Summer 8-15-2019

Automating Active Learning for Gaussian Processes

Gustavo Malkomes
Washington University in St. Louis

WASHINGTON UNIVERSITY IN ST. LOUIS

McKelvey School of Engineering
Department of Computer Science & Engineering

Dissertation Examination Committee:
Roman Garnett, Chair
Ayan Chakrabarti
Sanmay Das
Peter Frazier
Brendan Juba

Automating Active Learning for Gaussian Processes

by
Luiz Gustavo Sant Anna Malkomes Muniz

A dissertation presented to
The Graduate School
of Washington University in
partial fulfillment of the
requirements for the degree
of Doctor of Philosophy

August 2019
Saint Louis, Missouri

Contents

List of Tables

List of Figures

Acknowledgments

I would like to express my gratitude and appreciation to all those that helped me throughout the pursuit of this degree. Family, friends, collaborators, professors, colleagues, staff, committee members, anonymous reviewers; all have contributed to this thesis in some way. Below, I extend my special thanks to some of them.

To Roman Garnett, who has been an outstanding mentor during these past several years, nothing close to the stereotype of PhD Comics[1]. Roman is one of the brightest people I know, the most Bayesian of all, and excellent company for trivia nights and random, secret bars in Montreal. Thank you for giving me the freedom to pursue various projects, for being supportive and such a great advisor.

I was also fortunate to get the opportunity to work with Ben Mosley, who was practically my co-advisor. Ben has astonishing mathematical intuition and provided me all the assistance that I need during my first successful research project as a Ph.D. student. For valuable professional advice, great stories, and wonderful laughter, I thank you.

I am very lucky to have found my dearest wife from Minas Gerais here in St. Louis. Andressa, my best friend, thanks for being so supportive and always standing by my side while I completed my degree. My parents and sisters, who have been my foundation and balance throughout all my life. Mom, Dad, Joyce, Jozy, thank you for raising me with a questioning mind and always believing in me.

[1] www.phdcomics.com

To my lab-mates, Shali Jiang and Henry Chai. Shali for his effortless humor, incredible mathematical precision, and most importantly for always being a loyal, true friend. Henry, thank you for being kind, welcoming and helpful, especially, with proof-reading. To Kefu Lu and Mithun Chakraborty, who are not technically lab-mates, but help me tremendously during my graduate career, I also extend my gratitude.

Thank you to my collaborators. Kilian Weinberger, who has taught me how to be professional and inspired me to dedicate my best to my academic life. Matt Kusner, who is one of the kindest people that I know. To Charles Schaff, Jacob Gardner, Wenlin Chen, Dennis Barbour, John Cunningham, Richard Mann, Blakeley Hoffman, Matthew Abbott, Geoff Converse, Alyssa Shofner. I never could have completed my research without your help.

Evan, my best roommate, and rightful best man, despite me not having one; you won the bet over darts! Adam Drescher, my very first friend in this country, who has eventually introduced me to more shots than I should have had, certainly the friendliest customer from the Three Kings bar. To Levi Vasconcelos, for always being such a good friend over the many years that we have known each other, for playing Starcraft II and sharing a curiosity about machine learning.

Finally, I would like to acknowledge the support of the Brazilian Federal Agency for Support and Evaluation of Graduate Education (CAPES) and the National Science Foundation (NSF) under award number IIA–1355406.

Luiz Gustavo Sant Anna Malkomes Muniz

Washington University in Saint Louis
August 2019

Dedicated to my parents.

ABSTRACT OF THE DISSERTATION

Automating Active Learning for Gaussian Processes

by

Luiz Gustavo Sant Anna Malkomes Muniz

Doctor of Philosophy in Computer Science

Washington University in St. Louis, August 2019

Research Advisor: Professor Roman Garnett

In many problems in science, technology, and engineering, unlabeled data is abundant but acquiring labeled observations is expensive – it requires a human annotator, a costly laboratory experiment, or a time-consuming computer simulation. Active learning is a machine learning paradigm designed to minimize the cost of obtaining labeled data by carefully selecting which new data should be gathered next. However, excessive machine learning expertise is often required to effectively apply these techniques in their current form. In this dissertation, we propose solutions that further automate active learning. Our core contributions are active learning algorithms that are easy for non-experts to use but that deliver results competitive with or better than human-expert solutions.

We begin introducing a novel active search algorithm that automatically and dynamically balances exploration against exploitation — without relying on a parameter to control this tradeoff. We also provide a theoretical investigation on the hardness of this problem, proving that no polynomial-time policy can achieve a constant factor approximation ratio for the expected utility of the optimal policy.

Next, we introduce a novel information-theoretic approach for active model selection. Our method is based on maximizing the mutual information between the output variable and the model class. This is the first active-model-selection approach that does not require updating each model for every candidate point. As a result, we successfully developed an automated audiometry test for rapid screening of noise-induced hearing loss, a widespread and preventable disability, if diagnosed early.

We proceed by introducing a novel model selection algorithm for fixed-size datasets, called Bayesian optimization for model selection (BOMS). Our proposed model search method is based on Bayesian optimization in model space, where we reason about the model evidence as a function to be maximized. BOMS is capable of finding a model that explains the dataset well without any human assistance.

Finally, we extend BOMS to active learning, creating a fully automatic active learning framework. We apply this framework to Bayesian optimization, creating a sample-efficient automated system for black-box optimization. Crucially, we account for the uncertainty in the choice of model; our method uses multiple and carefully-selected models to represent its current belief about the latent objective function.

Our algorithms are completely general and can be extended to any class of probabilistic models. In this dissertation, however, we mainly use the powerful class of Gaussian process models to perform inference. Extensive experimental evidence is provided to demonstrate that all proposed algorithms outperform previously developed solutions to these problems.

Chapter 1

Introduction

Countless problems in science, technology, and engineering require decision-making. Scientists design experiments to investigate various phenomena, engineers decide how to optimize an industrial process, pharmaceutical researchers select new drugs for testing, etc. In these problems, many factors impact the quality of a decision and reasoning about all possible factors, interaction, and side-effects can be very difficult for individuals. As a further complication, poor decision-making can be expensive, leading to the waste of physical resources and time.

One alternative is to use machine learning to improve or assist human decision-making. In particular, we focus on a class of machine learning algorithms known as *active learning* [74]. In active learning, we assume that unlabeled data is abundant, but labeling data is expensive—it requires a human annotator, a costly laboratory experiment, or a time-consuming computer simulation. Thus, the goal is to minimize the cost of obtaining labeled data by carefully selecting which new data should be gathered next. Figure 1.1 shows an illustration of a typical active learning pipeline. We begin with a small set of observed data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}$. Then, we use the current data to train a machine learning model. The model is used first to analyze a larger set of unlabeled data and then to recommend which next point $\mathbf{x} \in \mathcal{X}$

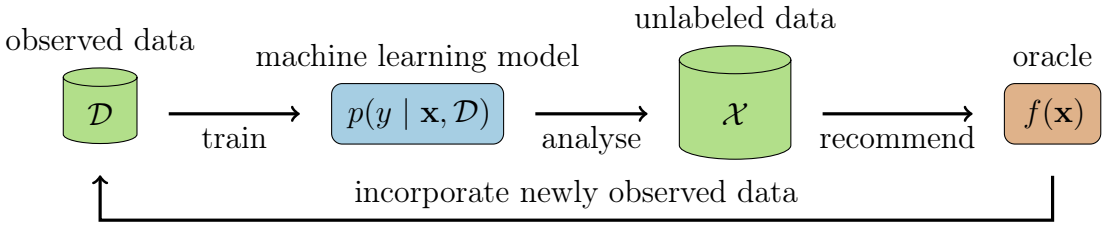


Figure 1.1: An illustration of the active learning loop. It starts with a small set of data $\mathcal{D}$. Then, an appropriate machine learning model is trained and used for explaining all current observations in $\mathcal{D}$. The model is later used to analyze the remaining options in $\mathcal{X}$ and recommend which next decision $\mathbf{x} \in \mathcal{X}$ we should investigate next. We repeat the whole process, adding the newly observed data to $\mathcal{D}$.

should be submitted to the expensive labeling (or oracle) function. We repeat these steps after including the new (potentially noisy) observation to our dataset.

Despite the success of (active) machine learning, current techniques still require a considerable amount of problem-specific expertise, making it difficult to be used in different problems or by non-expert users. Consider, for example, the critical problem of which machine learning model we should use for a given dataset. Even in the machine-learning community, this model-selection problem is considered a "black art" [13]. Typically, non-experts are limited to selecting (often blindly) models from off-the-shelf solutions. In this dissertation, we investigate solutions to automate both *active* and *supervised* machine learning. Our core contributions are active learning algorithms that are easy for non-experts to use, but that deliver results competitive with or better than human-expert solutions.

We begin with an investigation on *active search*, a learning paradigm for actively identifying as many members of a given class as possible under a labeling budget. We provide a theoretical hardness result for a particular mathematical formulation of active search, proving that no polynomial-time policy can achieve a constant factor approximation ratio for the expected utility of the optimal policy. After understanding the limitations of this problem,

we propose a novel computationally efficient active search strategy that approximates the optimal policy. Our algorithm is nonmyopic, i.e., always considering the entire remaining search budget. It automatically and dynamically balances exploration and exploitation consistent with the remaining budget — without relying on a parameter to control this tradeoff. Extensive experimental results demonstrate that our approach achieves an exceptional empirical performance on several real-world tasks.

Then we introduce a novel information-theoretic approach for *active model selection*. In this active-learning problem, we seek to gather new data to quickly identify which model — from a predefined, finite list of models — best describes the observed data. Our method is based on maximizing the mutual information between the output variable and the model class. Crucially, we develop an analytical approximation of this criterion for Gaussian process (GP) models with arbitrary observation likelihoods, enabling active structure learning for GPs. This is the first active-model-selection approach that does not require updating each model for every candidate point. As a result, in the experiments presented here, our approach expends seconds per iteration, whereas previous techniques would take hours to finish a single iteration. We apply this framework to active structure discovery, an active model selection problem for regression, and to rapid screening for noise-induced hearing loss (NIHL), a widespread and preventable disability, if diagnosed early.

We proceed in studying *automated model selection* — specifically, we propose a sophisticated method for automatically searching for an appropriate model given an infinite (or very large) space of potential choices. In contrast with active model selection, here we assume that we have already collected all data we can afford, and the goal is to find the best model to account for the gathered data. Our solution is based on Bayesian optimization in model space, where we reason about model evidence as a function to be maximized. We explicitly reason about

the data distribution and how it induces similarity between the model choices in terms of the explanations they can offer for observed data. In this light, we construct a novel kernel between models to explain a given dataset. Our method is capable of finding a model that explains the dataset well without any human assistance, often with fewer computations of model evidence than previous approaches, a claim we demonstrate empirically.

Our last contribution is an automated framework for active learning that can efficiently search for appropriate models in the data-acquisition regime. The key observation is that intelligent sampling policies for active learning benefit from improvements to the data representation. Searching for suitable models is thus critical to the success of active learning, especially when obtaining data is expensive. We apply this general automated active learning pipeline to Bayesian optimization, creating an *automated Bayesian optimization* method. Our approach dynamically selects promising models for explaining the observed data using an ancillary Bayesian optimization routine in the model space. Crucially, we account for the uncertainty in the choice of model; our method is capable of using multiple models to represent its current belief about the latent function and subsequently using this information for decision making. We argue, and demonstrate empirically, that our approach automatically finds suitable models, which ultimately results in more-efficient optimization.

1.1 Declaration of previous publications

All original work in this dissertation was the fruit of collaboration with other researchers, and most contributions were previously published in jointly authored peer-reviewed publications. The following outlines the original work presented here, with references to previous publications and a description of my contribution to them.

Chapter 2. Gaussian process and decision theory

This material is a review of fundamental concepts explored in the remaining of the dissertation.

Chapter 3. Active search

The work in this chapter has appeared in the following publication:

> Jiang, S., Malkomes, G., Converse, G., Shofner, A., Moseley, B., Garnett, R. Efficient nonmyopic active search. International Conference on Machine Learning (ICML), 2017.

Garnett, Moseley, Jiang, and Malkomes conceived and extensively contributed to the theoretical analysis. Malkomes main contribution was to piece together the strategy that led to the final hardness result. Malkomes also developed the demonstration of the nonmyopic behavior of the policy. Jiang worked out all the technical details about pruning, implemented the strategy, and carried out all experiments.

Chapter 4. Active Model Selection

The work in this chapter has appeared in the following publication:

> Gardner, J., Malkomes, G., Garnett, R., Weinberger, K., Barbour, D., Cunningham, J. Bayesian Active Model Selection with an Application to Automated Audiometry. Neural Information Processing Systems (NEURIPS), 2015

Garnett introduced the problem and conceived the idea of using mutual information to active model selection. Malkomes worked out the details and developed the software implementation for regression, and Gardner provided similar contributions to the classification setting. Malkomes carried out the experiments for active structure discovery. Gardner conducted all experiments related to audiometry tests.

Chapter 5. Automated model selection

The work in this chapter has appeared in the following publication:

> Malkomes, G., Schaff, C., Garnett, R. Bayesian optimization for automated model selection. Neural Information Processing Systems (NEURIPS), 2016

Malkomes and Garnett conceived the initial idea. Malkomes and Schaff worked out technical details and equally contributed to software implementation. Initial experimentation and results on predicting model evidence were conducted by Malkomes, and the experiments on model search were mainly carried out by Schaff. Malkomes took the lead in writing the manuscript. All authors provided critical feedback and contributed to the final version.

Chapter 6. Automated active learning

The work on this chapter has appeared in the following publication:

> Malkomes, G. and Garnett, R. Automating Bayesian optimization with Bayesian optimization. Neural Information Processing Systems (NEURIPS), 2018

Malkomes conceived the idea and Garnett refined it with technical contributions. Manuscript, software implementation, experimental designed and results were mainly developed by Malkomes with valuable feedback from Garnett.

Chapter 2

Gaussian process and decision theory

In this chapter, we briefly review the mathematical concepts that will be important for understanding the key contributions of this dissertation. We begin by introducing the Bayesian framework for model selection. Then, we review the important concepts related to the main inference procedure used throughout this dissertation — *Gaussian processes* (GP). We finish with a concise presentation of Bayesian decision theory.

2.1 Bayesian model comparison

Consider supervised learning problems defined on an input space $\mathcal{X}$ and an output space $\mathcal{Y}$. Suppose we are given a set of observations $\mathcal{D} = (\mathbf{X}, \mathbf{y})$, where $\mathbf{X}$ represents the design matrix of independent variables $\mathbf{x}_i \in \mathcal{X}$ and $\mathbf{y}$ the associated vector of dependent variables $y_i = y(\mathbf{x}_i) \in \mathcal{Y}$. Given this training data, we wish to make predictions for a new input location $\mathbf{x}^*$ that we have not seen before. Ultimately, we want to use the finite training set to construct a prediction distribution for all possible input values. Without making assumptions

about the data generation process, this is an ill-posed problem; all predictions agreeing with the observed data would be equally likely [69].

Many methods have been proposed to solve supervised learning problems [69, 63, 61, 7, 30]. Here we appeal to the Bayesian approach. Suppose we have a collection of plausible models $\{\mathcal{M}_i\}_{i=1}^M$ competing to account for the observed data. Each model captures a particular set of assumptions about the data, which in turn may lead to different outcomes. Before diving into the details of how we use these models to make prediction, we focus on the more elemental question of how we compare different models. After all, we will eventually need to know whether we trust our models' predictions.

The answer we seek is simple: we use probabilistic inference. We begin with an initial belief about the relative plausibility of the models, using probabilities to quantify uncertainty, $\{p(\mathcal{M}_i)\}_{i=1}^M$. We also assume each model $\mathcal{M}_i$ provides its own belief about how data looks like, if that model is true — $p(\mathcal{D} \mid \mathcal{M}_i)$. When we observe a particular dataset $\mathcal{D}$, we use Bayes' rule to update our beliefs about the plausibility of each model after observing the actual dataset $\mathcal{D}$ [57]. Thus, the posterior probability of the models given the data is:

$$p(\mathcal{M} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \mathcal{M})p(\mathcal{M})}{p(\mathcal{D})} = \frac{p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})p(\mathcal{M})}{\sum_i p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}_i)p(\mathcal{M}_i)}, \tag{2.1}$$

where $p(\mathcal{M})$ represents the prior probability distribution over the models. The posterior distribution above summarizes the uncertainty after observing the data; computational challenges apart, this is a simple and elegant solution to model comparison.

In practice, computing $p(\mathcal{D} \mid \mathcal{M})$ may be quite difficult. Fix, for instance, a probabilistic model $\mathcal{M}$. Interesting models have a set of unknown parameters (Θ) that can be adjusted to better explain the data. Let θ be an element of this parameter space Θ. Our quantity of

interest is called *model evidence*:

$$p(\mathcal{D} \mid \mathcal{M}) = p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}) = \int p(\mathbf{y} \mid \mathbf{X}, \theta, \mathcal{M}) p(\theta \mid \mathcal{M}) \, \mathrm{d}\theta, \tag{2.2}$$

and it represents the probability of having generating the observed data under the model, marginalized over θ to account for all possible members of that model under a prior $p(\theta \mid \mathcal{M})$ [56, 7]. The problem is: this is often an intractable integral! In the next section, we discuss approximatons for computing this quantity for the class of Gaussian process models.

2.2 Guassian process models

Assume that our observations are generated via a latent function $f \colon \mathcal{X} \to \mathbb{R}$ with a known observation model $p(\mathbf{y} \mid \mathbf{f})$, where $f_i = f(\mathbf{x}_i)$. A standard nonparametric Bayesian approach with such models is to place a Gaussian process (GP) prior distribution on f, $p(f) = \mathcal{GP}(f; \mu, K)$, where $\mu \colon \mathcal{X} \to \mathbb{R}$ is a mean function and $K \colon \mathcal{X}^2 \to R$ is a positive-definite covariance function or kernel [69]. We condition on the observed data to form a posterior distribution $p(f \mid \mathcal{D})$, which is typically an updated Gaussian process (making approximations if necessary). We make predictions at a new input $\mathbf{x}^*$ via the predictive distribution $p(y^* \,|\, \mathbf{x}^*, \mathcal{D}) = \int p(y^* \,|\, f^*, \mathcal{D}) p(f^* \,|\, \mathbf{x}^*, \mathcal{D}) \, \mathrm{d}f^*$, where $f^* = f(\mathbf{x}^*)$.

