineering disciplines is to set up experiments with the goal of finding the best instances (substances, compositions, designs) as evaluated on an unknown target function using limited resources. We study this problem using machine learning principles, and introduce the novel task of *active k-optimization*. The problem consists of approximating the k best instances with regard to an unknown function and the learner is active, that is, it can present a limited number of instances to an oracle for obtaining the target value. We also develop an algorithm based on Gaussian processes for tackling active k-optimization, and evaluate it on a challenging set of tasks related to structure-activity relationship prediction.

Key words: Active Learning, Chemical compounds, Optimization, QSAR

1 Introduction

The philosophy of science has since a long time studied scientific discovery processes, and recently, the artificial intelligence community has taken up the challenge as to study how scientific discovery processes can be automated [1]. One aspect that is quite central in scientific discovery, as well as in many engineering problems, is that of determining the next experiment to be carried out.

In this paper, we apply machine learning principles to select the next experiment in High Throughput Screening (HTS), an important step in the drug discovery process, in which many chemical compounds are screened against a biological assay. The goal of this step is to find a few lead compounds within the entire compound library that exhibit a very high activity in the assay. This is also the setting of the new robot that is currently under development in the robot scientist project [1] at the University of Aberystwyth. The task is akin to many other scientific and engineering disciplines, where the challenge is to identify or design those instances that have optimal performance according to some criterion that needs to be optimized. For instance, in membrane design, it is important to find those parameters of the process that yield the best performance [2]; in coherent laser control, the goal is to find the laser pulse that maximally catalyzes a chemical reaction [3]. In this type of application, the target criterion is unknown to the scientist or engineer and only partial information can be obtained by testing specific instances for their performance. Such tests correspond to experiments and can be quite expensive.

In HTS, it is not sufficient to find just a single optimal example. The optimal compound might ultimately not be usable as a starting point for the next step in the drug discovery process for various reasons unrelated to its performance in the assay. Therefore, a number of alternatives need to be found as well. Ideally, each of these near-optimal alternatives would have a different modus operandi. The challenge then is to identify the k best performing instances using as few experiments as possible. We will refer to this task as *active k-optimization*.

This task is closely related to global function optimization. It is also related to active learning in a regression setting [4], where the goal is to find a good approximation of the unknown target function by querying for the value of as few instances as possible. Whereas this approach allows one to identify the best scoring instances, it is also bound to waste resources in the low scoring regions of the function. Thus, in contrast to active regression, an extra ingredient is added to the problem that is reminiscent of reinforcement learning. The learner will have to find the right balance between exploring the space of possible instances and exploiting those regions of the search space that are expected to yield high scores according to the current approximation of the function. Finally, active k-optimization differs also from active concept-learning that has already been applied in applications such as structure-activity relationship prediction [5] in that a regression task has to be performed.

This paper is organized as follows: in Section 2 we formalize the problem, and in Section 3 we propose a Gaussian process model for tackling it. In Section 4 we investigate a number of different strategies for balancing exploration and exploitation. We evaluate our approach experimentally in Section 5. Finally, We discuss related work and possible extensions in Section 6.

2 Problem statement

Our work is especially motivated by the structure-activity relationship domain, where high-throughput approaches assume the availability of a large, diverse but fixed library of compounds. Hence, the pool-based active learning setting is most appropriate. In this setting, the learner incurs a cost only when asking for the measurement of the target value of a particular instance, which must be selected from a known, finite pool. In principle, the learner may be able to exploit the distribution of the examples in the pool without cost. To some extent, this setting is therefore also a semi-supervised learning setting.

The problem sketched in section 1 can be more formally specified as follows:

GIVEN:

- a pool $\mathcal{P}$ of instances,
- an unknown function f that maps instances $x \in \mathcal{P}$ on their target values $f(x)$,
- an oracle that can be queried for the target value of any example $x \in \mathcal{P}$,
- the maximal number N_{max} of queries that the oracle is willing to answer,
- the number k of best scoring examples searched for.