The mean and kernel functions are parameterized by hyperparameters that we concatenate into a vector θ. Different choices of these hyperparameters imply that the functions drawn from the GP will have a particular frequency, amplitude, and other properties. Together, μ and K define a model parametrized by the hyperparameters θ.

Much attention is paid to learning these hyperparameters in a fixed model class, sometimes under the unfortunate term "model selection." Note, however, that the *structural* (not hyperparameter) choices made in the mean function μ and covariance function K themselves are typically done by selecting (often blindly!) from several off-the-shelf solutions (see, for example, [12, 69]; though also see [13, 90]), and this choice has substantial bearing on the resulting functions f we can model. In Chapter 5, we discuss model selection in more details; for now, recall that we use model to refer to the structural, high-level modeling choice.

In what follows, we present three quantities that are not typically encountered in GP modeling and inference: the hyperparameter posterior $p(\theta \mid \mathcal{D}, \mathcal{M})$, the model evidence $p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})$, and the predictive distribution $p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M})$, where we have marginalized over θ in the latter two quantities. The most-common approaches to GP inference are maximum likelihood–II (MLE) or maximum *a posteriori*–II (MAP) estimation, where we maximize the hyperparameter posterior [89, 69]:[2]

$$\hat{\theta} = \arg\max_{\theta} \log p(\theta \mid \mathcal{D}, \mathcal{M}) = \arg\max_{\theta} \log p(\theta \mid \mathcal{M}) + \log(\mathbf{y} \mid \mathbf{X}, \theta, \mathcal{M}). \tag{2.3}$$

Typically, predictive distributions and other desired quantities are then reported at the MLE/MAP hyperparameters, implicitly making the assumption that $p(\theta \mid \mathcal{D}, \mathcal{M}) \approx \delta(\hat{\theta})$. Although a computationally convenient choice, this does not account for uncertainty in the hyperparameters, which can be nontrivial with small datasets [56]. Furthermore, accounting correctly for model parameter uncertainty is crucial to model comparasion, where it naturally introduces a model-complexity penalty. We discuss less-drastic approximations to these quantities below.

[2]Using a noninformative prior $p(\theta \mid \mathcal{M}) \propto 1$ in the case of maximum likelihood.

Approximating the model evidence and hyperparameter posterior

The model evidence $p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})$ and hyperparameter posterior distribution $p(\theta \mid \mathcal{D}, \mathcal{M})$ are in general intractable for GPs, as there is no conjugate prior distribution $p(\theta \mid \mathcal{M})$ available. Instead, we will use a Laplace approximation, where we make a second-order Taylor expansion of $\log p(\theta \mid \mathcal{D}, \mathcal{M})$ around its mode $\hat{\theta}$ (2.3). The result is a multivariate Gaussian approximation:

$$p(\theta \mid \mathcal{D}, \mathcal{M}) \approx \mathcal{N}(\theta; \hat{\theta}, \Sigma); \qquad \Sigma^{-1} = -\nabla^2 \log p(\theta \mid \mathcal{D}, \mathcal{M})\big|_{\theta=\hat{\theta}}. \tag{2.4}$$

The Laplace approximation also results in an approximation to the model evidence:

$$\log p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}) \approx \log p(\mathbf{y} \mid \mathbf{X}, \hat{\theta}, \mathcal{M}) + \log p(\hat{\theta} \mid \mathcal{M}) - \tfrac{1}{2} \log \det \Sigma^{-1} + \tfrac{d}{2} \log 2\pi, \tag{2.5}$$

where d is the dimension of θ [67, 43]. The Laplace approximation to the model evidence can be interpreted as rewarding explaining the data well while penalizing model complexity. Note that the *Bayesian information criterion* (BIC), commonly used for model selection, can be seen as an approximation to the Laplace approximation [73, 63].

Approximating the predictive distribution

We next consider the predictive distribution:

$$p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}) = \int p(y^* \mid f^*) \underbrace{\int p(f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M}) p(\theta \mid \mathcal{D}, \mathcal{M}) \, \mathrm{d}\theta}_{p(f^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M})} \, \mathrm{d}f^*. \tag{2.6}$$

The posterior $p(f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M})$ in (2.6) is typically a known Gaussian distribution, derived analytically for Gaussian observation likelihoods or approximately using standard approximate GP inference techniques [46, 60]. However, the integral over θ in (2.6) is intractible, even with a Gaussian approximation to the hyperparameter posterior as in (2.4).

Garnett et al. [24] introduced a mechanism for approximately marginalizing GP hyperparameters (called the MGP), which we will adopt here due to its strong empirical performance. The MGP assumes that we have a Gaussian approximation to the hyperparameter posterior, $p(\theta \mid \mathcal{D}, \mathcal{M}) \approx \mathcal{N}(\theta; \hat{\theta}, \Sigma)$.[3] We define the posterior predictive mean and variance functions as

$$\mu^*(\theta) = \mathbb{E}[f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M}]; \qquad \nu^*(\theta) = \mathrm{Var}[f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M}].$$

The MGP works by making an expansion of the predictive distribution around the posterior mean hyperparameters $\hat{\theta}$. The nature of this expansion is chosen so as to match various derivatives of the true predictive distribution; see [24] for details. The posterior distribution of f^* is approximated by

$$p(f^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}) \approx \mathcal{N}\big(f^*; \mu^*(\hat{\theta}), \sigma^2_{\text{MGP}}\big), \tag{2.7}$$

where

$$\sigma^2_{\text{MGP}} = \tfrac{4}{3}\nu^*(\hat{\theta}) + \big[\nabla\mu^*(\hat{\theta})\big]^\top \Sigma \big[\nabla\mu^*(\hat{\theta})\big] + \tfrac{1}{3\nu^*(\hat{\theta})}\big[\nabla\nu^*(\hat{\theta})\big]^\top \Sigma \big[\nabla\nu^*(\hat{\theta})\big]. \tag{2.8}$$

The MGP thus inflates the predictive variance from the the posterior mean hyperparameters $\hat{\theta}$ by a term that is commensurate with the uncertainty in θ, measured by the posterior covariance Σ, and the dependence of the latent predictive mean and variance on θ, measured by the gradients $\nabla\mu^*$ and $\nabla\nu^*$. With the Gaussian approximation in (6.12), the integral in

[3] This is arbitrary and need not be the Laplace approximation in (2.4), so this is a slight abuse of notation.

(2.6) now reduces to integrating the observation likelihood against a univariate Gaussian. This integral is often analytic [69] and at worse requires one-dimensional quadrature.

2.3 Decision theory

"Decision theory is trivial, apart from computational details". MacKay [56] (p. 451)

In previous sections, we discussed probabilistic inference, which the goal is to assign probabilities to a lternative hypothesis in light of observations. Here we turn to a brief discussion of decision theory that, when combined with probabilistic inference, allows us to make optimal decisions in situations involving uncertainty [7].

We begin with a set of potential actions $a \in \mathcal{A}$ that we could make, and a description of the world given by possible states $\mathbf{s} \in \mathcal{S}$. Then, we construct a *utility function* $U(\mathbf{s}, a)$ that quantifies the *benefit* of taking action a when the world is in state $\mathbf{s}$[4]. Bayesian decision theory dictates that we should choose the action maximizing the expected utility:

$$\arg\max_{a \in \mathcal{A}} \int U(\mathbf{s}, a) P(\mathbf{s} \mid a) \, \mathrm{d}\mathbf{s};$$

and that is it. This concise recipe for Bayesian decision theory give us tools to make optimal decision. In practice, however, we will see that there are many computational challenges associated with this simple, yet powerful framework. Next chapter, we apply Bayesian decision theory to *active search*, providing insights on how this framework works in practical settings.

[4]Similarly, one could also consider a *loss function* that summarizes how much you lose by taking action a if the state of world is $\mathbf{s}$. In this case, the goal is to minimize the expected loss, instead of maximizing the expected utility. We follow the optimistic view, presuming that our decisions will lead to gains, not losses

Chapter 3

Active search

In Chapter 1, we introduced active learning as a general sequential decision making problem for obtaining expensive observations. Much of the active learning literature, howerver, has focused on learning machine learning models with fewer training examples. Here, we consider our first special and atypical realization of active learning: the *active search* problem [23].

In active search, we seek to sequentially inspect data so as to discover members of a rare, desired class. The labels are not known *a priori* but can be revealed by querying a costly labeling oracle. Our goal is to design a policy to sequentially query points to find as many valuable points as possible under a labeling budget. Several real-world problems can be naturally posed in terms of active search; drug discovery, fraud detection, and product recommendation are a few examples. The crux of success for active search is the fundamental dilemma between *exploration* and *exploitation*; whether to search for new regions of valuable points (exploration) or take advantage of the currently most-promising regions (exploitation).

Previous work developed policies for active search by appealing to Bayesian decision theory [22, 23]. Garnett et al. [23] derived the optimal policy in this framework with a natural utility function. Not surprisingly, realizing this policy in the general case requires exponential

computation. To overcome this intractability, the authors of that work proposed using myopic lookahead policies in practice, which compute the optimal policy only up to a limited number of steps into the future. This defines a family of policies ranging in complexity from completely greedy one-step lookahead to the optimal policy, which looks ahead to the depletion of the entire budget. The authors demonstrated improved performance on active search over the greedy policy even when looking just two steps into the future. The main limitation of these strategies is that they completely ignore what can happen beyond the chosen horizon, which for typical problems is necessarily limited to $\ell \leq 3$, even with aggressive pruning.

The contributions of this chapter are two-fold. First, we prove that no polynomial time policy for active search can have nontrivial approximation ratio with respect to the optimal policy in terms of expected utility. This extends the result by Garnett et al. [23] that myopic approximations to the optimal policy cannot approximate the optimal policy. The proof of this theorem is constructive, creating a family of explicitly difficult active search instances and showing that no polynomial time algorithm can perform well compared to the optimal (exponential cost) policy on these.

Second, we introduce a novel *nonmyopic* policy for active search that considers not only the potential immediate contribution of each unlabeled point but also its potential impact on the remaining points that could be chosen afterwards. Our policy *automatically* balances exploitation against exploration consistent with the labeling budget without requiring any parameters controlling this tradeoff. We also develop an effective strategy for pruning unlabeled points by bounding their potential impact on the search problem. We empirically show that our method can prune over 92% of the unlabeled points on average for a family of real drug-discovery datasets, achieving massive speedups and keeping the policy computationally tractable, even on large datasets. We compare our method with several baselines by conducting

experiments on numerous real datasets spanning several domains including citation networks, materials science, and drug discovery. Our results thoroughly demonstrate that our policy typically significantly outperforms previously proposed active search approaches.

3.1 Problem formulation

Suppose we are given a finite domain of elements $\mathcal{X} \triangleq \{x_i\}$. We know that there is a rare subset $\mathcal{R} \subset \mathcal{X}$, the members of which are considered valuable, but their identities are unknown *a priori*. We will call the elements of $\mathcal{R}$ *targets* or *positive* items. Assume that there is an oracle that can determine whether a specified element $x \in \mathcal{X}$ is a target, producing the binary output $y \triangleq \mathbb{1}\{x \in \mathcal{R}\}$. The oracle, however, is assumed to be expensive and may only be queried t times. We seek to design a policy to sequentially query elements to maximize the number of targets found.

We will express our preference over different sets of observations $\mathcal{D} \triangleq \big\{(x_i, y_i)\big\}$ through a simple utility:

$$u(\mathcal{D}) \triangleq \textstyle\sum_{y_i \in \mathcal{D}} y_i, \tag{3.1}$$

which simply counts the number of targets in $\mathcal{D}$. Then, the problem is to sequentially construct a set of t observed points $\mathcal{D}$ with the goal of maximizing $u(\mathcal{D})$. Throughout this work, we use a subscript to specify a set of observed data after $i \leq t$ queries, defining $\mathcal{D}_i \triangleq \big\{(x_j, y_j)\big\}_{j=1}^{i}$.

3.2 Bayesian optimal policy

Following previous work, we consider the active search problem in the standard Bayesian framework. Assume we have a probabilistic classification model that provides the posterior probability of a point x belonging to $\mathcal{R}$, given observed data $\mathcal{D}$: $\Pr(y = 1 \mid x, \mathcal{D})$. Recall that we are allowed to perform t labeling queries, and suppose we are at some iteration i for $i \leq t$; having already observed $i - 1$ examples, $\mathcal{D}_{i-1}$. We wish to submit the ith item to the oracle.

Bayesian decision theory (§2.3) compels us to select the item that if we evaluate next maximizes the *expected utility* of the final observed dataset:

$$x_i^* = \underset{x_i \in \mathcal{X} \setminus \mathcal{D}_{i-1}}{\arg\max}\, \mathbb{E}\big[u(\mathcal{D}_t) \mid x_i, \mathcal{D}_{i-1}\big]. \tag{3.2}$$

In other words, we choose a point x_i^* maximizing the expected number of targets found at termination. Unfortunately, as we shall see later, computing $\mathbb{E}\big[u(\mathcal{D}_t) \mid x_i, \mathcal{D}_{i-1}\big]$ is computationally impractical.

To better understand the optimal policy, consider the case $i = t$, so we already have $t - 1$ observations $\mathcal{D}_{t-1}$ and there is only one more query left. The expected utility is

$$\mathbb{E}\big[u(\mathcal{D}_t) \mid x_t, \mathcal{D}_{t-1}\big] = \textstyle\sum_{y_t} u(\mathcal{D}_t) \Pr(y_t \mid x_t, \mathcal{D}_{t-1}) = u(\mathcal{D}_{t-1}) + \Pr(y_t = 1 \mid x_t, \mathcal{D}_{t-1}). \tag{3.3}$$

Note $u(\mathcal{D}_{t-1})$ is a constant, since $\mathcal{D}_{t-1}$ was already observed. Thus, when there is one query remaining, the optimal decision is to greedily choose the remaining point with maximum probability of being a target.

When two or more queries are left, the optimal policy is not as trivial. The challenge is that after the first choice, the probability model changes, affecting all future decisions. Below, we show the expected utility for $i = t - 1$.

$$\mathbb{E}\big[u(\mathcal{D}_t) \mid x_{t-1}, \mathcal{D}_{t-2}\big] = u(\mathcal{D}_{t-2}) + \Pr(y_{t-1} = 1 \mid x_{t-1}, \mathcal{D}_{t-2}) + \mathbb{E}_{y_{t-1}}\big[\max_{x_t} \Pr(y_t = 1 \mid x_t, \mathcal{D}_{t-1})\big]. \quad (3.4)$$

This expression has an intuitive interpretation. First, we have the reward for the data already observed, $u(\mathcal{D}_{t-2})$. The second term is the expected reward contribution from the point x_{t-1} under consideration, $\Pr(y_{t-1} = 1 \mid x_{t-1}, \mathcal{D}_{t-2})$. Finally, the last term is the expected future reward, which is the expected reward to be gathered on the next step; from our previous analysis, we know that this will be maximized by a greedy selection (3.3). These latter two terms can be interpreted as encouraging exploitation and exploration, respectively, with the optimal second-to-last query.

In general, we can compute expected utility (3.2) at time $i \leq t$ recursively as [23]:

$$\mathbb{E}\big[u(\mathcal{D}_t) \mid x_i, \mathcal{D}_{i-1}\big] = u(\mathcal{D}_{i-1}) + \underbrace{\Pr(y_i = 1 \mid x_i, \mathcal{D}_{i-1})}_{\text{exploitation, } < 1} + \underbrace{\mathbb{E}_{y_i}\Big[\max_{x'} \mathbb{E}\big[u(\mathcal{D}_t \setminus \mathcal{D}_i) \mid x', \mathcal{D}_i\big]\Big]}_{\text{exploration, } < t-i}. \quad (3.5)$$

It is easy to show that the time complexity for computing Eq. (3.5) is $\mathcal{O}\big((2n)^\ell\big)$, where $\ell = t - i + 1$ is the lookahead and n is the total number of unlabeled points.

This exponential running time complexity makes the Bayeisan optimal policy infeasible to compute, even for small-scale applications. A typical workaround is to pretend there are only

a few steps left in the search problem at each iteration, and sequentially apply a myopic policy (e.g., (3.3) or (3.4)). We will refer to these policies as the one-step and two-step myopic policies, respectively, and more generally to the ℓ-step myopic policy, with $\ell < t - i + 1$.

Since these myopic approaches cannot plan more than ℓ steps ahead, they can underestimate the potential benefit of exploration. In particular, the potential magnitude of the exploration term in (3.5) depends linearly on the budget, whereas in an ℓ-step myopic policy, the magnitude of the equivalent term can go no higher than a fixed upper bound of ℓ. In fact, Garnett et al. [23] showed via an explicit construction that the expected performance of the ℓ-step policy can be arbitrarily worse than any m-step policy with $\ell < m$, exploiting this inability to "see past" the horizon. When following this suggestion, we must thus trade off the potential benefits of nonmyopia and the rapidly increasing computational burden of lookahead when choosing a policy.

3.3 Hardness of approximation

We extend the above hardness result to show that no polynomial-time active search policy can be an approximation algorithm with respect to the optimal policy, in terms of expected utility. In particular, under the assumption that algorithms only have access to a unit cost conditional marginal probability $\Pr(y = 1 \mid x, \mathcal{D})$ for any x and $\mathcal{D}$, where $|\mathcal{D}|$ is less than the budget,[5] then:

Theorem 1. *No polynomial-time policy for active search can have expected utility within a constant factor of the optimal policy.*

[5]The optimal policy operates under these restrictions.

We prove this theorem in the appendix. The main idea is to construct a class of instances where a small "secret" set of elements encodes the locations of a large "treasure" of targets. The probability of revealing the treasure is vanishingly small without discovering the secret set; however, it is extremely unlikely to observe any information about this secret set with polynomial-time effort.

Despite the negative result of Theorem 1, we may still search for policies that are empirically effective on real problems. In the next section, we propose a novel alternative approximation to the optimal policy (3.2) that is nonmyopic, computationally efficient, and shows impressive empirical performance.

3.4 Efficient nonmyopic active search

We have seen above how to myopically approximate the Bayesian optimal policy using an ℓ-step-lookahead approximate policy (3.5). Such an approximation, however, effectively assumes that the search procedure will terminate after the next ℓ evaluations, which does not reward exploratory behavior that improves performance beyond that horizon. We propose to continue to exactly compute the expected utility to some fixed horizon, but to approximate the remainder of the search differently. We will approximate the expected utility from any remaining portion of the search by assuming that any remaining points, $\{x_{i+1}, x_{i+2}, \ldots, x_t\}$, in our budget will be selected *simultaneously* in one big batch. One rationale is if we assume that after observing $\mathcal{D}_i$, the labels of all remaining unlabeled points are conditionally independent, then this approximation recovers the Bayesian optimal policy exactly. By exploiting linearity of expectation, it is easy to work out the optimal policy for selecting such a simultaneous batch observation: we simply select the points with the highest probability of being valuable.

The resulting approximation is

$$\max_{x'} \mathbb{E}\big[u(\mathcal{D}_t \backslash \mathcal{D}_i) \mid x', \mathcal{D}_i\big] \approx \textstyle\sum'_{t-i} \Pr(y = 1 \mid x, \mathcal{D}_i), \tag{3.6}$$

where the summation-with-prime symbol $\sum'_k$ indicates that we only sum the largest k values.

Our proposed policy selects points by maximizing the approximate final expected utility using:

$$\mathbb{E}\big[u(\mathcal{D}_t) \mid x_i, \mathcal{D}_{i-1}\big] \approx u(\mathcal{D}_{i-1}) + \Pr(y_i = 1 \mid x_i, \mathcal{D}_{i-1}) + \underbrace{\mathbb{E}_{y_i}\Big[\textstyle\sum'_{t-i} \Pr\big(y = 1 \mid x, \mathcal{D}_i\big)\Big]}_{\text{exploration, } < t-i}. \tag{3.7}$$

We will call this policy *efficient nonmyopic search*. As in the optimal policy, we can interpret (3.7) naturally as rewarding both exploitation and exploration, where the exploration benefit is judged by a point's capability to increase the top probabilities among currently unlabeled points. We note further that in (3.7) the reward for exploration *naturally decreases over time* as the budget is depleted, exactly as in the optimal policy. In particular, the very last point x_t is chosen greedily by maximizing probability, agreeing with the true optimal policy. The second-to-last point is also guaranteed to match the optimal policy.

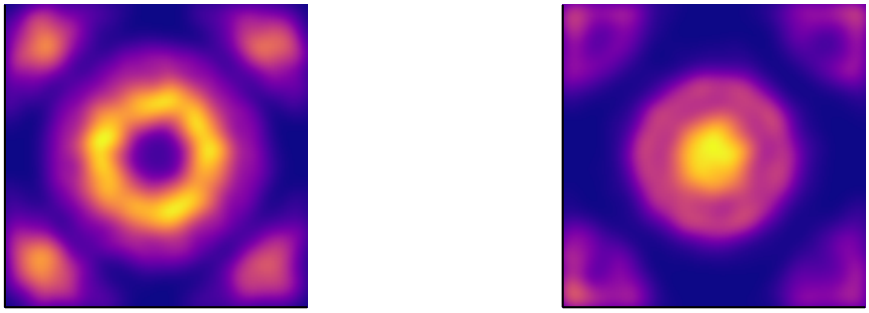
Note that we may also use the approximation in (3.6) as part of a finite-horizon lookahead with $\ell > 1$, producing a family of increasingly expensive but higher-fidelity approximations to the optimal policy, all retaining the same budget consciousness. The approximation in (3.7) is equivalent to a one-step maximization of (3.6). We will see in our experiments that this is often enough to show massive gains in performance, and that even this policy shows

clear awareness of the remaining budget throughout the search process, automatically and dynamically trading off exploration and exploitation.

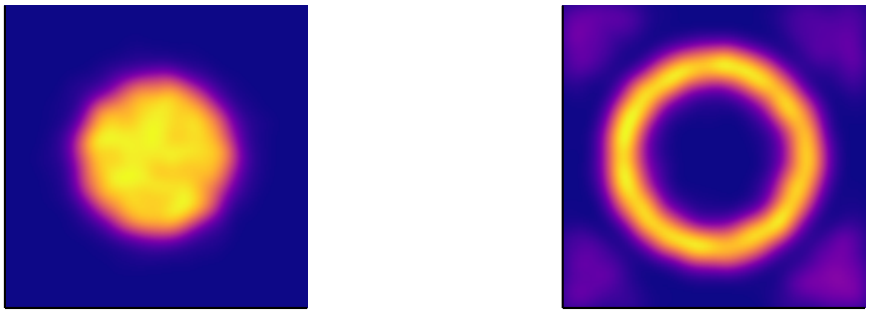
Nonmyopic Behavior

A fundamental difference between our score function (3.7) from others is the ability of automatically balance exploration against exploitation. This behavior emerges because we are able to (approximately) incorporating the impact of the remaining observations when each candidate point is evaluated. To illustrate the nonmyopic behavior of our policy, we have adapted the toy example presented by Garnett et al. [23]. Let $I \triangleq [0,1]^2$ be the unit square. We repeated the following experiment 100 times. We selected 500 points iid uniformly at random from I to form the input space $\mathcal{X}$. We create an active search problem by defining the set of targets $\mathcal{R} \subseteq \mathcal{X}$ to be all points within Euclidean distance $1/4$ from either the center or any corner of I. We took the closest point to the center (always a target) as an initial training set. We then applied ENS and the two-step-lookahead (3.4) policies to sequentially select 200 further points for labeling.

Figure 3.1 shows a kernel density estimate of the distribution of locations selected by both methods during two time intervals. Figures 3.1(a–b) correspond to our method; Figures 3.1(c–d) to two-step lookahead. Figures 3.1(a, c) consider the distribution of the first 100 selected locations; Figures 3.1(b, d) consider the last 100. The qualitative difference between these strategies is clear. The myopic policy focused on collecting all targets around the center (Figure 3.1(c)), whereas our policy explores the boundaries of the center clump with considerable intensity, as well as some of the corners (Figure 3.1(a)). As a result, our policy is capable of finding some of targets in the corners, whereas two-step lookahead hardly ever can (Figure 3.1(d)). We can also see that the highest probability mass in Figure 3.1(b) is



(a) (b)



(c) (d)

Figure 3.1: Kernel density estimates of the distribution of points chosen by ENS (top) and 2-step lookahead (bottom) during two different time intervals. The figures on the left show the kernel density estimates for the first 100 locations; the figures on the right, the last 100 chosen locations.

the center, which shows that our policy typically saves many high-probability points until the end. On average, the ENS policy found about 40 more targets at termination than the two-step-lookahead policy.

Implementation and time complexity

The complexity of our policy (3.7) is $\mathcal{O}\big(n\big(2(n+n\log n)\big)\big) = \mathcal{O}(n^2 \log n)$, for $n = |\mathcal{X}|$, because we need to compute the approximate expected utility for all n points, evaluate an expectation over its label, conditioning the model and sorting the posterior probabilities in the expectation. However, for some classification models $\Pr(y = 1 \mid x, \mathcal{D})$, observing one point will only affect

the probabilities on a small portion of the other points (e.g., in a k-nn model). We can exploit such structure to reduce the complexity of our method by avoiding unnecessary computation.

Suppose that after observing a point we only need to update the probabilities of at-most m other points. We can avoid repeatedly sorting the probabilities of every unlabeled point when computing the score of each candidate point. Once the *current* probabilities are sorted ($\mathcal{O}(n \log n)$), we only need to update m probabilities and perform a sort algorithm ($\mathcal{O}(m \log m)$); now we can merge both lists to get the top $t - i$ posterior probabilities in time $\mathcal{O}(t - i)$, where i is the index of current iteration. In summary, we can reduce the computational complexity to $\mathcal{O}\big(n(\log n + m \log m + t)\big)$. This is about the same complexity as two-step lookahead, which is $\mathcal{O}\big(n(\log n + m)\big)$ using a similar implementation.

Pruning the search space

To further reduce the computational complexity, we can use a similar strategy as suggested by Garnett et al. [23] to bound the score function (3.7) and prune points that cannot possibly maximize our score. We consider the same two assumptions proposed by these authors. First, observing a new negative point will not raise the probability of any other point being a target. Second, we are able to bound the maximum probability of the unlabeled points after conditioning on a given number of additional targets; that is, we assume there is a function $p^*(n, \mathcal{D})$ such that

$$p^*(n, \mathcal{D}) \geq \max_{x \in \mathcal{X} \setminus \mathcal{D}} \Pr(y = 1 \mid x, \mathcal{D} \cup \mathcal{D}', \textstyle\sum_{y' \in \mathcal{D}'} y' \leq n).$$

That is, the probability of any unlabeled point can become at most $p^*(n, \mathcal{D})$ after further conditioning on n or fewer additional target points.

Consider an unlabeled point x at time i, and define $\pi(x) = \Pr(y = 1 \mid x, \mathcal{D}_i)$ for the remainder of this discussion. The score (3.7), denoted $f(x)$ here for simplicity, can be upper bounded by

$$f(x) \leq \pi \cdot \big(1 + (t-i)p^*(1, \mathcal{D}_i)\big) + \\ (1-\pi) \cdot \big(\textstyle\sum_{t-i}' \Pr(y' = 1 \mid x', \mathcal{D}_i)\big) \triangleq U(\pi).$$

Note this upper bound is only a function of the current probability π. Let x^+ be the point with maximum probability. Then $f(x^+)$ is certainly a lower bound of $\max_x f(x)$. Hence, those points satisfying $U\big(\pi(x)\big) < f(x^+)$ can be safely removed from consideration. Solving this inequality, we have

$$\pi(x) < \frac{f(x^+) - \sum_{t-i}' \Pr(y' = 1 \mid x', \mathcal{D})}{1 + p^*(1, \mathcal{D})(t-i) - \sum_{t-i}' \Pr(y' = 1 \mid x', \mathcal{D})}. \tag{3.8}$$

Then, all points with current probability lower than the RHS of (3.8) can be removed from consideration. We will show empirically that a large fraction of points can often be pruned on massive datasets.

3.5 Related Work

Our method falls into the broader framework of active learning. The particular setting of finding elements of a valuable class is rather unusual in active learning, which typically considers the goal of training a high-fidelity model [47]. For an exhaustive introduction to active learning, we refer the reader to Settles [74].

The multi-armed bandit (MAB) problem shares some similarities with active search, where selecting an item can understood as "pulling an arm." However, in active search the items are correlated, and, critically, they can never be played twice. Despite the difference, we note that our ENS policy is somewhat similar to the *knowledge gradient* policy introduced by Frazier et al. [15]. Active search can be seen as a special case of *Bayesian optimization* [8, 78] with binary observations and cumulative reward. Several nonmyopic policies have been proposed for Bayesian optimization in the regression setting (e.g., Ling et al. [49]), and our method is spiritually similar to the recently propopsed GLASSES algorithm [27].

Vanchinathan et al. [85] proposed a method called GP-SELECT to solve a class of problems the authors call "adaptive valuable item discovery," which generalizes active search to the regression setting. GP-SELECT employs a Gaussian process regression model in a manner inspired by the Gaussian process upper confidence bound (GP-UCB) algorithm [80]. A parameter must be specified to balance exploration and exploitation, whereas our method automatically and dynamically trades off these quantities. The method is also critically tied to Gaussian process regression as the underlying model, which is inappropriate for classification. Our decision-theoretic approach does not make assumptions about the classification model.

Active search can also be seen as a special case of (partially observable) Markov decision processes ((PO)MDPs), for which there are known hardness results. Sabbadin et al. [70], for example, defined the class of so-called "purely epistemic" MDPs (EMDPs), where the state does not evolve over time. The authors showed that the optimal policy for these problems cannot admit polynomial-time constant approximations. These hardness results, for the very rich class of EMDPs are not trivially transferred to the more-specific active search problem.

Our proposed approximation is similar in nature to the active search policy proposed by Wang et al. [87], which only considered the effect of raising probabilities after observing a

positive label, and did not consider the budget. Rather, the proposed score always encourages maximal exploration, in opposition to the optimal policy.

There has been some attention to active search in the graph setting where the input domain $\mathcal{X}$ is the nodes of a graph [22, 87, 66, 54]. Our method does not restrict the input space. Further, the classification models used in these settings are often difficult to scale to large datasets, e.g., requiring the pseudoinverse of the graph Laplacian.

Finally, variations on the active search problem have also been considered. Ma et al. [53] proposed the *active area search* problem, wherein a continuous function is sampled to discover regions with large mean value, and Ma et al. [55] extended this idea to define the more-general *active pointillistic pattern search* problem. These settings do not allow querying for labels directly and offer no insight to the core active search problem.

3.6 Experimental results

We implemented our approximation to the Bayesian optimal policy with the MATLAB active learning toolbox,[6] and have compared the performance of our proposed ENS policy with several baselines. First we compare with the myopic one-step (greedy) and two-step approximations to the Bayesian optimal policy, presented in (3.3–3.4). In [23, 25], there is thoroughly comparasion between the one-step and two-step policies; the authors observed that the less-myopic two-step algorithm usually performs better in terms of targets found, as one would expect. In our experiments we will mainly focus on comparing our algorithm with the myopic two-step approximate policy since it represents the state-of-the-art approach.

[6] https://github.com/rmgarnett/active_learning

We also consider a simple baseline which we call RANDOM-GREEDY (RG). Here we randomly select points to query (exploration) during the first half of the budget, and select the remainder using greedy selection (exploitation). Although naïve, this policy adapts to the budget.

We further compare with the score function proposed by Wang et al. [87], which we refer to as IMS:

$$\text{IMS}(x) = \Pr(y = 1 \mid x, \mathcal{D})\big(1 + \alpha\, \text{IM}(x)\big); \tag{3.9}$$

where $\text{IM}(x)$ measures the "expected impact", the sum of the raised probabilities x results in if it is positive. Note that it is difficult to determine the tradeoff parameter α without (expensive) cross validation. The empirical results in [87] indicate that $\alpha = 10^{-4}$ performs well on average; we will fix this value in our experiments.

Finally, we have also considered the following UCB-style [3] score function: $\alpha(x, \mathcal{D}) = \pi + \gamma\sqrt{\pi(1-\pi)}$, where $\pi = \Pr(y = 1 \mid x, \mathcal{D})$ and γ is a tradeoff parameter. The UCB score function is very popular and is the essence of the methods in [85, 80] developed for Gaussian processes, including GP-SELECT. We considered various γ values and our experiments show that it is no better than two-step lookahead, so we ommit these results in this presentation.

The probability model $\Pr(y = 1 \mid x, \mathcal{D})$ we will adopt is the k-nearest-neighbor (k-NN) classifier as described in Section 7 of [23]. This model, while being rather simple, shows reasonable generalization error, is nonparameteric, and can be rapidly updated given new training data, an important property in the active setting we consider here. We will also adopt the probability bound (3.8) for this model described in that work. Note IMS was proposed together (but orthogonally) with a graph model for the probability, which is computationally infeasible ($\mathcal{O}(n^3)$) for our datasets. So we also use k-NN model for IMS.

Table 3.1: CiteSeer^x (left) and BMG (right) data: Average number of targets found by the one- and two-step myopic policies and ENS with different five budgets, varying from 100 to 900, at specific time steps. Best method performance at each time waypoint is in bold.

CiteSeer^x data

	query number				
policy	100	300	500	700	900
RG	19.7	60.0	104	140	176
IMS	26.3	86.3	147	214	281
one-step	25.5	80.5	141	209	273
two-step	24.9	89.8	155	220	287
ENS–900	25.9	94.3	163	239	**308**
ENS–700	28.0	105	188	**259**	
ENS–500	28.7	**112**	**189**		
ENS–300	26.4	105			
ENS–100	**30.7**				

BMG data

	query number				
policy	100	300	500	700	900
RG	48.6	144	243	336	427
IMS	93.6	276	451	629	799
one-step	90.8	273	450	633	798
two-step	91.0	273	452	632	802
ENS–900	89.0	270	453	635	**815**
ENS–700	91.3	276	460	**645**	
ENS–500	92.4	279	**466**		
ENS–300	92.8	**279**			
ENS–100	**94.5**				

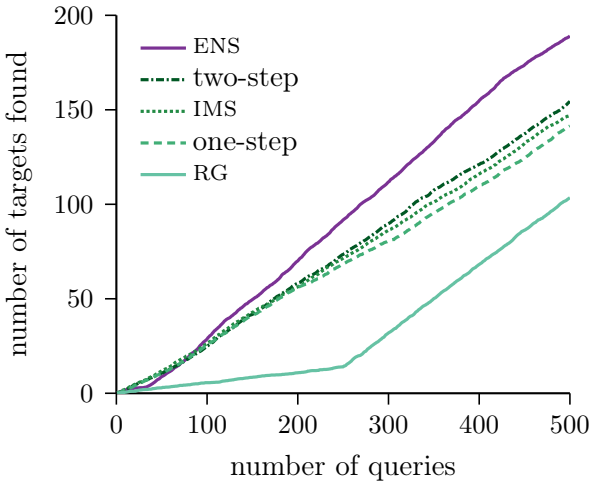


Figure 3.2: The learning curve of our policy and other baselines on the CiteSeer^x dataset.

CiteSeer^x Data

For our first real data experiment, we consider a subset of the CiteSeer^x citation network, first described in [23]. This dataset comprises 39 788 computer science papers published in the top-50 most-popular computer science venues. We form an undirected citation network from these papers. The target class is papers published in the NEURIPS proceedings; there are 2 190 such papers, 5.5% of the whole dataset. Note that distinguishing NEURIPS papers in the citation network is not an easy task, because many other highly related venues such as ICML, AAAI, IJCAI, etc. are also among the most-popular venues. A feature vector for each paper is computed by performing graph principal component analysis [14] on the citation network and retaining the first 20 principal components.