FIND:

– the top k instances in $\mathcal{P}$, that is, the k instances in $\mathcal{P}$ that have the highest values for f.

One can see that the above combinatorial optimization problem is a close relative to the problem of global function optimization. Algorithms developed in the discipline of global function optimization only consider $k = 1$ and are optimized for continuous domains. Still, largely the same concepts and techniques can be used.

From a machine learning perspective, the key challenge is to determine the policy for determining the next query to be asked, based on the already known examples. This policy will have to keep the right balance between exploring the whole pool of examples and exploiting those regions in the pool that look most promising.

3 Gaussian process model

We will use a Gaussian process model [6] for learning, also known as Kriging. In this section we briefly review the necessary theory. More detailed explanations can be found in several textbooks on the subject [7, 8].

We first introduce some notation. We assume that there is a feature map $\phi : \mathcal{P} \to F$ mapping examples to a feature space F. We denote with $X_N = [x_1 x_2 \dots x_N]^\top$ the vector of the N first examples, with $T_N = [t_1 t_2 \dots t_N]^\top$ the vector of their target values, and with $\Phi_N = [\phi(x_1)\phi(x_2) \dots \phi(x_N)]^\top$ the matrix where each row is the image of an example (abusing notation in case F has infinite dimension). For our objective criterion, $\|T_N\|_{\text{best}-k}$ is the average of the k largest elements of the vector T_N, where we assume all target values to be positive. The notation $\|\cdots\|_{\text{best}-k}$ is warranted since the function satisfies all properties of a vector norm under this assumption.

We assume that there is a linear approximate model for the target value $t(x)$ of instances

$$m(x) = w^\top \phi(x) \tag{1}$$

(with $w \in F$ a weight vector) such that the values of the modeling error $t(x) - m(x)$ for examples randomly drawn from the pool $\mathcal{P}$ are independently Gaussian distributed with zero mean and variance σ^2. We use the following notation to denote that a random variable has a Gaussian distribution:

$$t(x) - m(x) \sim \mathcal{N}(0, \sigma^2). \tag{2}$$

In matrix notation Equation (2) becomes $P(T_N|\Phi_N, w) \sim \mathcal{N}(\Phi_N w, \sigma^2 I)$. Our prior belief for the vector w is Gaussian with zero mean and covariance matrix Σ_{pw}. One can compute the posterior $P(w|T_N, \Phi_N)$ using

$$P(w|T_N, \Phi_N) \propto P(w)P(T_N|\Phi_N, w). \tag{3}$$

A straightforward derivation gives

$$w|T_N, X_N \sim \mathcal{N}(\bar{w}_N, \Sigma_{w,N}) \tag{4}$$

where the mean $\bar{w}_N$ and variance $\Sigma_{w,N}$ is

$$\bar{w}_N = \sigma^{-2}\Sigma_{w,N}\Phi_N^\top T_N$$
$$\Sigma_{w,N} = (\sigma^{-2}\Phi_N^\top\Phi_N + \Sigma_{pw}^{-1})^{-1}.$$

For a new example x_* we can then estimate the target value by

$$t_*|X_N, T_N, x_* \sim \mathcal{N}\big(\bar{w}_N^\top\phi(x_*), \phi(x_*)^\top\Sigma_{w,N}\phi(x_*)\big) \tag{5}$$

One can show that this formula is equivalent to the following distribution which does not refer to feature space explicitly:

$$t_*|X_N, T_N, x_* \sim \mathcal{N}\big(\bar{t}_*, var(t_*)\big) \tag{6}$$

where

$$\bar{t}_* = k(x_*, X_N)(k(X_N, X_N) + \sigma^2 I_N)^{-1}T_N \tag{7}$$

$$var(t_*) = k(x_*, x_*) - k(x_*, X_N)(k(X_N, X_N) + \sigma^2 I_N)^{-1}k(X_N, x_*) \tag{8}$$

and where k is a kernel defined by

$$k(x, y) = \phi(x)^\top\Sigma_{pw}\phi(y) \tag{9}$$

Here, we use the abbreviations

$$k(x_*, X_N) = \big[k(x_*, x_1)k(x_*, x_2)\dots k(x_*, x_N)\big]^\top$$
$$k(X_N, x_*) = k(x_*, X_N)^\top$$

for vectors of kernel values, and

$$k(X_N, X_N) = [k(x_1, X_N)k(x_2, X_N)\dots k(x_N, X_N)]$$

for a matrix of kernel values. $k(X_N, X_N)$ is called the Gram matrix.