We select a single target (i.e., a NEURIPS paper) uniformly at random to form an initial training set. The budget is set to $t = 500$, and we use $k = 50$ in the k-NN model. These parameters match the choices in [23]. We use each policy to sequentially select t papers for labeling. The experiment was repeated 20 times, varying the initial seed target. Figure 3.2 shows the average number of targets found for each method as a function of the number of queries. We first observe that the ranking of the performance is ENS, two-step, IMS, one-step, and RG, and our policy outperforms the two-step policy in this task by a large margin. The mean difference in number of targets found at termination vs. two-step is 34.6 (189 vs. 155), an improvement on average of 22%. A two-sided paired t-test testing the hypothesis that the average difference of targets found is zero returns a p-value of $p < 10^{-4}$, and a 95% confidence interval on the increase in number of targets found of $[19.80, 49.30]$.

Another interesting observation is that during the initial $\sim$80 queries, ENS actually performs *worse* on average than all baseline policies except RG, after which it quickly outperforms

them. This feature perfectly illustrates an automatic exploration–exploitation transition made by our policy. As we are always cognizant of our budget, we spend the initial stage thoroughly exploring the domain, without immediate reward. Once complete, we exploit what we learned for the remainder of the budget. This tradeoff happens automatically and without any need for an explicit two-stage approach or arbitrary tuning parameters.

Varying the Budget. A distinguishing feature of our method is that it always takes the remaining budget into consideration when selecting a point, so we would expect different behavior with different budgets. We repeated the above experiment for budgets $t \in \{100, 300, 500, 700, 900\}$, and report in Table 3.1 the average number of targets found at these time points for each method. We have the following observations from the table. First, ENS performs better than all other baseline policies for every budget. Second, ENS is able to adapt to the specified budget. For example, when comparing performance after 100 queries, ENS–100 has located many more targets than the ENS methods with greater budgets, which at that time are still strongly rewarding exploration. A similar pattern holds when comparing other pairs of ENS variations, with one minor exception.

Finding Bulk Metallic Glasses

Our next dataset considers an application from materials science: discovering novel alloys forming bulk metallic glasses (BMGs). BMGs have numerous desirable properties, including high toughness and good wear resistance compared to crystalline alloys. We compiled a database of 118 678 known alloys from the materials literature (e.g., [41, 1]), an extension of the dataset from [88]. Of these, 4 746 ($\sim$4%) are known to exhibit glass-forming ability, which we define to be targets. We conduct the same experiments described for the CiteSeer^x data above and show the results in Table 3.1. We can see the results again demonstrate

Table 3.2: Number of active compounds found by various active search policies at termination for each fingerprint, averaged over 120 active classes and 20 experiments. Also shown is the difference of performance between ENS and two-step lookahead and the results of the corresponding paired t-test.

	policy					t-test results			
fingerprint	100-NN	RG	one-step	two-step	ENS	difference	p-value	95% CI	
ECFP4	189	189	289	297	**303**	5.29	1.76×10^{-3}	2.01	8.56
GpiDAPH3	134	170	255	261	**276**	14.8	3.90×10^{-13}	11.2	18.4

our policy's superior performance over all other methods, and its ability of adapting to the remaining budget.

Virtual Drug Screening Data

We further conduct experiments on a massive database of chemoinformatic data. The basic setting is to screen a large database of compounds searching for those that show binding activity against some biological target. This is a basic component of drug-discovery pipelines. The dataset comprises 120 activity classes of human biological importance selected from the Binding DB [50] database. For each activity class, there are a small number of compounds with significant binding activity; the number of targets varies from 200 to 1 488 across the activity classes. From these we define 120 different active search problems. There are also 100 000 presumed inactive compounds selected at random from the ZINC database [81]; these are used as a shared negative class for each of these problems. For each compound, we consider two different feature representations, also known as chemoinformatic fingerprints, called ECFP4 and GpiDAPH3. These fingerprints are binary vectors encoding the relevant

chemical characteristics of the compounds; see [25] for more details.[7] So in total we have 240 active search problems, each with more than 100 000 points, and with targets less than 1.5%.

As is standard in this setting, we compute fingerprint similarities via the Jaccard index [37], which are used to define the weight matrix of the k-NN model from above, setting $k = 100$ for all the experiments. For active search policies, we again randomly select one positive as the initial training set, and sequentially query $t = 500$ further points. We also report the performance of a baseline where we randomly sample a stratified sample of size 5% of the database ($\sim$5 000 points, more than 10 times the budget of the active search policies). From this sample, we train the same k-NN model, compute the active probability of the remaining points, and query the 500 points with the highest posterior activity probability. All experiments were repeated 20 times, varying the initial training point. Note we did not test IMS on these data due to computational expense. Our policy nominally has higher time complexity, but our pruning strategy can reduce the computation significantly in practice, as we show in Section 3.6.

Table 3.2 summarizes the results. First we notice that all active search policies perform much better than the recall of a simple classification algorithm, even though they observe less than one-tenth the data. Interestingly, even the naïve random-greedy (RG) policy performs much better than this baseline, albeit much worse than other active search policies. The two-step policy is again better than the greedy policy for both fingerprints, which is consistent with the results reported in [25]. The ENS policy performs significantly better than two-step lookahead; a two-sided paired t-test overwhelmingly rejects the hypothesis that the performance at termination is equal in both cases.

[7] We did not conduct experiments on the MACCS fingerprint. It was inferior in the findings of Garnett et al. [25]. A reviewer of [37] noted that it is no longer used, due to clear underperformance compared to, e.g., ECFP4 and GPiDAPH3.

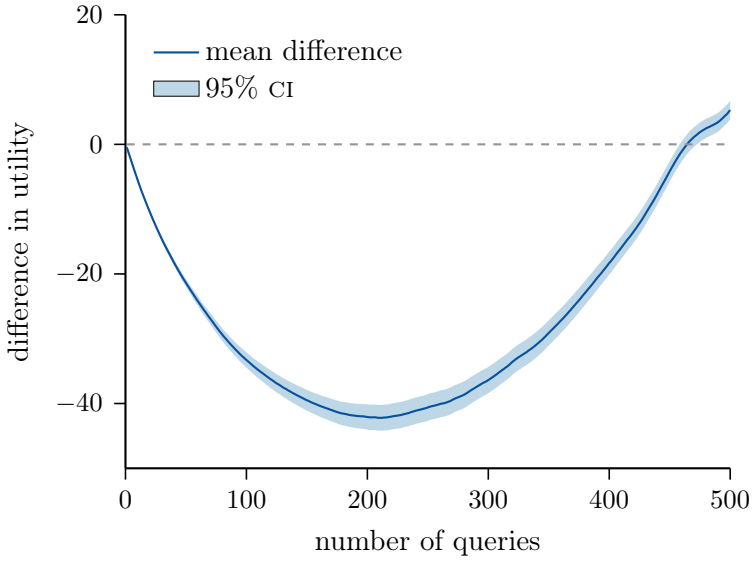


Figure 3.3: The average difference in cumulative targets found between ENS and the two-step policy, averaged over 120 activity classes and 20 experiments on the ECFP4 fingerprint.

Figure 3.3 shows the mean difference in cumulative targets found between ENS and the two-step policy for the ECFP4 fingerprint. Again, we very clearly observe the automatic trade-off between exploration and exploitation by our method. In the initial stage of the search, we explore the space without much initial reward, but around query 200, our algorithm switches automatically to exploitation, outperforming the myopic policy significantly at termination. The mean difference curves for the other fingerprint is similar

Effect of Pruning

To investigate how pruning can improve the efficiency of computing the policy, we computed the average number of pruned points across all $120 \times 20 \times 500 = 3\,000\,000$ iterations of active search, for each fingerprint. On average about 93% of the unlabeled points are pruned, as detailed in Table 3.3, dramatically improving the computational efficiency by approximately

Table 3.3: Average number of pruned points in each iteration for the three chemical datasets.

fingerprint	# pruned	# total	pruned %
ECFP4	94 995	100 518	94.5%
GPiDAPH3	93 173	100 518	92.7 %

a corresponding linear factor. The time for each experiment was effectively reduced from on the order of one day to that of one hour.

3.7 Conclusion

In this chapter we proved the theoretical hardness of active search and proposed an well-motivated and empirically better-performing policy for solving this problem. In particular, we proved that no polynomial-time algorithm can approximate the expected utility of the optimal policy within a constant approximation ratio. We then proposed a novel method, efficient nonmyopic search (ENS), for the active search problem. Our method approximates the Bayesian optimal policy by computing, conditioned on the location of the next point, how many targets are expected at termination, if the remaining budget is spent simultaneously. By taking the remaining budget into consideration in each step, we are able to automatically balance exploration and exploitation. We also derived an effective pruning strategy that can reduce the number of candidate points we must consider at each step, which can further improve the efficiency dramatically in practice. We conducted a massive empirical evaluation that clearly demonstrated superior overall performance on various domains, as well as our automatic balance between exploration and exploitation.

Chapter 4

Bayesian Active Model Selection

Personalized medicine has long been a critical application area for machine learning [42, 72, 5], in which automated decision making and diagnosis are key components. Beyond improving quality of life, machine learning in diagnostic settings is particularly important because collecting additional medical data often incurs significant financial burden, time cost, and patient discomfort. In machine learning one often considers this problem to be one of active feature selection: acquiring each new feature (e.g., a blood test) incurs some cost, but will, with hope, better inform diagnosis, treatment, and prognosis. By careful analysis, we may optimize this trade off.

However, many settings do not involve feature selection, but rather involve querying a sample space to discriminate between different candidate models that compete to account for the observed data. A particular, clarifying example that motivates this work is *noise-induced hearing loss* (NIHL), a prevalent disorder affecting 26 million working-age adults in the United States alone [77] and affecting over half of workers in particular occupations such as mining and construction. Most tragically, NIHL is preventable with simple, low-cost solutions (e.g., earplugs), if diagnosed early.

To be tested for NIHL, patients must complete a time-consuming audiometric exam that presents a series of tones at various frequencies and intensities; at each tone the patient indicates whether he/she hears the tone [9, 36, 11]. From the responses, the clinician infers the patient's audible threshold on a set of discrete frequencies (the *audiogram*); this process requires the delivery of up to hundreds of tones. Audiologists scan the audiogram for a hearing deficit with a characteristic *notch* shape—a narrow band that can be anywhere in the frequency domain that is indicative of NIHL. Unfortunately, at early stages of the disorder, notches can be small enough that they are undetectable in a standard audiogram, leaving many cases undiagnosed until the condition has become severe. Increasing audiogram resolution would require higher sample counts (more presented tones) and thus only lengthen an already burdensome procedure. We present here a better approach.

The NIHL diagnostic challenge is not one of feature selection (choosing the next test to run and classifying the result), but rather of model selection: is this patient's hearing better described by a normal hearing model, or a notched NIHL model? We propose a novel *active model selection* algorithm to make the NIHL diagnosis in as few tones as possible, which directly reflects the time and personnel resources required to make accurate diagnoses in large populations. This is a model-selection problem in the truest sense: a diagnosis corresponds to selecting between two or more sets of indexed probability distributions (models), rather than the more-common misnomer of choosing an index from within a model (i.e., hyperparameter optimization). In the NIHL case this distinction is critical. We are choosing between two models, the set of possible NIHL hearing functions and the set of normal hearing functions. This approach suggests a very different and more direct algorithm than first learning the most likely NIHL function and then accepting or rejecting it as different from normal, the standard approach.

In this chapter, we make the following contributions: first, we design a completely general active-model-selection method based on maximizing the mutual information between the response to a tone and the posterior on the model class. Critically, we develop an analytical approximation of this criterion for Gaussian process (GP) models with arbitrary observation likelihoods, enabling active structure learning for GPs. Second, we extend the work of Gardner et al. [17] (which uses active learning to speed up audiogram inference) to the broader question of identifying *which model*—normal or NIHL—best fits a given patient. Finally, we develop a novel GP prior mean that parameterizes notched hearing loss for NIHL patients. To our knowledge, this is the first work with an active model-selection approach that does not require updating each model for every candidate point, allowing audiometric diagnosis of NIHL to be performed in a real application. Finally, using patient data from a clinical trial, we show empirically that our method typically automatically detects simulated noise-induced hearing loss with fewer than 15 query tones. This is vastly fewer than the number required to infer a conventional audiogram or even an actively learned audiogram [17], highlighting the importance of both the active-learning approach and our focus on model selection.

4.1 Active Bayesian model selection

Suppose that we have a mechanism for actively selecting new data—choosing $\mathbf{x}^* \in \mathcal{X}$ and observing $y^* = y(\mathbf{x}^*)$—to add to our dataset $\mathcal{D} = (\mathbf{X}, \mathbf{y})$, in order to better distinguish the candidate models $\{\mathcal{M}_i\}$. After making this observation, we will form an augmented dataset $\mathcal{D}' = \mathcal{D} \cup \left\{(\mathbf{x}^*, y^*)\right\}$, from which we can recompute a new model posterior $p(\mathcal{M} \mid \mathcal{D}')$.

An approach motivated by information theory is to select the location maximizing the *mutual information* between the observation value y^* and the unknown model:

$$I(y^*; \mathcal{M} \mid \mathbf{x}^*, \mathcal{D}) = H[\mathcal{M} \mid \mathcal{D}] - \mathbb{E}_{y^*}\big[H[\mathcal{M} \mid \mathcal{D}']\big] \tag{4.1}$$

$$= H[y^* \mid \mathbf{x}^*, \mathcal{D}] - \mathbb{E}_{\mathcal{M}}\big[H[y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}]\big], \tag{4.2}$$

where H indicates (differential) entropy. Whereas Equation (4.1) is computationally problematic (involving costly model retraining), the equivalent expression (4.2) is typically more tractable, has been applied fruitfully in various active-learning settings [32, 24, 17, 31, 33], and requires only computing the differential entropy of the model-marginal predictive distribution:

$$p(y^* \mid \mathbf{x}^*, \mathcal{D}) = \sum_{i=1}^{M} p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}_i) p(\mathcal{M}_i \mid \mathcal{D}) \tag{4.3}$$

and the model-conditional predictive distributions $\big\{p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}_i)\big\}$ with all models trained with the currently available data. In contrast to (4.1), this does not involve any retraining cost. Although computing the entropy in (4.3) might be problematic, we note that this is a one-dimensional integral that can easily be resolved with quadrature. Our proposed approach, which we call *Bayesian active model selection* (BAMS) is then to compute, for each candidate location $\mathbf{x}^*$, the mutual information between y^* and the unknown model, and query where this is maximized:

$$\arg\max_{\mathbf{x}^*} I(y^*; \mathcal{M} \mid \mathbf{x}^*, \mathcal{D}). \tag{4.4}$$

In the next section, we describe in details how to implement our approach using Gaussian processes. It is worthy to mention that the framework is general enough to capture any probabilistic model, and, to the best of our knowledge, it is the first method which does not required re-training before probing the next x^*.

4.2 Related work

Although active learning and model selection have been widely investigated, active model selection has received comparatively less attention. Ali et al. [2] proposed an active learning model selection method that requires leave-two-out cross validation when evaluating each candidate $\mathbf{x}^*$, requiring $\mathcal{O}\big(B^2M|\mathbf{X}^*|\big)$ model updates per iteration, where B is the total budget. Kulick et al. [44] also considered an information-theoretic approach to active model selection, suggesting maximizing the expected cross entropy between the current model posterior $p(\mathcal{M} \mid \mathcal{D})$ and the updated distribution $p(\mathcal{M} \mid \mathcal{D}')$. This approach also requires extensive model retraining, with $\mathcal{O}\big(M|\mathbf{X}^*|\big)$ model updates per iteration, to estimate this expectation for each candidate. These approaches become prohibitively expensive for real-time applications with large number of candidates. In our audiometric experiments, for example, we consider 10 000 candidate points, expending 1–2 seconds per iteration, whereas these mentioned techniques would take several hours to selected the next point to query.

After the development of this work, it has been brought to our attention the work of Cavagnaro et al. [10], which is closely related ours. In that paper, the authors have also proposed to use mutual information to *model discrimination*, which, in essence, is the same active model selection problem discussed here. The main difference is how we compute mutual information. In [10], the authors compute the more-costly expression (4.1), which requires model retraining. Our approach avoids sampling y^* by using (4.2), which does not scale with $\mathcal{O}\big(|\mathbf{X}^*|\big)$.

Bayesian active model selection is close in spirit to the classical parlor game "Twenty Questions". One player chooses an instance from a general category, *e.g.* famous people, and the other players try to guess the particular instance by asking questions. There are many variations of this problem, and some of them were formalized and investigated [35, 19, 20, 52]. In

particular, the following papers have proposed information-theoretical approaches that are related to our approach, but different in the problem setting: [26, 71, 38, 84].

4.3 Active model selection for Gaussian processes

In the previous section, we proposed a general framework for performing sequential active Bayesian model selection, without making any assumptions about the forms of the models $\{\mathcal{M}_i\}$. Here we will discuss specific details of our proposal when these models represent alternative structures for Gaussian process priors on a latent function.

To connect to our active learning formulation, let $\{\mathcal{M}_i\}$ be a set of Gaussian process models for the latent function f. Each model comprises a mean function μ_i, covariance function K_i, and associated hyperparameters θ_i. Our goal is to automatically and actively distinguish the GP models through intelligent sampling BAMS (4.2). Below we give explicit formulas for computing the required entropies for common likelihoods: regression with Gaussian noise and probit regression.

Regression with Gaussian noise

For regression problems with zero-mean homoskedastic Gaussian noise with variance σ_n^2, we have

$$p(y \mid f) = \mathcal{N}(y; f, \sigma_n^2).$$

We may integrate this against a Gaussian distribution on the latent value f (such as that resulting from the MGP approximation (6.12)) to find the predictive distribution. Suppose

$p(f \mid \mathbf{x}, \mathcal{D}) = \mathcal{N}(f; \mu, \sigma^2)$. Then

$$p(y \mid \mathbf{x}, \mathcal{D}) = \mathcal{N}(y; \mu, \sigma^2 + \sigma_n^2).$$

The model-conditional predictive distributions $\{p(y^* \mid \mathbf{X}^*, \mathcal{D}, \mathcal{M}_i)\}$ are thus Gaussian under the MGP approximation:

$$p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}_i) = \mathcal{N}\big(y^*; \mu_i^*, (\nu_{\text{MGP}}^*)_i + \sigma_n^2\big).$$

The differential entropy of a one-dimensional Gaussian is

$$H\big[\mathcal{N}(y; \mu, \sigma^2)\big] = \tfrac{1}{2}\log(2\pi e\sigma^2);$$

with these results and an approximation to the model posterior, we may compute the second term in (4.2). The marginal predictive distribution $p(y^* \mid \mathbf{x}^*, \mathcal{D})$ is now a mixture of Gaussians weighted by the approximate model posterior. The differential entropy of a mixture of Gaussians does not have a closed form, so for the first term in (4.2), we must resort to (one-dimensional) quadrature.

Note that we may also treat the noise variance σ_n^2 as a model-dependent hyperparameter and use the MGP to approximately marginalize it as well, which changes the above only slightly. An extension to heteroskedastic noise is also trivial.

Probit regression

For binary classification problems with a probit likelihood, we have

$$\Pr(y = 1 \mid f) = \Phi(f),$$

where Φ is the univariate standard normal CDF. We may integrate this against a Gaussian distribution on the latent value f to find the predictive distribution. Suppose $p(f \mid \mathbf{x}, \mathcal{D}) = \mathcal{N}(f; \mu, \sigma^2)$. Then

$$\Pr(y = 1 \mid \mathbf{x}, \mathcal{D}) = \int \Phi(f)\mathcal{N}(f; \mu, \sigma^2)\,\mathrm{d}f = \Phi\left(\frac{\mu}{\sqrt{1+\sigma^2}}\right).$$

The model-conditional predictive distributions $\left\{p(y^* \mid \mathbf{X}^*, \mathcal{D}, \mathcal{M}_i)\right\}$ are thus Bernoulli distributions under the MGP approximation:

$$p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}_i) = \mathcal{B}\left(\Phi\left(\frac{\mu_i^*}{\sqrt{1 + (\nu_{\mathrm{MGP}}^*)_i}}\right)\right).$$

The differential entropy of a Bernoulli distribution with success probability p is given by the Bernoulli entropy function h:

$$H\big[\mathcal{B}(p)\big] = h(p) = -p\log p - (1-p)\log(1-p);$$

with these results and an approximation to the model posterior, we may compute the second term in (4.2). The marginal predictive distribution $p(y^* \mid \mathbf{x}^*, \mathcal{D})$ is now a mixture of Bernoullis weighted by the approximate model posterior. Bernoulli distributions are closed under taking mixtures, and therefore in this case we may compute both terms in (4.2) without resorting to quadrature.

dataset	expert model	CKS model
Airline passengers	**LIN+(PER×RQ)**	SE×(LIN+LIN×(PER+RQ))
Mauna Loa atmospheric CO_2	SE+(PER×SE)+RQ+SE	**LIN×SE+SE×(PER+RQ)**

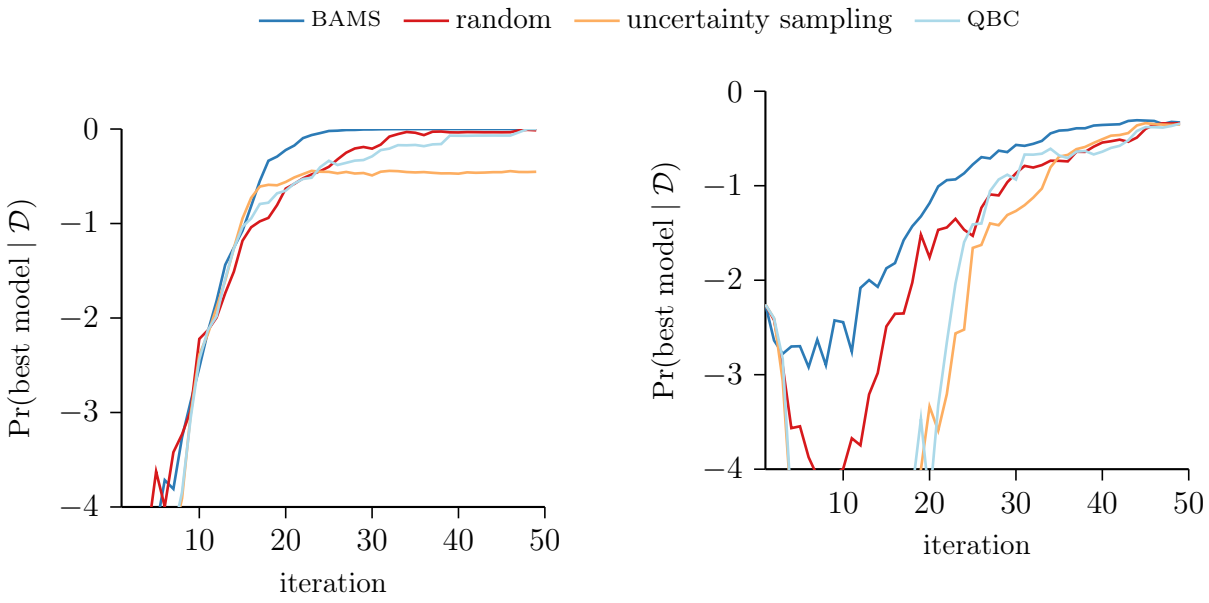
Table 4.1: Kernels used as experts suggestion and selected by the method of [13] for each time series. The kernel that best explains the data using all points is shown in bold. The expert kernel for the airline data was our guess after plotting the data, whereas for Mauna Loa is the one presented in [69].

Implementation

We may now efficiently compute an approximation to the BAMS criterion for active GP model selection. Given currently observed data $\mathcal{D}$, for each of our candidate models $\mathcal{M}_i$, we first find the Laplace approximation to the hyperparameter posterior (2.4) and model evidence (5.1). Given the approximations to the model evidence, we may compute an approximation to the model posterior (6.4). Suppose we have a set of candidate points $\mathbf{X}^*$ from which we may select our next point. For each of our models, we compute the MGP approximation (6.12) to the latent posteriors $\{p(\mathbf{f}^* \mid \mathbf{X}^*, \mathcal{D}, \mathcal{M}_i)\}$, from which we use standard techniques to compute the predictive distributions $\{p(\mathbf{y}^* \mid \mathbf{X}^*, \mathcal{D}, \mathcal{M}_i)\}$. Finally, with the ability to compute the differential entropies of these model-conditional predictive distributions, as well as the marginal predictive distribution (4.3), we may compute the mutual information of each candidate in parallel.

4.4 Active structure discovery demonstration

We evaluated the proposed method using the time series shown in Table 1, also used by Duvenaud et al. [13]. For each dataset, we considered 18 models: four base kernels (squared exponential (SE), periodic (PER), linear (LIN), and rational quadratic (RQ)), 12 kernels



(a) Airline data set

(b) Mauna Loa atmospheric CO_2

Figure 4.1: Posterior probability of best model as function of iteration number.

obtained by combining these basis kernels with sum and products, an expert model designing by a human (see Table 1) and the kernel found by the compositional kernel search (CKS) [13] method. Using all points, we computed the model posterior to determine which model best describes the data, shown in Table 1 (bold).

Then, we randomly selected two points and applied several techniques to choose a total of 50 points: BAMS, random sampling, uncertainty sampling, and query by committee (QBC)[8] [75]. We repeated this experiment 30 times, using different initial random seeds for each. Figure 2 shows the average posterior probability of the best model as function of iteration number, along with the standard error. The results show that BAMS outperforms all the baselines, more quickly finding the best model.

[8]We adapted QBC using the entropy of the model-marginal predictive distribution (4.3) as the disagreement criterion. Furthermore, to consider every committee member's (i.e. each model's) predictions equally important, we compute this quantity assuming a uniform model posterior.

4.5 Audiometric threshold testing

Standard audiometric tests [9, 36, 11] are calibrated such that the average human subject has a 50% chance of hearing a tone at any frequency; this empirical unit of intensity is defined as 0 dB HL. Humans give binary reports (whether or not a tone was heard) in response to stimuli, and these observations are inherently noisy. Typical audiometric tests present tones in a predefined order on a grid, in increments of 5–10 dB HL at each of six octaves. Recently, Gardner et al. [17] demonstrated that Bayesian active learning of a patient's audiometric function significantly improves the state-of-the-art in terms of accuracy and number of stimuli required.

However, learning a patient's entire audiometric function may not always be necessary. Audiometric testing is frequently performed on otherwise young and healthy patients to detect *noise-induced hearing loss* (NIHL). Noise-induced hearing loss occurs when an otherwise healthy individual is habitually subjected to high-intensity sound [59]. This can result in sharp, notch-shaped hearing loss in a narrow (sometimes less than one octave) frequency range. Early detection of NIHL is critical to desirable long-term clinical outcomes, so large-scale screenings of susceptible populations (for example, factory workers), is commonplace [64]. Noise-induced hearing loss is difficult to diagnose with standard audiometry, because a frequency–intensity grid must be very fine to ensure that a notch is detected. The full audiometric test of Gardner et al. [17] may also be inefficient if the only goal of testing is to determine whether a notch is present, as would be the case for large-scale screening.

We cast the detection of noise-induced hearing loss as an active model selection problem. We will describe two Gaussian process models of audiometric functions: a baseline model of normal human hearing, and a model reflecting NIHL. We then use the BAMS framework

introduced above to, as rapidly as possible for a given patient, determine which model best describes his or her hearing.

Normal-patient model

To model a healthy patient's audiometric function, we use the model described in [17]. The GP prior proposed in that work combines a constant prior mean $\mu_{\text{healthy}} = c$ (modeling a frequency-independent natural threshold) with a kernel taken to be the sum of two components: a linear covariance in intensity and a squared-exponential covariance in frequency. Let $[i, \phi]$ represent a tone stimulus, with i representing its intensity and ϕ its frequency. We define:

$$K\big([i,\phi],[i',\phi']\big) = \alpha i i' + \beta \exp\big(-\tfrac{1}{2\ell^2}|\phi - \phi'|^2\big), \tag{4.5}$$

where $\alpha, \beta > 0$ weight each component and $\ell > 0$ is a length scale of frequency-dependent random deviations from a constant hearing threshold. This kernel encodes two fundamental properties of human audiologic response. First, hearing is monotonic in intensity. The linear contribution $\alpha i i'$ ensures that the posterior probability of detecting a fixed frequency will be monotonically increasing after conditioning on a few tones. Second, human hearing ability is locally smooth in frequency, because nearby locations in the cochlea are mechanically coupled. The combination of μ_{healthy} with K specifies our healthy model $\mathcal{M}_{\text{healthy}}$, with parameters $\theta_{\text{healthy}} = [c, \alpha, \beta, \ell]^\top$.

Noise-induced hearing loss model

We extend the model above to create a second GP model reflecting a localized, notch-shaped hearing deficit characteristic of NIHL. We create a novel, flexible prior mean function for this purpose, the parameters of which specify the exact nature of the hearing loss. Our proposed notch mean is:

$$\mu_{\text{NIHL}}(i, \phi) = c - d\mathcal{N}'(\phi; \nu, w^2), \tag{4.6}$$

where $\mathcal{N}'(\phi; \nu, w)$ denotes the unnormalized normal probability density function with mean ν and standard deviation w, which we scale by a depth parameter $d > 0$ to reflect the prominence of the hearing loss. This contribution results in a localized subtractive notch feature with tunable center, width, and height. We retain a constant offset c to revert to the normal-hearing model outside the vicinity of the localized hearing deficit. Note that we completely model the effect of NIHL on patient responses with this mean notch function; the kernel K above remains appropriate. The combination of μ_{NIHL} with K specifies our NIHL model $\mathcal{M}_{\text{NIHL}}$ with, in addition to the parameters of our healthy model, the additional parameters $\theta_{\text{NIHL}} = [\nu, w, d]^\top$.

4.6 Diagnosing noise-induced hearing loss

To test BAMS on our NIHL detection task, we evaluate our algorithm using audiometric data, comparing to several baselines. From the results of a small-scale clinical trial, we have examples of high-fidelity audiometric functions inferred for several human patients using the method of Gardner et al. [17]. We may use these to simulate audiometric examinations of healthy patients using different methods to select tone presentations. We simulate patients

with NIHL by adjusting ground truth inferred from nine healthy patients with in-model samples from our notch mean prior. Recall that high-resolution audiogram data is extremely scarce.

We first took a thorough pure-tone audiometric test of each of nine patients from our trial with normal hearing using 100 samples selected using the algorithm in [17] on the domain $\mathcal{X} = [250, 8000]\,\text{Hz} \times [-10, 80]\,\text{dB HL}$,[9] typical ranges for audiometric testing [36]. We inferred the audiometric function over the entire domain from the measured responses, using the healthy-patient GP model $\mathcal{M}_{\text{healthy}}$ with parameters learned via MLE–II inference. The observation model was $p(y = 1 \mid f) = \Phi(f)$, where Φ is the standard normal CDF, and approximate GP inference was performed via a Laplace approximation. We then used the approximate GP posterior $p(f \mid \mathcal{D}, \hat{\theta}, \mathcal{M}_{\text{healthy}})$ for this patient as ground-truth for simulating a healthy patient's responses. The posterior probability of tone detection learned from one patient is shown in the background of Figure 6.1(a). We simulated a healthy patient's response to a given query tone $\mathbf{x}^* = [i^*, \phi^*]$ by sampling a conditionally independent Bernoulli random variable with parameter $p(y^* = 1 \mid \mathbf{x}^*, \mathcal{D}, \hat{\theta}, \mathcal{M}_{\text{healthy}})$.

We simulated a patient with NIHL by then drawing notch parameters (the parameters of (4.6)) from an expert-informed prior, adding the corresponding notch to the learned healthy ground-truth latent mean, recomputing the detection probabilities, and proceeding as above. Example NIHL ground-truth detection probabilities generated in this manner are depicted in the background of Figure 6.1(b).

[9] Inference was done in log-frequency domain.

Experimental results

To test our active model-selection approach to diagnosing NIHL, we simulated a series of audiometric tests, selecting tones using three alternatives: BAMS, the algorithm of [17], and random sampling.[10] Each algorithm shared a candidate set of 10 000 quasi-random tones $\mathbf{X}^*$ generated using a scrambled Halton set so as to densely cover the two-dimensional search space. We simulated nine healthy patients and a total of 27 patients exhibiting a range of NIHL presentations, using independent draws from our notch mean prior in the latter case. For each audiometric test simulation, we initialized with five random tones, then allowed each algorithm to actively select a maximum of 25 additional tones, a very small fraction of the hundreds typically used in a regular audiometric test. We repeated this procedure for each of our nine healthy patients using the normal-patient ground-truth model. We further simulated, for each patient, three separate presentations of NIHL as described above. We plot the posterior probability of the correct model after each iteration in Figure 4.3.

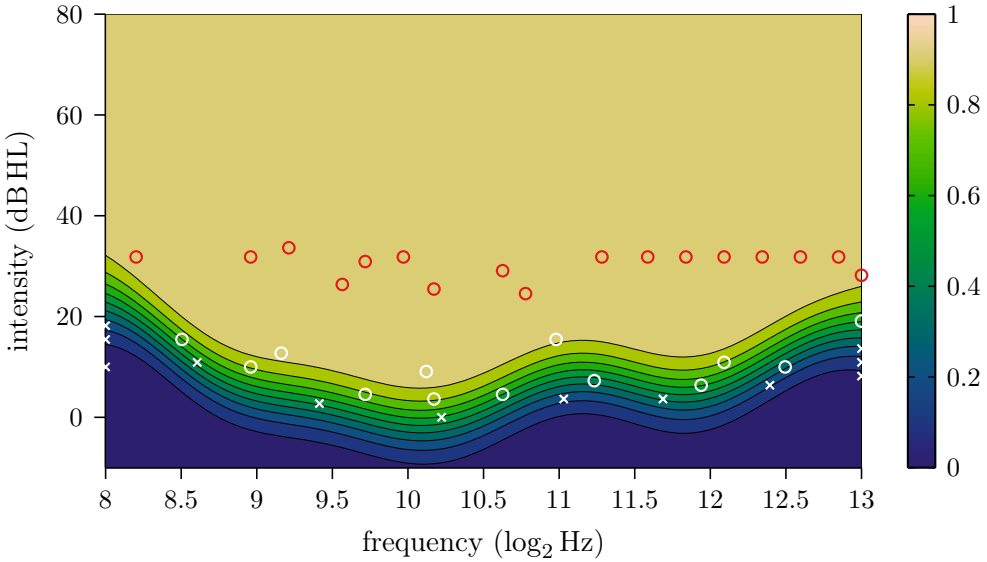
In all runs with both ground-truth models, BAMS was able to rapidly achieve greater than 99% confidence in the correct model without expending the entire budget. Although all methods correctly inferred high healthy posterior probability for the healthy patient, BAMS wass more confident. For the NIHL patients, neither baseline inferred the correct model, whereas BAMS rarely required more than 15 actively chosen samples to confidently make the correct diagnosis. Note that, when BAMS was used on NIHL patients, there was often an initial period during which the healthy model was favored, followed by a rapid shift towards the correct model. This is because our method penalizes the increased complexity of the notch model until sufficient evidence for a notch is acquired.

[10] We also compared with uncertainty sampling and query by committee (QBC); the performance was comparable to random sampling and is omitted for clarity.