4 Selection strategies

Different example selection strategies exist. In geostatistics, they are called infill sampling criteria [9].

In active learning, in line with the customary goal of inducing a model with maximal accuracy on future examples, most approaches involve a strategy aiming at greedily improving the quality of the model in regions of the example space where its quality is lowest. One can select new examples for which the predictions of the model are least certain or most ambiguous. Depending on the learning algorithm, this translates to near decision boundary selection, ensemble entropy

reduction, version space shrinking, and others. In our model, it translates to *maximum variance* on the predicted value or $\arg\max(var(t_*))$.

Since our goal is not model accuracy but finding good instances, a more appropriate strategy is to select the example that the current model predicts to have the best target value, or $\arg\max(\bar{t}_*)$. We will refer to this as the *maximum predicted* strategy. For continuous domains, it is not guaranteed to find the global, or even a local minimum [10].

A less vulnerable strategy is Cox and John's lower confidence bound criterion [11], which we will refer to as the *optimistic* strategy. The idea is to not sample the example in the database where the expected reward $\bar{t}_*$ is maximal, but the example where $\bar{t}_* + b \cdot \sqrt{var(t_*)}$ is maximal. The parameter b is the level of optimism. It determines the balance between exploitation and exploration. It is obvious that the maximum predicted and maximum variance strategies are special cases of the optimistic strategy, with $b = 0$ and $b = \infty$ respectively. In a continuous domain, this strategy is not guaranteed to find the global optimum because its sampling is not dense [10].

Another strategy is to select the example x_{N+1} that has the highest probability of improving the current solution [12]. One can estimate this probability as follows. Let the current step be N, the value of the set of k best examples be $\|T_N\|_{\text{best}-k}$ and the k-th best example be $x_{\#(k,N)}$ with target value $t_{\#(k,N)}$. When we query example x_{N+1}, either t_{N+1} is smaller than or equal to $t_{\#(k,N)}$, or t_{N+1} is greater. In the first case, our set of k best examples does not change, and $\|T_{N+1}\|_{\text{best}-k} = \|T_N\|_{\text{best}-k}$. In the latter case, x_{N+1} will replace the k-th best example in the set and the solution will improve. Therefore, this strategy selects the example x_{N+1} that maximizes $P(t_{N+1} > t_{\#(k,N)})$. We can evaluate this probability computing the cumulative Gaussian

$$P(t_{N+1} > t_{\#(k,N)}) = \int_{t=t_{\#(k,N)}}^{\infty} \mathcal{N}(\bar{t}_*, var(t_*))dt \ , \tag{10}$$

where $\bar{t}_{N+1}$ and $\text{var}(t_{N+1})$ can be obtained from Equations (7, 8). In agreement with [13], we call this the *most probable improvement* (MPI) strategy.

Yet another variant is the strategy used in the Efficient Global Optimization (EGO) algorithm [14]. EGO selects the example it expects to improve most upon the current best, i.e the one with highest

$$\mathbb{E}[\max(0, t - t_{\#(k,N)})] = \int_{t=t_{\#(k,N)}}^{\infty} (t - t_{\#(k,N)})\mathcal{N}(\bar{t}_*, var(t_*))dt \ . \tag{11}$$

This criterion is called *maximum expected improvement* (MEI).

In real-world applications it is not only important to find a solution quickly, but also to know when the optimal (or an adequate) solution has been found. The trade-off one has to make here is between budget and quality.

In a large number of situations, one will have a fixed budget and the goal will be to have an optimal solution when the budget is exhausted. Sometimes however, one can save significantly on the budget when a slightly suboptimal solution is acceptable or when the risk of having a suboptimal solution is small.