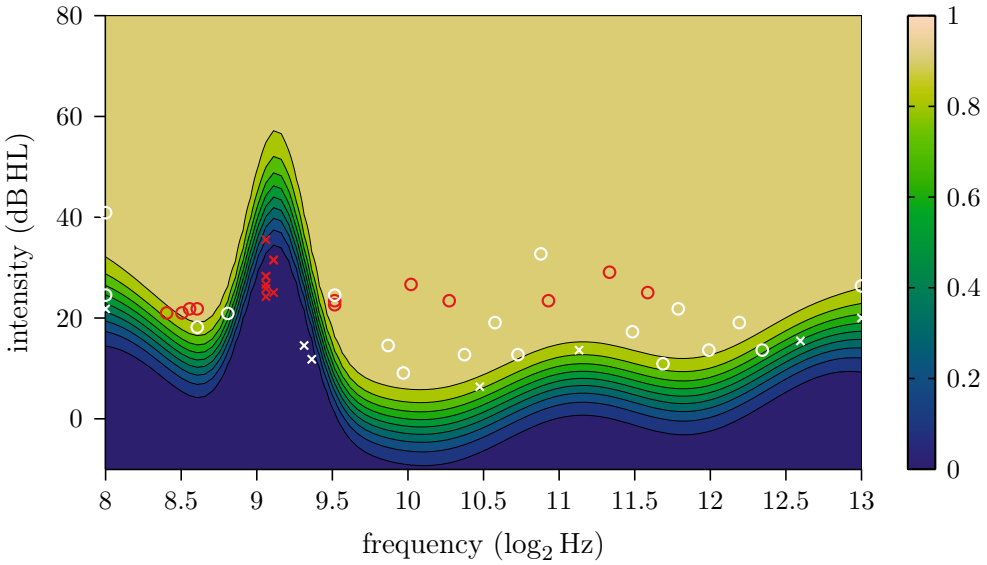
Figure 6.1 shows the samples selected by BAMS for typical healthy and NIHL patients. The fundamental strategy employed by BAMS in this application is logical: it samples in a row of relatively high-intensity tones. The intuition for this design is that failure to recognize a normally heard, high-intensity sound is strong evidence of a notch deficit. Once the notch has been found (Figure 6.1(b)), BAMS continues to sample within the notch to confirm its existence and rule out the possibility of the miss (tone not heard) being due to the stochasticity of the process. Once satisfied, the BAMS approach then samples on the periphery of the notch to further solidify its belief.

The BAMS algorithm sequentially makes observations where the healthy and NIHL model disagree the most, typically in the top-center of the MAP notch location. The exact intensity at which BAMS samples is determined by the prior over the notch-depth parameter d. When we changed the notch depth prior to support shallower or deeper notches (data not shown), BAMS sampled at lower or higher intensities, respectively, to continue to maximize model disagreement. Similarly, the spacing between samples is controlled by the prior over the notch-width parameter w.

Finally, it is worth emphasizing the stark difference between the sampling pattern of BAMS and the audiometric tests of [17]; see Figure 6.1. Indeed, when the goal is learning the patient's audiometric function, the audiometric testing algorithm proposed in that work typically has a very good estimate after 20 samples. However, when using BAMS, the primary goal is to detect or rule out NIHL. As a result, the samples selected by BAMS reveal little about the nuances of the patient's audiometric function, while being highly informative about the correct model to explain the data. This is precisely the tradeoff one seeks in a large-scale diagnostic setting, highlighting the critical importance of focusing on the model-selection problem directly.

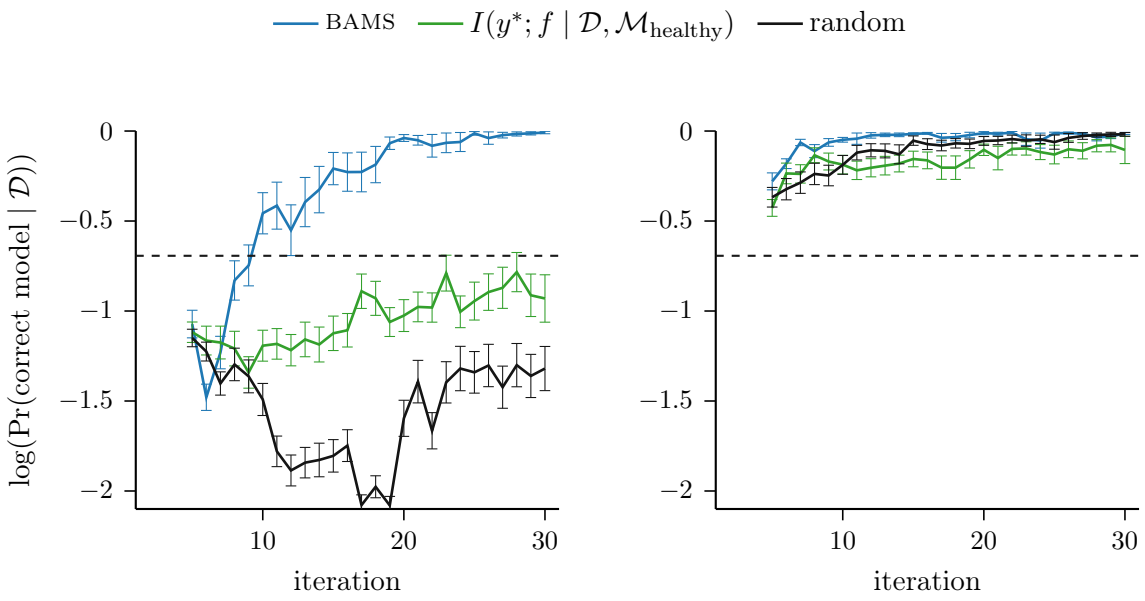


(a) Normal hearing model ground truth.



(b) Notch model ground truth.

Figure 4.2: Samples selected by BAMS (red) and the method of Gardner et al. [17] (white) when run on (a) the normal-hearing ground truth, and (b), the NIHL model ground truth. Contours denote probability of detection at 10% intervals. Circles indicate presentations that were heard by the simulated patient; exes indicate presentations that were not heard by the simulated patient.



(a) Notch model ground truth.

(b) Normal hearing model ground truth.

Figure 4.3: Posterior probability of the correct model as a function of iteration number.

4.7 Conclusion

We introduced a novel information-theoretic approach for active model selection, *Bayesian active model selection*, and successfully applied it to rapid screening for noise-induced hearing loss. Our method for active model selection does not require model retraining to evaluate candidate points, making it more feasible than previous approaches. Further, we provided an effective and efficient analytic approximation to our criterion that can be used for automatically learning the model class of Gaussian processes with arbitrary observation likelihoods, a rich and commonly used class of potential models.

Chapter 5

Automated model selection

Over the past decades, enormous human effort has been devoted to machine learning; preprocessing data, model selection, and hyperparameter optimization are some examples of critical and often expert-dependent tasks. The complexity of these tasks has in some cases relegated them to the realm of "dark art." In kernel methods in particular, the selection of an appropriate kernel to explain a given dataset is critical to success in terms of the fidelity of predictions, but the vast space of potential kernels renders the problem nontrivial. We consider the problem of automatically finding an appropriate probabilistic model to explain a given dataset. Although our proposed algorithm is general, we will focus on the case where a model can be completely specified by a kernel, as is the case for example for centered Gaussian processes (GPs).

Recent work has begun to tackle the kernel-selection problem in a systematic way. Duvenaud et al. [13] and Grosse et al. [28] described generative grammars for enumerating a countably infinite space of arbitrarily complex kernels via exploiting the closure of kernels under additive and multiplicative composition. We adopt this kernel grammar in this work as well. Given a dataset, Duvenaud et al. [13] proposed searching this infinite space of models using a greedy search mechanism. Beginning at the root of the grammar, we traverse the tree

greedily attempting to maximize the (approximate) evidence for the data given by a GP model incorporating the kernel.

Here, we develop a more sophisticated mechanism for searching through this space. The greedy search described above only considers a given dataset by querying a model's evidence. Our search performs a *metalearning* procedure, which, conditional on a dataset, establishes similarities among the models in terms of the space of explanations they can offer for the data. With this viewpoint, we construct a novel kernel between models (a "kernel kernel"). We then approach the model-search problem via Bayesian optimization, treating the model evidence as an expensive black-box function to be optimized as a function of the kernel. The dependence of our kernel between models on the distribution of the data is critical; depending on a given dataset, the kernels generated by a compositional grammar could be especially rich or deceptively so.

We develop an automatic framework for exploring a set of potential models, seeking the model that best explains a given dataset. Although we focus on Gaussian process models defined by a grammar, our method could be easily extended to any probabilistic model with a parametric or structured model space. Our search appears to perform competitively with other baselines across a variety of datasets, including the greedy method from [13], especially in terms of the number of models for which we must compute the (expensive) evidence, which typically scales cubically for kernel methods.

5.1 Bayesian optimization for model search

Suppose we face a classical supervised learning problem defined on an input space $\mathcal{X}$ and output space $\mathcal{Y}$. We are given a set of training observations $\mathcal{D} = (\mathbf{X}, \mathbf{y})$, where $\mathbf{X}$ represents the design matrix of explanatory variables $\mathbf{x}_i \in \mathcal{X}$, and $y_i \in \mathcal{Y}$ is the respective value or label to be predicted. Ultimately, we want to use $\mathcal{D}$ to predict the value y_* associated with an unseen point $\mathbf{x}_*$. Given a probabilistic model $\mathcal{M}$, we may accomplish this via formation of the predictive distribution.

Suppose, however, that we are given a collection of probabilistic models $\mathbb{M}$ that could have plausibly generated the data. Ideally, finding the source of $\mathcal{D}$ would let us solve our prediction task with the highest fidelity. Let $\mathcal{M} \in \mathbb{M}$ be a probabilistic model, and let $\Theta_{\mathcal{M}}$ be the corresponding parameter space. These models are typically parametric families of distributions, each of which encodes a *structural* assumption about the data, for example, that the data can be described by a linear, quadratic, or periodic trend. Further, the member distributions ($\mathcal{M}_\theta \in \mathcal{M}$, $\theta \in \Theta_{\mathcal{M}}$) of $\mathcal{M}$ differ from each other by a particular value of some properties—represented by the *hyperprameters* θ—related to the data such as amplitude, characteristic length scales, etc.

We wish to select one model from this collection of models $\mathbb{M}$ to explain $\mathcal{D}$. From a Bayesian perspective, the principle approach for solving this problem is *Bayesian model selection.*[11] The critical value is *model evidence*, the probability of generating the observed data given a model $\mathcal{M}$:

$$p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}) = \int_{\Theta_{\mathcal{M}}} p(\mathbf{y} \mid \mathbf{X}, \theta, \mathcal{M})\, p(\theta \mid \mathcal{M})\, \mathrm{d}\theta. \tag{5.1}$$

[11] "Model selection" is unfortunately sometimes also used in GP literature for the process of hyperparameter learning (selecting some $\mathcal{M}_\theta \in \mathcal{M}$), rather than selecting a model class $\mathcal{M}$, the focus of our work.

The evidence (also called *marginal likelihood*) integrates over θ to account for all possible explanations of the data offered by the model, under a prior $p(\theta \mid \mathcal{M})$ associated with that model.

Our goal is to automatically explore a space of models $\mathbb{M}$ to select a model[12] $\mathcal{M}^* \in \mathbb{M}$ that explains a given dataset $\mathcal{D}$ as well as possible, according to the model evidence. The essence of our method, which we call Bayesian optimization for automated model selection (BOMS), is viewing the evidence as a function $g\colon \mathbb{M} \to \mathbb{R}$ to be optimized. We note two important aspects of g. First, for large datasets and/or complex models, g is an expensive function, for example growing cubically with $|\mathcal{D}|$ for GP models. Further, gradient information about g is impossible to compute due to the discrete nature of $\mathbb{M}$. We can, however, query a model's evidence as a black-box function. For these reasons, we propose to optimize evidence over $\mathbb{M}$ using *Bayesian optimization,* a technique well-suited for optimizing expensive, gradient-free, black-box objectives [76, 62, 8, 78, 21].

In this framework, we seek an optimal model

$$\mathcal{M}^* = \arg\max_{\mathcal{M} \in \mathbb{M}} g(\mathcal{M}; \mathcal{D}), \tag{5.2}$$

where $g(\mathcal{M}; \mathcal{D})$ is the (log) model evidence: $g(\mathcal{M}; \mathcal{D}) = \log p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})$. We begin by placing a Gaussian process (GP) prior on g, $p(g) = \mathcal{GP}(g; \mu_g, K_g)$, where $\mu_g\colon \mathbb{M} \to \mathbb{R}$ is a mean function and $K_g\colon \mathbb{M}^2 \to \mathbb{R}$ is a covariance function appropriately defined over the model space $\mathbb{M}$. This is a nontrivial task due to the discrete and potentially complex nature of $\mathbb{M}$. We will suggest useful choices for μ_g and K_g when $\mathbb{M}$ is a space of Gaussian process models

[12] We could also select a *set* of models but, for simplicity, we assume that there is one model that best explains that data with overwhelming probability, which would imply that there is not benefit in considering more than one model, e.g., via Bayesian model averaging.

below. Now, given observations of the evidence of a selected set of models,

$$\mathcal{D}_g = \big\{\big(\mathcal{M}_i, g(\mathcal{M}_i; \mathcal{D})\big)\big\}, \tag{5.3}$$

we may compute the posterior distribution on g conditioned on $\mathcal{D}_g$, which will be an updated Gaussian process [89, 69]. Bayesian optimization uses this probabilistic belief about g to induce an inexpensive acquisition function to select which model we should select to evaluate next. Here we use the classical *expected improvement* (EI) [76, 40] acquisition function, or a slight variation described below, because it naturally considers the trade off between exploration and exploitation. The exact choice of acquisition function, however, is not critical to our proposal. In each round of our model search, we will evaluate the acquisition function in the optimal model evidence for a number of candidate models $\mathcal{C}(\mathcal{D}_g) = \{\mathcal{M}_i\}$, and compute the evidence of the candidate where this is maximized:

$$\mathcal{M}' = \arg\max_{\mathcal{M} \in \mathcal{C}} \alpha_{\text{EI}}(\mathcal{M}; \mathcal{D}_g).$$

We then incorporate the chosen model $\mathcal{M}'$ and the observed model evidence $g(\mathcal{M}'; \mathcal{D})$ into our model evidence training set $\mathcal{D}_g$, update the posterior on g, select a new set of candidates, and continue. We repeat this iterative procedure until a budget is expended, typically measured in terms of the number of models considered.

We have observed that expected improvement works well especially for small and/or low-dimensional problems. When the dataset is large and/or high-dimensional, training costs can be considerable and variable, especially for complex models. To give better anytime performance on such datasets, we use *expected improvement per second,* where we divide the expected improvement by an estimate of the time required to compute the evidence. In our

experiments, this estimation was performed by fitting a linear regression model to the log time to compute $g(\mathcal{M}; \mathcal{D})$ as a function of the number of hyperparameters (the dimension of $\Theta_{\mathcal{M}}$) that we train on the models available in $\mathcal{D}_g$.

The acquisition function allows us to quickly determine which models are more promising than others, given the evidence we have observed so far. Since $\mathbb{M}$ is an infinite set of models, we cannot consider every model in every round. Instead, we will define a heuristic to evaluate the acquisition function at a smaller set of active candidate models below.

5.2 Bayesian optimization for GP kernel search

We introduced above a general framework for searching over a space of probabilistic models $\mathbb{M}$ to explain a dataset $\mathcal{D}$ without making further assumptions about the nature of the models. In the following, we will provide specific suggestions in the case that all members of $\mathbb{M}$ are Gaussian process priors on a latent function.

We assume that our observations $\mathbf{y}$ were generated according to an unknown function $f\colon \mathcal{X} \to \mathbb{R}$ via a fixed probabilistic observation mechanism $p(\mathbf{y} \mid \mathbf{f})$, where $f_i = f(\mathbf{x}_i)$. In our experiments, we will consider regression with additive Gaussian observation noise, but this is not integral to our approach. We further assume a GP prior distribution on f, $p(f) = \mathcal{GP}(f; \mu_f, K_f)$, where $\mu_f\colon \mathcal{X} \to \mathbb{R}$ is a mean function and $K_f\colon \mathcal{X}^2 \to \mathbb{R}$ is a positive-definite covariance function or kernel. For simplicity, we will assume that the prior on f is centered, $\mu_f(x) = 0$, which lets us fully define the prior on f by the kernel function K_f. We assume that the kernel function is parameterized by hyperparameters that we concatenate into a vector θ.

In this restricted context, a model $\mathcal{M}$ is completely determined by the choice of kernel function and an associated hyperparameter prior $p(\theta \mid \mathcal{M})$. Below we briefly review a previously suggested method for constructing an infinite space of potential kernels to model the latent function f, and thus an infinite family of models $\mathbb{M}$.

Space of compositional GP kernels

We adopt the same space of kernels defined by Duvenaud et al. [13], which we briefly summarize here. We refer the reader to the original paper for more details. Given a set of simple, so-called *base kernels,* such as the common squared exponential (SE), periodic (PER), linear (LIN), and rational quadratic (RQ) kernels, we create new and potentially complex kernels by summation and multiplication of these base units. The entire kernel space can be describe by the following grammar rules:

1. Any subexpression $\mathcal{S}$ can be replaced with $\mathcal{S} + \mathcal{B}$, where $\mathcal{B}$ is a base kernel.
2. Any subexpression $\mathcal{S}$ can be replaced with $\mathcal{S} \times \mathcal{B}$, where $\mathcal{B}$ is a base kernel.
3. Any base kernel $\mathcal{B}$ may be replaced with another base kernel $\mathcal{B}'$.

Hyperparameter priors

The base kernels we will use are well understood, as are their hyperparameters, which have simple interpretations that can be thematically grouped together. We take advantage of the Bayesian framework to encode prior knowledge over hyperparameters, i.e., $p(\theta \mid \mathcal{M})$. Conveniently, these priors can also potentially mitigate numerical problems during the training

of the GPs. Here we derive a consistent method to construct such priors for arbitrary kernels and datasets in regression problems.

We first standardize the dataset, i.e., we subtract the mean and divide by the standard deviation of both the predictive features $\{x_i\}$ and the outputs $\mathbf{y}$. This gives each dataset a consistent scale. Now we can reason about what real-world datasets usually look like in this scale. For example, we do not typically expect to see datasets spanning 10 000 length scales. Here we encode what we judge to be reasonable priors for groups of thematically related hyperparameters for most datasets. These include three types of hyperparameters common to virtually any problem: length scales ℓ (including, for example, the period parameter of a periodic covariance), signal variance σ, and observation noise σ_n. We also consider separately three other parameters specific to particular covariances we use here: the α parameter of the rational quadratic covariance [69, (4.19)], the "length scale" of the periodic covariance ℓ_p [69, ℓ in (4.31)], and the offset σ_0 in the linear covariance. We define the following:

$$p(\log \ell) = \mathcal{N}(0.1, 0.7^2) \qquad p(\log \sigma) = \mathcal{N}(0.4, 0.7^2) \qquad p(\log \sigma_n) = \mathcal{N}(0.1, 1^2)$$
$$p(\log \alpha) = \mathcal{N}(0.05, 0.7^2) \qquad p(\log \ell_p) = \mathcal{N}(2, 0.7^2) \qquad p(\sigma_0) = \mathcal{N}(0, 2^2)$$

Given these, each model was given an independent prior over each of its hyperparameters, using the appropriate selection from the above for each.

Creating a "kernel kernel"

In §5.2, we focused on modeling a latent function f with a GP, creating an infinite space of models $\mathbb{M}$ to explain f (along with associated hyperparameter priors), and approximating the log model evidence function $g(\mathcal{M}; \mathcal{D})$. The evidence function g is the objective function

we are trying to optimize via Bayesian optimization. We described in §5.1 how this search progresses in the general case, described in terms of an arbitrary Gaussian process prior on g. Here we will provide specific suggestions for the modeling of g in the case that the model family $\mathbb{M}$ comprises Gaussian process priors on a latent function f, as discussed here and considered in our experiments.

Our prior belief about g is given by a GP prior $p(g) = \mathcal{GP}(g; \mu_g, K_g)$, which is fully specified by the mean μ_g and covariance functions K_g. We define the former as a simple constant mean function $\mu_g(\mathcal{M}) = \theta_\mu$, where θ_μ is a hyperparameter to be learned through a regular GP training procedure given a set of observations. The latter we will construct as follows.

The basic idea in our construction is that is that we will consider the distribution of the observation locations in our dataset $\mathcal{D}$, $\mathbf{X}$ (the design matrix of the underlying problem). We note that selecting a model class $\mathcal{M}$ induces a prior distribution over the latent function values at $\mathbf{X}$, $p(\mathbf{f} \mid \mathbf{X}, \mathcal{M})$:

$$p(\mathbf{f} \mid \mathbf{X}, \mathcal{M}) = \int p(\mathbf{f} \mid \mathbf{X}, \mathcal{M}, \theta)\, p(\theta \mid \mathcal{M})\, \mathrm{d}\theta.$$

This prior distribution is an infinite mixture of multivariate Gaussian prior distributions, each conditioned on specific hyperparameters θ. We consider these prior distributions as different explanations of the latent function f, restricted to the observed locations, offered by the model $\mathcal{M}$. We will compare two models in $\mathbb{M}$ according to how different the explanations they offer for $\mathbf{f}$ are, *a priori.*

The *Hellinger distance* is a probability metric that we adopt as a basic measure of similarity between two distributions. Although this quantity is defined between arbitrary probability distributions (and thus could be used with non-GP model spaces), we focus on the multivariate

normal case. Suppose that $\mathcal{M}, \mathcal{M}' \in \mathbb{M}$ are two models that we wish to compare, in the context of explaining a fixed dataset $\mathcal{D}$. For now, suppose that we have conditioned each of these models on arbitrary hyperparameters (that is, we select a particular prior for f from each of these two families), giving $\mathcal{M}_\theta$ and $\mathcal{M}'_{\theta'}$, with $\theta \in \Theta_{\mathcal{M}}$ and $\theta' \in \Theta_{\mathcal{M}'}$. Now, we define the two distributions

$$P = p(\mathbf{f} \mid \mathbf{X}, \mathcal{M}, \theta) = \mathcal{N}(\mathbf{f}; \mu_P, \Sigma_P) \qquad Q = p(\mathbf{f} \mid \mathbf{X}, \mathcal{M}', \theta') = \mathcal{N}(\mathbf{f}; \mu_Q, \Sigma_Q).$$

The squared *Hellinger distance* between P and Q is

$$d_{\text{H}}^2(P, Q) = 1 - \frac{|\Sigma_P|^{1/4}|\Sigma_Q|^{1/4}}{\left|\frac{\Sigma_P + \Sigma_Q}{2}\right|^{1/2}} \exp\left\{-\frac{1}{8}(\mu_P - \mu_Q)^\top \left(\frac{\Sigma_P + \Sigma_Q}{2}\right)^{-1} (\mu_P - \mu_Q)\right\}. \tag{5.4}$$

The Hellinger distance will be small when P and Q are highly overlapping, and thus $\mathcal{M}_\theta$ and $\mathcal{M}'_{\theta'}$ provide similar explanations *for this dataset.* The distance will be larger, conversely, when $\mathcal{M}_\theta$ and $\mathcal{M}'_{\theta'}$ provide divergent explanations. Critically, we note that this distance depends on the dataset under consideration in addition to the GP priors.

Observe that the distance above is not sufficient to compare the similarity of two models $\mathcal{M}$, $\mathcal{M}'$ due to the fixing of hyperparameters above. To properly account for the different hyperparameters of different models, and the priors associated with them, we define the *expected squared Hellinger distance* of two models $\mathcal{M}, \mathcal{M}' \in \mathbb{M}$ as

$$\bar{d}_{\text{H}}^2(\mathcal{M}, \mathcal{M}'; \mathbf{X}) = \mathbb{E}\left[d_{\text{H}}^2(\mathcal{M}_\theta, \mathcal{M}'_{\theta'})\right] = \iint d_{\text{H}}^2(\mathcal{M}_\theta, \mathcal{M}'_{\theta'}; \mathbf{X})\, p(\theta \mid \mathcal{M})\, p(\theta' \mid \mathcal{M}')\, \mathrm{d}\theta\, \mathrm{d}\theta', \tag{5.5}$$

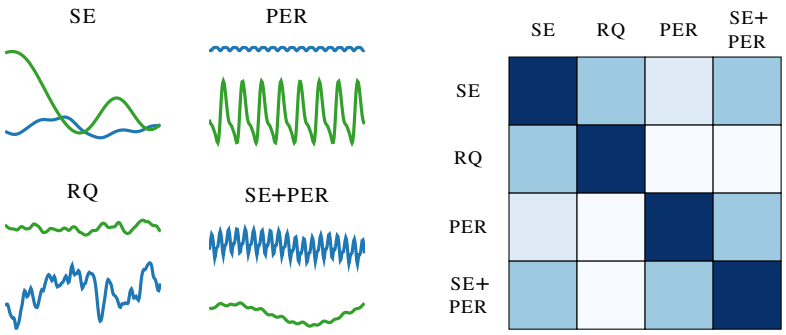


Figure 5.1: A demonstration of our model kernel K_g (5.6) based on expected Hellinger distance of induced latent priors. Left: four simple model classes on a $1d$ domain, showing samples from the prior $p(f \mid \mathcal{M}) \propto p(f \mid \theta, \mathcal{M})\, p(\theta \mid \mathcal{M})$. Right: our Hellinger squared exponential covariance evaluated for the grid domains on the left. Increasing intensity indicates stronger covariance. The sets $\{\text{SE}, \text{RQ}\}$ and $\{\text{SE}, \text{PER}, \text{SE}{+}\text{PER}\}$ show strong mutual correlation.

where the distance is understood to be evaluated between the priors provided on $\mathbf{f}$ induced at $\mathbf{X}$. Finally, we construct the *Hellinger squared exponential* covariance between models as

$$K_g(\mathcal{M}, \mathcal{M}'; \theta_g, \mathbf{X}) = \sigma^2 \exp\left(-\frac{1}{2}\frac{\bar{d}^2_{\text{H}}(\mathcal{M}, \mathcal{M}'; \mathbf{X})}{\ell^2}\right), \tag{5.6}$$

where $\theta_g = (\sigma, \ell)$ specifies output and length scale hyperparameters in this kernel/evidence space. This covariance is illustrated in Figure 5.1 for a few simple kernels on a fictitious domain.

We make two notes before continuing. The first observation is that computing (5.4) scales cubically with $|\mathbf{X}|$, so it might appear that we might as well compute the evidence instead. This is misleading for two reasons. First, the (approximate) computation of a given model's evidence via either a Laplace approximation or the BIC requires optimizing its hyperparameters. Especially for complex models this can require hundreds-to-thousands of computations that each require cubic time. Further, as a result of our investigations, we have concluded that in practice we may approximate (5.4) and (5.5) by considering only a small *subset* of the

observation locations $\mathbf{X}$ and that this usually sufficient to capture the similarity between models in terms of explaining a given dataset. In our experiments, we choose 20 points uniformly at random from those available in each dataset, fixed once for the entire procedure and for all kernels under consideration in the search. We then used these points to compute distances (5.4–5.6), significantly reducing the overall time to compute K_g.

Second, we note that the expectation in (5.5) is intractable. We approximate the expectation via quasi-Monte Carlo, using a low-discrepancy sequence (a Sobol sequence) of the appropriate dimension, and inverse transform sampling, to give consistent, representative samples from the hyperparameter space of each model.

Active set of candidate models

Another challenging of exploring an infinite set of models is how we advance the search. Each round, we only compute the acquisition function on a set of candidate models $\mathcal{C}$. Here we discuss our policy for creating and maintaining this set. From the kernel grammar (§5.2), we can define a model graph where two models are connected if we can apply one rule to produce the other. We seek to traverse this graph, balancing exploration (diversity) against exploitation (models likely to have higher evidence). We begin each round with a set of already chosen candidates $\mathcal{C}$. To encourage exploitation, we add to $\mathcal{C}$ all neighbors of the best model seen thus far. To encourage exploration, we perform random walks to create diverse models, which we also add to $\mathcal{C}$. We start each random walk from the empty kernel and repeatedly apply a random number of grammatical transformations. The number of such steps is sampled from a geometric distribution with termination probability $\frac{1}{3}$. We find that 15 random walks works well. To constrain the number of candidates, we discard the models with the lowest EI values at the end of each round, keeping $|\mathcal{C}|$ no larger than 600.

5.3 Experimental results

Here we evaluate our proposed algorithm. We split our evaluation into two parts: first, we show that our GP model for predicting a model's evidence is suitable; we then demonstrate that our model search method quickly finds a good model for a range of regression datasets. The datasets we consider are publicly available[13] and were used in previous related work [13, 4]. AIRLINE, MAUNA LOA, METHANE, and SOLAR are $1d$ time series, and CONCRETE and HOUSING have, respectively, 8 and 13 dimensions. To facilitate comparison of evidence across datasets, we report log evidence divided by dataset size, redefining

$$g(\mathcal{M}; \mathcal{D}) = \log(p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}))/|\mathcal{D}|. \tag{5.7}$$

We use the aforementioned base kernels $\{\text{SE}, \text{RQ}, \text{LIN}, \text{PER}\}$ when the dataset is one-dimensional. For multi-dimensional datasets, we consider the set $\{\text{SE}_i\} \cup \{\text{RQ}_i\}$, where the subscript indicates that the kernel is applied only to the ith dimension.

Predicting a model's evidence

We first demonstrate that our proposed regression model in model space (i.e., the GP on $g \colon \mathbb{M} \to \mathbb{R}$) is sound. We set up a simple prediction task where we predict model evidence on a set of models given training data. We construct a dataset $\mathcal{D}_g$ (5.3) of 1 000 models as follows. We initialize a set $\mathbb{M}$ with the set of base kernels, which varies for each dataset (see above). Then, we select one model uniformly at random from $\mathbb{M}$ and add its neighbors in the model grammar to $\mathbb{M}$. We repeat this procedure until $|\mathbb{M}| = 1\,000$ and computed $g(\mathcal{M}; \mathcal{D})$ for the entire set generated. We train several baselines on a subset of $\mathcal{D}_g$ and test their ability

[13] `https://archive.ics.uci.edu/ml/datasets.html`

Table 5.1: Root mean square error for model-evidence regression experiment.

Dataset	Train %	Mean	k-NN (SP)	k-NN ($\bar{d}_{\mathrm{H}}$)	GP ($\bar{d}_{\mathrm{H}}$)
CONCRETE	20	0.109 (0.000)	0.200 (0.020)	0.233 (0.008)	**0.107** (0.001)
	40	0.107 (0.000)	0.260 (0.025)	0.221 (0.007)	**0.102** (0.001)
	60	0.107 (0.000)	0.266 (0.007)	0.215 (0.005)	**0.097** (0.001)
	80	0.106 (0.000)	0.339 (0.015)	0.200 (0.003)	**0.093** (0.002)
HOUSING	20	0.210 (0.001)	0.226 (0.002)	0.347 (0.004)	**0.175** (0.002)
	40	0.207 (0.001)	0.235 (0.004)	0.348 (0.004)	**0.140** (0.002)
	60	0.206 (0.000)	0.235 (0.004)	0.348 (0.004)	**0.123** (0.002)
	80	0.206 (0.000)	0.257 (0.004)	0.344 (0.004)	**0.114** (0.002)
MAUNA LOA	20	0.543 (0.002)	0.736 (0.051)	0.685 (0.010)	**0.513** (0.003)
	40	0.537 (0.001)	0.878 (0.062)	0.667 (0.005)	**0.499** (0.003)
	60	0.535 (0.001)	1.051 (0.058)	0.686 (0.010)	**0.487** (0.004)
	80	0.534 (0.001)	1.207 (0.048)	0.707 (0.005)	**0.474** (0.004)

to predict the evidence of the remaining models, as measured by the root mean squared error (RMSE). To achieve reliable results we repeat this experiment ten times. We considered a subset of the datasets (including both high-dimensional problems), because training 1 000 models demands considerable time. We compare with several alternatives:

1. **Mean prediction.** Predicts the mean evidence on the training models.

2. **k-nearest neighbors.** We perform k-NN regression with two distances: shortest-path distance in the directed model graph described in §5.2 (SP), and the expected squared Hellinger distance (5.5). Inverse distance was used as weights.

We select k for both k-NN algorithms through cross-validation, trying all values of k from 1 to 10. We show the average RMSE along with standard error in Table 5.1. The GP with our Hellinger distance model covariance universally achieves the lowest error. Both k-NN methods are outperformed by the simple mean prediction. We note that in these experiments, many models perform similarly in terms of evidence (usually, this is because many models are "bad"

in the same way, e.g., explaining the dataset away entirely as independent noise). We note, however, that the GP model is able to exploit correlations in *deviations* from the mean, for example in "good pockets" of model space, to achieve better performance. We also note that both the k-NN and GP models have decreasing error with the number of training models, suggesting our novel model distance is also useful in itself.

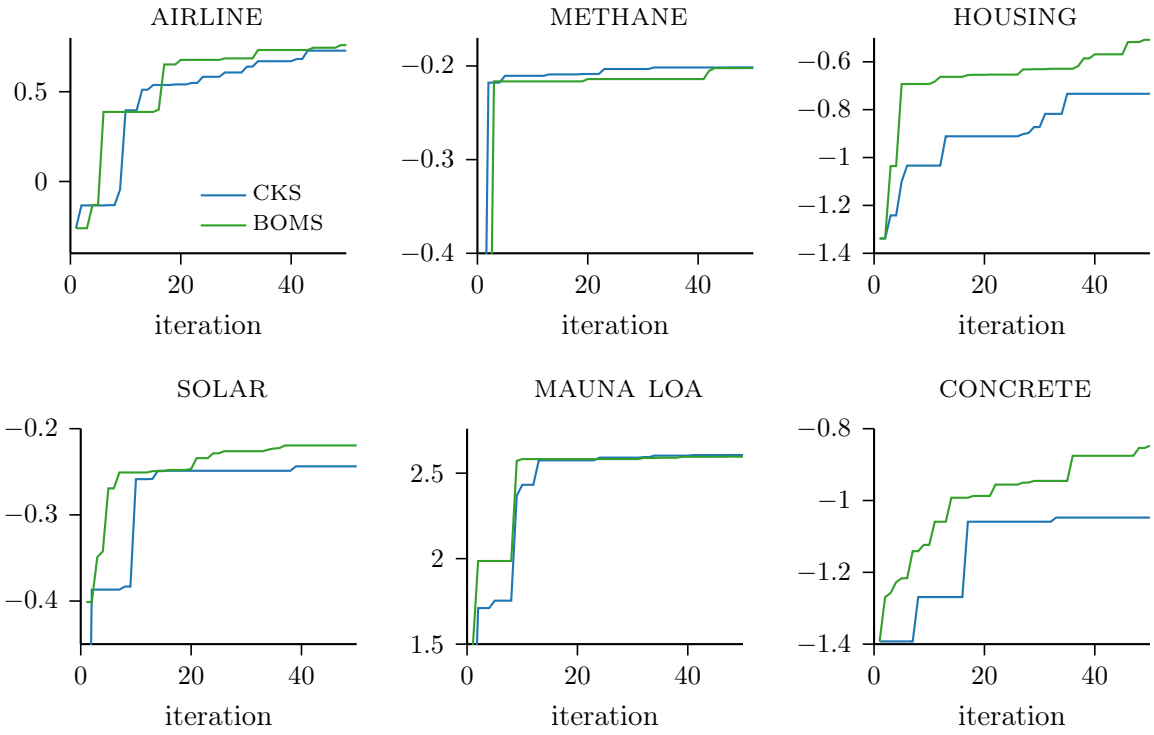


Figure 5.2: A plot of the best model evidence found (normalized by $|\mathcal{D}|$, (5.7)) as a function of the number of models evaluated, $g(\mathcal{M}^*; \mathcal{D})$, for six of the datasets considered (identical vertical axis labels omitted for greater horizontal resolution).

Model search

We also evaluate our method's ability to quickly find a suitable model to explain a given dataset. We compare our approach with the greedy *compositional kernel search* (CKS) of [13]. Both algorithms used the same kernel grammar (§5.2), hyperparameter priors (§5.2), and model

evidence approximation (§2.2, (5.1)). We used L-BFGS to optimize model hyperparameters, using multiple restarts to avoid bad local maxima; each restart begins from a sample from $p(\theta \mid \mathcal{M})$.

For BOMS, we always began our search evaluating SE first. The active set of models $\mathcal{C}$ (§5.2) was initialized with all models that are at most two edges distant from the base kernels. To avoid unnecessary re-training over g, we optimized the hyperparameters of μ_g and K_g every 10 iterations. This also allows us to perform rank-one updates for fast inference during the intervening iterations.