One approach is to bound the probability that any of the non-queried examples is better than the k-th best example so far. From Equation (10) we can compute for a particular example x that has not been queried the probability that its target value t will be larger than $t_{\#(k,N)}$. We can then write

$$P(\exists x \in \mathcal{P} \setminus X_N : f(x) > t_{\#(k,N)}) \leq \sum_{x \in \mathcal{P} \setminus X_N} P(f(x) > t_{\#(k,N)}) \qquad (12)$$

which is a tight upper bound if the individual $P(f(x) > t_{\#(k,N)})$ are small (as is the case when we consider to stop querying) and independent.

5 Experimental Evaluation

As sketched in the introduction, we shall experimentally evaluate our collection of methodsin the area of high throughput screening in the context of drug lead discovery. In particular, we shall evaluate the algorithms on the US National Cancer Institute (NCI) 60 anticancer drug screen (NCI60) dataset [15]. This repository contains measurements of the inhibitory power of tens of thousands of chemical compounds against 59 different[1] cancer cell lines. NCI reports the log-concentration required for 50% cancer cell growth inhibition (GI_{50}) as well as cytostatic and cytotoxic effect measures, but we only used the log GI_{50} data. Real world drug discovery screening operations would normally include non-toxicity in the measure to optimize for[2].

To perform a measurement, each compound is diluted repeatedly, yielding a geometric series of concentrations. The actual GI_{50} can turn out to be outside the range of concentrations chosen a-priori. In that case, one only knows an upper or lower bound for the value, and a new measurement for that compound must be performed to collapse the interval to a point value. We ignored such out of bounds measurements.

To improve interpretability of the experiments, an equally sized pool of 2,000 compounds was randomly selected from each assay. This also saved computational resources, though it would have been possible to use the entire dataset, for all algorithms are only of complexity $O(N_{max} \cdot \#\mathcal{P})$ as long as both the number of features and the budget are constant.

We used a linear kernel. The chemical structure of each compound was represented as 1024 FP2 fingerprints, calculated using Open Babel 2.1.0. The algorithms were bootstrapped with GI_{50} measurements of ten random compounds. Since the result depends on this random boot sample, each experiment was repeated 20 times and the results were averaged.

In each assay, NCI measured some compounds repeatedly. For these compounds, the dataset lists the standard deviation among the measurements, as

[1] One of the originally 60 cell lines was evicted because it was essentially a replicate of another [16].

[2] E.g. the specificity index, usually defined as the log-ratio of the toxic concentration to the effective concentration.

well as the average. In order to estimate the measurement error for each assay, we used the unweighted average standard deviation over all repeated measurements in the assay. This value was used as the standard deviation σ in the Gaussian term of our model in Equation (2).

To evaluate our algorithms in practice, we recorded $\|T_N\|_{\text{best}-k}$ as a function of the fraction of compounds tested. For every setting (selection strategy, value of k), these functions were then averaged over the 59 datasets considered. Figure 1 plots these curves for $k \in \{1, 10, 25, 100\}$ for all described strategies and random selection. For the optimistic strategy, we tested optimism levels of 0.5, 1, and 2.

Table 1 lists for several budgets N_{max} which strategy is best (attains the highest $\|T_{N_{max}}\|_{\text{best}-k}$). The budget is shown as a percentage of the pool size. For each different strategy, the table then also gives the Wilcoxon signed-rank test p-value for the null hypothesis that the difference between the top-k values of this strategy and those of the best strategy is on average 0.

We are now well equipped to answer four important questions about our algorithm:
Q1 Do active k-optimization strategies isolate valuable instances quicker than random selection?
Q2 What is the relative performance of the different selection strategies listed in Section 4?
Q3 Do strategies that take k into account perform better than strategies that do not?
Q4 Can the stopping criterion (Eq. 12) be used to decide when a near-optimal solution has been found?

5.1 Expedience

From the results presented in Table 1 and Figure 1, one can see that random example selection clearly performs worse than all other selection methods in all settings, except for the maximum variance strategy which does still worse for large budgets, especially for $k = 100$. We can conclude that the answer to question **Q**1 is positive, because actively choosing examples with one of the presented strategies substantially speeds up the finding of examples with high target values.

It is remarkable that the starting points of the random strategy are lower for higher k. This is due to the fact that the distribution of target values is skewed: compounds with very small target values are sparser than compounds with very large target values. In this way, $\|T_N\|_{\text{best}-k}$ decreases only slowly while $\|T_N\|_{\text{worst}-k}$ (the average of the k smallest elements of T_N) increases quickly for larger k. This causes the value of a random sample to be lower when scaled to a $[0, 1]$ interval. That the non-random strategies start higher than the random strategy for $k = 25$ and $k = 100$ is due to the fact that there are only 10 bootstrapping examples, and the non-random strategies actively select 15 (for $k = 25$) or 90 (for $k = 100$) examples before they can be evaluated a first time.

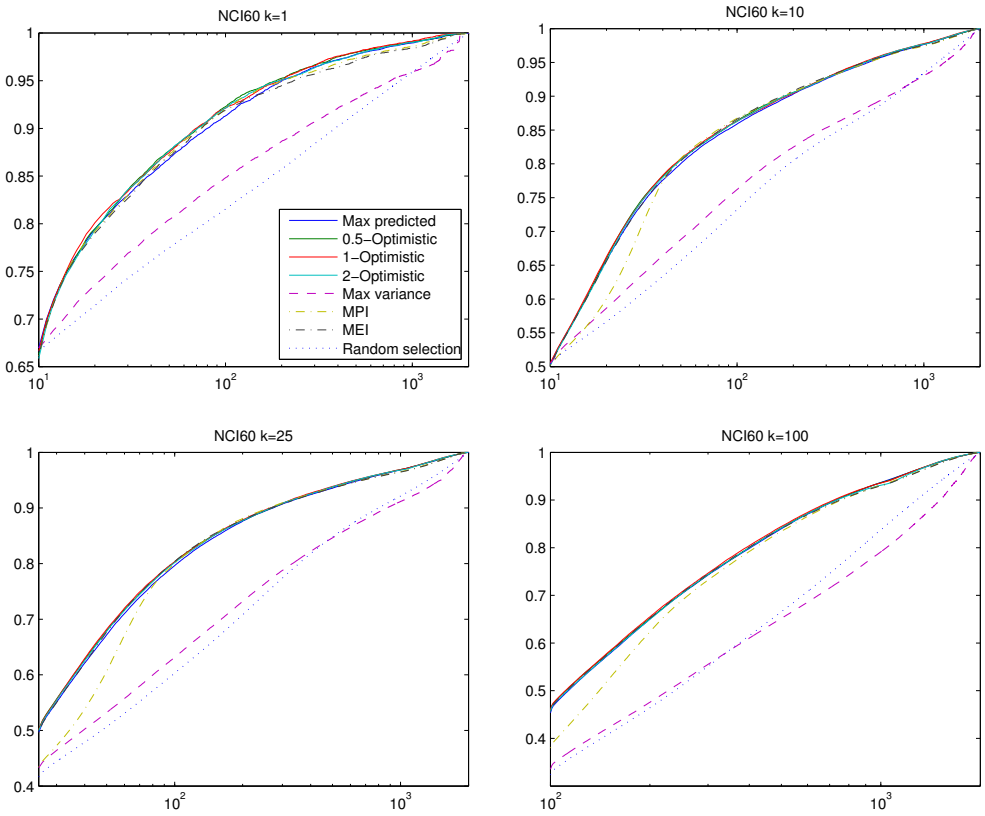

Fig. 1. The value of $\|T_N\|_{\text{best}-k}$ in each step, for all proposed active learning strategies and random selection, averaged over 20 runs for each of the 59 datasets. A log scale is used on the horizontal axis to reveal the performance for small as well as large budgets. The vertical axis is scaled to place the aggregate target value of the overall k best compounds at one and the worst k compounds at zero.

5.2 Relative performance

Unsurprisingly, one can see that querying the maximally uncertain example is (in contrast to settings where one tries to optimize accuracy) not a good k-optimization strategy.

Overall, on the NCI60 datasets, the optimistic strategy with an optimism level of 1 was most robust. In all situations considered, it performed either best or not significantly worse than the best strategy. The difference with 2 and 0.5 optimism is more pronounced for higher values of k. Note that we exploited the information in the NCI datasets about the accuracy of the measurements. For other datasets that do not allow to estimate the accuracy of the input data, it may be harder to come up with a good value for $var(t_*)$. One can use a maximum likelihood estimate, at the cost of some robustness [9].

Budget	10%	15%	20%	25%	10%	15%	20%	25%
k	1				10			
Max predicted	0.305	0.304	0.039	0.088	0.106	0.497	0.040	0.021
Optimistic ($b = 0.5$)	**Best**	**Best**	0.282	0.392	0.274	0.456	0.251	0.111
Optimistic ($b = 1$)	0.837	0.776	**Best**	**Best**	0.141	0.390	**Best**	**Best**
Optimistic ($b = 2$)	0.898	0.472	0.094	0.229	0.179	0.298	0.179	0.298
Max variance	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ
MPI	0.946	0.538	0.108	0.174	0.455	0.230	0.189	0.052
MEI	0.037	0.057	0.005	0.047	**Best**	**Best**	0.934	0.809
Random	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ
Budget	10%	15%	20%	25%	10%	15%	20%	25%
k	25				100			
Max predicted	0.074	0.437	0.015	0.015	0.046	0.063	0.003	0.010
Optimistic (b= 0.5)	0.202	0.319	0.177	0.192	0.197	0.118	0.007	0.022
Optimistic ($b = 1$)	0.280	0.634	**Best**	**Best**	**Best**	**Best**	**Best**	**Best**
Optimistic ($b = 2$)	0.083	0.673	0.264	0.478	0.042	0.170	0.016	0.068
Max variance	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ
MPI	**Best**	**Best**	0.385	0.141	10^{-5}	0.005	0.003	0.001
MEI	0.487	0.184	0.158	0.083	0.254	0.492	0.019	0.001
Random	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ	ϵ

Table 1. p-values for $k \in \{1, 10, 25, 100\}$. ϵ indicates that $p < 10^{-8}$.

Greedily querying the example for which the highest target value is predicted, performs slightly worse than the optimistic strategy. The MPI strategy performs worse than the optimistic strategies in the very beginning, except for $k = 1$. It performs (and allegedly behaves) similarly to the maximum variance strategy when it hasn't seen many more examples than the 10 random bootstraps. From about 5% for $k = 10$ and 10% for $k = 25$, its performance is competitive and sometimes best, but it again becomes suboptimal for high budgets. The MEI strategy performs extremely well for $k = 10$, but is outperformed in some other settings. This concludes our answer to question **Q**2.

5.3 Utility of advance knowledge of k

In Table 2 and Figure 2 we see that the active learning strategies that explicitly take k into account, perform far better than their global optimization ($k = 1$) peers, except for the warmup of MPI. However, from question **Q**2 we learned that the most robust strategy on our datasets, 1-optimism, performs as well. Since optimism does not rely on prior knowledge of k, the answer to question **Q**3 is negative.

5.4 Stopping criterion

To evaluate the stopping criterion, we used Equation (12) to estimate $P(\exists x \in \mathcal{P} \setminus X_N : f(x) > t_{\#(k,N)})$, the probability that there exists an unseen example

Budget	10%	15%	20%	25%
MPI $k_{target} = 1$	10^{-7}	ϵ	ϵ	ϵ
MEI $k_{target} = 1$	ϵ	ϵ	ϵ	ϵ
MPI $k_{target} = 25$	**Best**	**Best**	**Best**	**Best**
MEI $k_{target} = 25$	0.487	0.184	0.394	0.640

Table 2. p-values for $k = 25$. ϵ indicates that $p < 10^{-8}$.

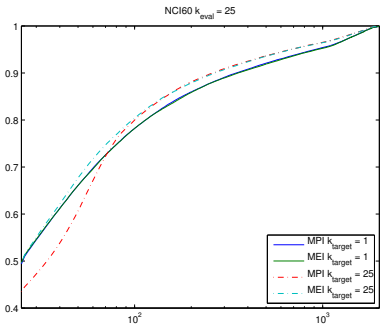

Fig. 2. The value of $\|T_N\|_{\text{best}-25}$ in each step, for the MPI and MEI strategies, optimizing for either k=1 or k=25.

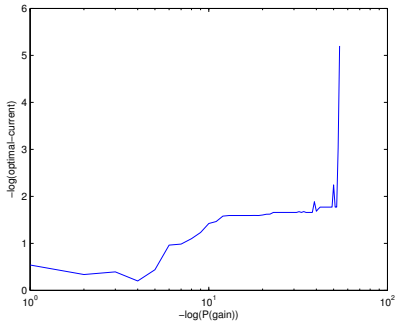

Fig. 3. Stopping criterion: negative logarithm of the difference between the optimal solution and current solution plotted against the negative logarithm of predicted probability of suboptimality according to Equation (12)

in the pool $\mathcal{P}$ which is better than the k-th best seen so far. We did so during one experiment with the MPI strategy for every data set, and recorded these probabilities together with the differences between the solution at that point and the optimal solution. In this way we can evaluate how much value one would lose on average if one would stop the screening when the probability of finding anything better would drop below a certain threshold.

In Figure 3, the negative logarithm of the differences between solution so far and optimal solution, i.e. $-\log(\|f(\mathcal{P})\|_{\text{best}-k} - \|T_N\|_{\text{best}-k})$, is plotted against the negative logarithm of the estimated probability that there is still a better solution, i.e. $-\log(P(\exists x \in \mathcal{P} \setminus X_N : f(x) > t_{\#(k,N)}))$. The standard deviations on the points in this curve are all below 0.2.

From Figure 3 one can see that there is a good relation between the estimated probability that the best solution has not yet been found and the optimality of the current solution. In particular, when the stopping criterion predicts a very small probability of finding a better solution, one can be confident that querying more examples will not be very useful. This answers question **Q**4 positively.

6 Related work and possible extensions

To summarize, we introduced the active k-optimization problem in a machine learning context, we developed an approach based on Gaussian processes to tackling it, and we applied it to a challenging structure-activity relationship prediction task, demonstrating good performance.

Our work is related to several articles that combine kernel methods and Gaussian processes both in the machine learning and the global optimization communities. In machine learning, one aims at improving prediction accuracy, and common strategies select the most uncertain examples, or select the examples that maximize information gain. In global optimization, Gaussian processes are a popular surrogate to save on expensive function evaluations [9, 13].

The setting we introduced is also important for applications in HTS, where so far active learning has only been applied for classification or regression purposes but not for optimization. E.g. [5] shows that the maximum-predicted strategy works well for discriminating rare active compounds from inactives using an SVM. Furthermore, the NCI database has been used as benchmark for several machine learning approaches [18–20]. As the results show, classification of compounds can be learned to a certain extent, but accurate prediction (classifying borderline cases) is still harder than finding extreme values as in our setting.

Two interesting further questions for research are 1) whether one could make further gains by devising a strategy that also takes into account a budget that is fixed from the start, and 2) whether one can select several examples to be queried together in a single batch before getting the target values for all of them. This is often needed in HTS. To address the first question, one could e.g. focus the first fraction of the budget more on exploration and the last part only on exploitation. The second question requires one to spread selections over the space in order to avoid obtaining too many correlated values [17]. A few authors have touched upon this problem in the context of surrogate-based optimization, but the batch size was algorithm driven as opposed to application constraint driven, e.g. [10]. This raises a more general problem: given some collected X_N, T_N training data and a pool $\mathcal{P}$ of examples that one could query next, select n new examples to query. In such a situation, it may not be optimal to select examples that individually optimize some criterion. In the ideal case, one would like to optimize the joint contribution of the entire batch. E.g. if $k = 1$, the probability that querying examples $x_{N+1} \dots x_{N+n}$ would improve the solution would be

$$P\big(\max\{t_{N+1} \dots t_{N+n}\} > t_{\#(k,N)} \mid X_N, T_N\big)$$

which evaluates to the integral of a Gaussian over a union of half-spaces. For large n, it is nontrivial to select the n examples for which this value is maximized. However, one can efficiently select $x_{N+1} \dots x_{N+n}$ in order, such that in every step the example x_{N+i} is selected that maximizes

$$P\big(\max\{t_{N+1} \dots t_{N+i}\} > t_{\#(k,N)} \mid X_{N+i-1}, T_{N+i-1}\big) \ .$$

Acknowledgements Kurt De Grave is supported by GOA/08/008 "Probabilistic Logic Learning". Jan Ramon is a post-doctoral fellow of the Fund for Scientific Research (FWO) of Flanders. High performance computational resources were provided by http://ludit.kuleuven.be/hpc.

References

1. King, R. et al.: Functional genomic hypothesis generation and experimentation by a robot scientist. Nature **427** (2004) 247–252
2. Vandezande, P. et al.: High throughput screening for rapid development of membranes and membrane processes. J. Membrane Science **250**(1-2) (2005) 305–310
3. Form, N. et al.: Parameterisation of an acousto-optic programmable dispersive filter for closed-loop learning experiments. J. Modern Optics **55**(1) (Jan 2007) 1–13
4. Cohn, D., Ghahramani, Z., Jordan, M.I.: Active Learning with Statistical Models. J. Artificial Intelligence Research **4** (1996) 129–145
5. Warmuth, M.K. et al.: Active learning with support vector machines in the drug discovery process. J. Chem. Inf. Comput. Sci. **43**(2) (March 2003) 667–673
6. Gibbs, M.: Bayesian Gaussian Processes for Regression and Classification. PhD thesis, University of Cambridge (1997)
7. Rasmussen, C., Williams, C.: Gaussian Processes for Machine Learning. The MIT Press, Cambridge, MA, USA (2006)
8. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006)
9. Sasena, M.J.: Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations. PhD thesis, University of Michigan (2002)
10. Jones, D.R.: A taxonomy of global optimization methods based on response surfaces. J. Global Optimization **21** (2001) 345–383
11. Cox, D.D., John, S.: SDO: a statistical method for global optimization. Multidisciplinary Design Optimization (Hampton, VA, 1995). SIAM, Philadelphia, PA (1997) 315–329
12. Kushner, H.J.: A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. J. Basic Engineering (Mar. 1964) 97–106
13. Lizotte, D. et al.: Automatic gait optimization with gaussian process regression. In: Proc. 20th Int. Joint Conference on Artificial Intelligence. (2007) 944–949
14. Jones, D.R., Schonlau, M.: Efficient global optimization of expensive black-box functions. J. Global Optimization **13**(4) (December 1998) 455–492
15. Shoemaker, R.: The NCI60 human tumour cell line anticancer drug screen. Nat. Rev. Cancer **6** (Oct 2006) 813–823
16. Nishizuka, S. et al.: Proteomic profiling of the NCI-60 cancer cell lines using new high-density reverse-phase lysate microarrays. PNAS 100. (Nov 2003) 14229–14234
17. Guestrin, C., Krause, A., Singh, A.P.: Near-optimal sensor placement in gaussian processes. ICML-2005. 265–272
18. Swamidass, S.J. et al.: Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity. Bioinformatics **21**(suppl 1) (2005) i359–368
19. Ceroni, A., Costa, F., Frasconi, P.: Classification of small molecules by two- and three-dimensional decomposition kernels. Bioinformatics **23**(16) (2007) 2038–2045
20. Menchetti, S., Costa, F., Frasconi, P.: Weighted decomposition kernels. ICML-2005. 585–592