Results are depicted in Figure 5.2 for a budget of 50 evaluations of the model evidence. In four of the six datasets we substantially outperform CKS. Note the vertical axis is in the log domain. The overhead for computing the kernel K_g and performing the inference about g was approximately 10% of the total running time. On MAUNA LOA our method is competitive since we find a model with similar quality, but earlier. The results for METHANE, on the other hand, indicate that our search initially focused on a suboptimal region of the graph, but we eventually do catch up.

5.4 Conclusion

We introduced a novel automated search for an appropriate kernel to explain a given dataset. Our mechanism explores a space of infinite candidate kernels and quickly and effectively selects a promising model. Focusing on the case where the models represent structural assumptions in GPs, we introduced a novel "kernel kernel" to capture the similarity in prior explanations that two models ascribe to a given dataset. We have empirically demonstrated

that our choice of modeling the evidence (or marginal likelihood) with a GP in model space is capable of predicting the evidence value of unseen models with enough fidelity to effectively explore model space via Bayesian optimization.

Chapter 6

Automated Active learning

In the previous chapter, we developed an automated model search mechanism for fixed-size datasets. Here, we want to extend this approach to active learning. Model selection is particularly difficult in active learning problems, as only a small amount of training data may be available. It is difficult to choose a model to describe the data that one has not seen yet; unless a human expert can provide assistance. In the following, we develop an automated framework for active learning. We focus on automating active optimization but these ideas can be easily extended to any active learning setting.

6.1 Automated Bayesian optimization

Global optimization of expensive, potentially gradient-free functions has long been a critical component of many complex problems in science and engineering. As an example, imagine that we want to tune the hyperparameters of a deep neural network in a self-driving car. That is, we want to maximize the generalization performance of the machine learning algorithm, but the functional form of the objective function f is *unknown* and even a single function

evaluation is *costly* — it might take hours (or even days!) to train the network. These features render the optimization particularly difficult.

Bayesian optimization has nonetheless shown remarkable success on optimizing expensive gradient-free functions [40, 6, 78]. Bayesian optimization works by maintaining a probabilistic belief about the objective function and designing a so-called *acquisition function* that intelligently indicates the most-promising locations to evaluate f next. Although the design of acquisition functions has been the subject of a great deal of research, how to appropriately model f has received comparatively less attention [8, 78], despite being a decisive factor for performance. In fact, this was considered *the most important problem* in Bayesian optimization by Močkus [62], in a seminal work in the field:

> "The development of some system of *a priori* distributions suitable for different classes of the function f is probably the most important problem in the application of [the] Bayesian approach to ... global optimization" (Močkus 1974, p. 404).

We develop a search mechanism for appropriate surrogate models (prior distributions) to the objective function f. Our model-search procedure operates via Bayesian optimization in model space. Our method does not prematurely commit to a single model; instead, it uses several models to form a belief about the objective function and plan where the next evaluation should be. Our adaptive model averaging approach accounts for model uncertainty, which more realistically copes with the limited information available in practical Bayesian optimization applications. In Figure 1, we show two instances of Bayesian optimization where our goal is to maximize the red objective function f. Both instances use expected improvement as acquisition function. The difference between is the belief about f: using a single model (left) or combining several models using our automated Bayesian optimization

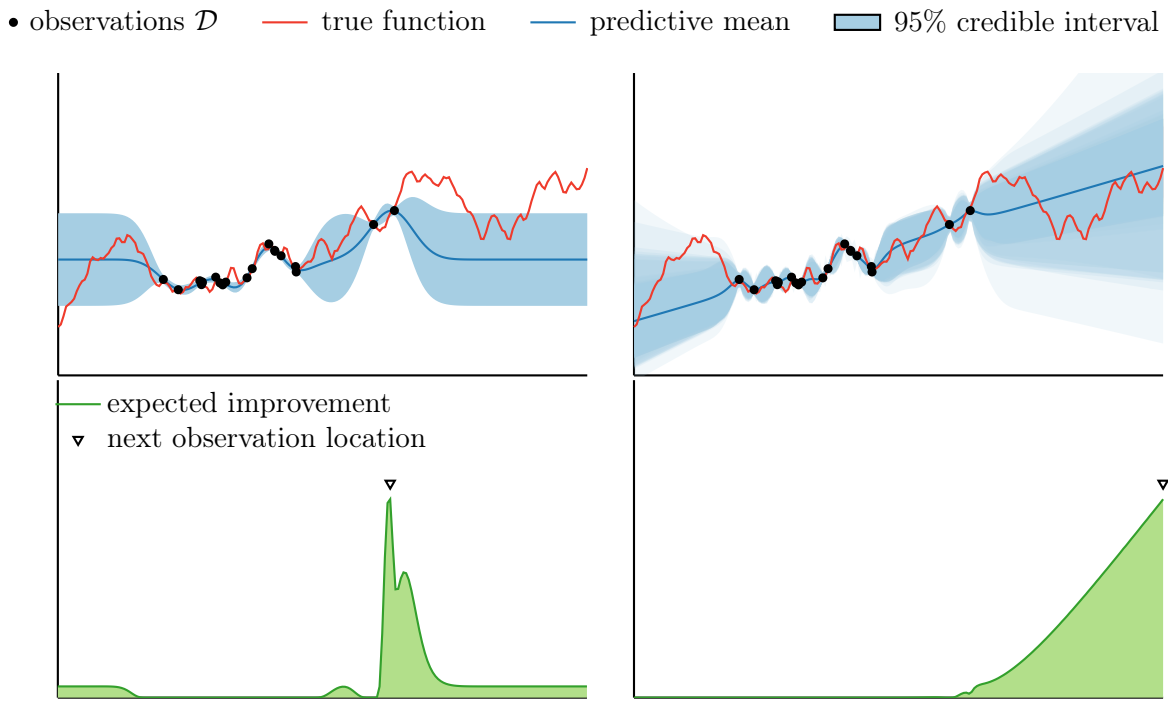


Figure 6.1: Importance of model selection in Bayesian optimization. Top left: one model represents the belief about the objective. Top right: a mixture of models selected by our approach represents the belief about f. Bottom: the acquisition function value (expected improvement) computed using the respective beliefs about the objective. ABO places the next observation at the optimum.

(ABO) approach (right). A single model does not capture the nuances of the true function. In contrast, ABO captures the linear increasing trend of the true function and produces a credible interval which successfully captures the function's behavior. Consequently, ABO finds the optimum in the next iteration.

Finally, we demonstrate empirically that our approach is consistently competitive with or outperforms other strong baselines across several domains: benchmark functions for global optimization functions, hyperparameter tuning of machine learning algorithms, reinforcement learning for robotics, and determining cosmological parameters of a physical model of the Universe.

Related Work

Our approach is inspired by some recent developments in the field of automated model selection [58, 13, 28]. Here, we take these ideas one step further and consider automated model selection when actively acquiring new data.

Gardner et al. [16], Kulick et al. [44] also tackled the problem of model selection in an active learning context but with a different goal. Given a *fixed set* of candidate models, the authors proposed a method for gathering data to quickly identify which model best explains the data. Here our ultimate goal is to perform global optimization (6.1) when we can dynamically change our set of models. In future work, it would be interesting to examine whether it may be possible to combine our ideas with their proposed method to actively learn in model space.

More recently, Gardner et al. [18] developed an automated model search for Bayesian optimization similar to our method. Their approach, however, uses a MCMC strategy for sampling new promising models, whereas we adapt the Bayesian optimization search of

proposed by Malkomes et al. [58]. We will discuss further differences between our approach and their MCMC method in the next section.

6.2 Bayesian optimization with multiple models

Suppose we want to optimize an expensive, perhaps black-box function $f\colon \mathcal{X} \to \mathbb{R}$ on some compact set $X \subseteq \mathcal{X}$. We may query f at any point $\mathbf{x}$ and observe a possibly noisy value $y = f(\mathbf{x}) + \varepsilon$. Our ultimate goal is to find the global optimum:

$$\mathbf{x}_{\text{OPT}} = \arg\min_{\mathbf{x} \in X} f(\mathbf{x}) \tag{6.1}$$

through a sequence of evaluations of the objective function f. This problem becomes particularly challenging when we may only make a limited number of function evaluations, representing a real-world *budget* B limiting the total cost of evaluating f. Throughout this text, we denote by $\mathcal{D}$ a set of gathered observations $\mathcal{D} = (\mathbf{X}, \mathbf{y})$, where $\mathbf{X}$ is a matrix aggregating the input variables $\mathbf{x}_i \in X$, and $\mathbf{y}$ is the respective vector of observed values $y_i = f(\mathbf{x}_i) + \varepsilon$.

Modeling the objective function. Assume we are given a prior distribution over the objective function $p(f)$ and, after observing new information, we have means of updating our belief about f using Bayes' rule:

$$p(f \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid f)p(f)}{p(\mathcal{D})}. \tag{6.2}$$

The posterior distribution above is then used for decision making, i.e., selecting the $\mathbf{x}$ we should query next. When dealing with a single model, the posterior distribution (6.2) suffices.

Here, however, we want to make our model of f more flexible, accounting for potential misspecification. Suppose we are given a collection of probabilistic models $\{\mathcal{M}_i\}$ that offer plausible explanations for the data. Each model $\mathcal{M}$ is a set of probability distributions indexed by a parameter θ from the corresponding model's parameter space $\Theta_{\mathcal{M}}$. With multiple models, we need a means of aggregating their beliefs. We take a fully Bayesian approach and we use the *model evidence* (or *marginal likelihood*), the probability of generating the observed data given a model $\mathcal{M}$,

$$p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}) = \int p(\mathbf{y} \mid \mathbf{X}, \theta, \mathcal{M})\, p(\theta \mid \mathcal{M})\, \mathrm{d}\theta, \tag{6.3}$$

as the key quantity for measuring the fit of each model to the data. The evidence integrates over the parameters θ to compute the probability of the model generating the observed data under a hyperprior distribution $p(\theta \mid \mathcal{M})$. Given (6.3), one can easily compute the *model posterior*,

$$p(\mathcal{M} \mid \mathcal{D}) = \frac{p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})p(\mathcal{M})}{p(\mathbf{y} \mid \mathbf{X})} = \frac{p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})p(\mathcal{M})}{\sum_i p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}_i)p(\mathcal{M}_i)}, \tag{6.4}$$

where $p(\mathcal{M})$ represents a prior probability distribution over the models. The model posterior gives us a principled way of combining the beliefs of all models. Our model of f can now be summarized with the following model-marginalized posterior distribution:

$$p(f \mid \mathcal{D}) = \sum_i p(\mathcal{M}_i \mid \mathcal{D}) \underbrace{\int p(f \mid \mathcal{D}, \theta, \mathcal{M}_i) p(\theta \mid \mathcal{D}, \mathcal{M}_i)\, \mathrm{d}\theta}_{p(f \mid \mathcal{D}, \mathcal{M}_i)}. \tag{6.5}$$

Note that (6.5) takes into consideration all plausible models $\{\mathcal{M}_i\}$ and the integral $p(f \mid \mathcal{D}, \mathcal{M}_i)$ accounts for the uncertainty in each model's hyperparameters $\theta \in \Theta_{\mathcal{M}_i}$. Unfortunately, the latter is often intractable and we will discuss means of approximating it in Section 6.4.

Next, we describe how to use the model-marginalized posterior to intelligently optimize the objective function.

Selecting where to evaluate next. Given our belief about f, we want to use this information to select which point $\mathbf{x}$ we want to evaluate next. This is typically done by maximizing an *acquisition function* $\alpha\colon \mathcal{X} \to \mathbb{R}$. Instead of solving (6.1) directly, we optimize the proxy (and simpler) problem

$$\mathbf{x}^* = \arg\max_{\mathbf{x} \in X} \alpha(\mathbf{x}; \mathcal{D}). \tag{6.6}$$

We use *expected improvement* (EI) [62] as our acquisition function. Suppose that f' is the minimal value observed so far.[14] EI selects the point $\mathbf{x}$ that, in expectation, improves upon f' the most:

$$\alpha_{\text{EI}}(\mathbf{x}\,; \mathcal{D}, \mathcal{M}) = \mathbb{E}_y\big[\max(f' - y, 0) \mid \mathbf{x}, \mathcal{D}, \mathcal{M}\big]. \tag{6.7}$$

Note that if $p(y \mid \mathbf{x}, \mathcal{D}, \mathcal{M})$ is a Gaussian distribution (or can be approximated as one), the expected improvement can be computed in closed form. Usually, acquisition functions are evaluated for a given model choice $\mathcal{M}$. As before, we want to incorporate multiple models in this framework. For EI, we can easily take into account all models as follows:

$$\alpha_{\text{EI}}(\mathbf{x}\,; \mathcal{D}) = \mathbb{E}_{y,\mathcal{M}}\big[\max(f' - y, 0) \mid \mathbf{x}, \mathcal{D}\big] = \mathbb{E}_{\mathcal{M}}\big[\alpha_{\text{EI}}(\mathbf{x}\,; \mathcal{D}, \mathcal{M})\big]. \tag{6.8}$$

We could also derive similar results for other acquisition functions such as probability of improvement [45] and GP upper confidence bound (GP-UCB) [79].

[14] We make a simplifying assumption that the noise level is small, thus $f' \approx \min_i \mu_{f|\mathcal{D}}(\mathbf{x}_i)$ and $y(\mathbf{x}) \approx f(\mathbf{x})$.

6.3 Automated model selection for fixed-size datasets

Before introducing our automated method for Bayesian optimization, we review some key concepts from the previous chapter.

Space of models. First we need a space of GP models that is general enough to explain virtually any dataset. We adopt the generative kernel grammar of [13] due to its ability to create arbitrarily complex models. We start with a set of so-called *base* (one-dimensional) kernels, such as the common squared exponential (SE) and rational quadratic (RQ) kernels. Then, we create new and potentially more complex kernels by summation and multiplication, over individual dimensions, of the base kernels. This let us create kernels over multidimensional inputs. As a result, we have a space of kernels that allows one to search for appropriate structures (different kernel choices) as well as relevant features (subsets of the input). Now, we need an efficient method for searching models from this given space.

Bayesian optimization for model search. Suppose we are given a space of probabilistic models $\mathbb{M}$ such as the above-cited generative kernel grammar. As mentioned before, the key quantity for model comparison in a Bayesian framework is the model *evidence* (6.3). Previous work has shown that we can search for promising models $\mathcal{M} \in \mathbb{M}$ by viewing the evidence as a function $g\colon \mathbb{M} \to \mathbb{R}$ to be optimized. Our method consists of a Bayesian optimization approach to model selection (BOMS), in which we try to find the optimal model

$$\mathcal{M}_{\text{OPT}} = \arg\max_{\mathcal{M} \in \mathbb{M}} g(\mathcal{M}; \mathcal{D}), \tag{6.9}$$

where $g(\mathcal{M}; \mathcal{D})$ is the (log) model evidence: $g(\mathcal{M}; \mathcal{D}) = \log p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})$. Two key aspects to remember are: the unusual GP prior, $p(g) = \mathcal{GP}(g; \mu_g, K_g)$, where the mean and covariance

functions are appropriately defined over the model space $\mathbb{M}$; and the heuristic for traversing $\mathbb{M}$ by maintaining a set of candidate models $\mathcal{C}$. The precise mechanism for traversing the space of models is not particular relevant for this exposition, but the fact that $\mathcal{C}$ is changing as we search for better models is.

6.4 Automating Bayesian optimization with Bayesian optimization

Here, we present our automated Bayesian optimization (ABO) algorithm. ABO is a two-level Bayesian optimization procedure. The "outer level" solves the standard Bayesian optimization problem, where we want to search for the optimum of the objective function f. Inside the Bayesian optimization loop, we use a second "inner" Bayesian optimization, where the goal is to search for appropriate models $\{\mathcal{M}_i\}$ to the objetive function f. The inner optimization seeks models maximizing the model evidence as in BOMS (Section 3). The motivation is to refine the set of models $\{\mathcal{M}_i\}$ before choosing where we want to query the (expensive) objective function f next. Given a set of models, we can use the methodology presented in Section 2 to perform Bayesian optimization with multiple models.

In the next subsection, we will describe the inner Bayesian optimization method which we refer to as active BOMS (ABOMS). Before going to the second Bayesian optimization level, we summarize ABO in Algorithm 1. First, we initialize our set of promising models $\{\mathcal{M}_i\}$ with random models chosen from the grammar of kernel, same used in [13]. To select these models, we perform random walks from the the empty kernel and repeatedly apply a random number of grammatical operations. The number of operations is sampled from a geometric

Algorithm 1 Automated Bayesian Optimization

Input: function f, budget B, initial data $\mathcal{D}$
$\{\mathcal{M}_i\} \leftarrow$ Initial set of promising models
repeat
  $\{\mathcal{M}_i\} \leftarrow$ update models $(\{\mathcal{M}_i\}, \mathcal{D})$
  $\{\mathcal{M}_i\} \leftarrow$ ABOMS$(\{\mathcal{M}_i\}, \mathcal{D})$
  $p(\mathcal{M} \mid \mathcal{D}) \leftarrow$ compute model posterior
  discard irrelevant models $p(\mathcal{M}_i \mid \mathcal{D}) < 10^{-4}$
  $\mathbf{x}^* \leftarrow \arg\max_{\mathbf{x} \in X} \alpha_{\text{EI}}(\mathbf{x}\,; \mathcal{D})$.
  $y^* \leftarrow f(\mathbf{x}^*) + \varepsilon$
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}^*, y^*)\}$
until budget B is depleted

distribution with termination probability of $\frac{1}{3}$. Then, at each iteration: we update all models with current data, computing the corresponding model evidence of each model; use ABOMS (the inner model-search optimization) to include more promising candidate models in $\{\mathcal{M}_i\}$; exclude all models that are unlikely to explain the current data, those with $p(\mathcal{M} \mid \mathcal{D}) < 10^{-4}$; sample the function at location $\mathbf{x}^*$ using (6.8) and all models $\{\mathcal{M}_i\}$; finally, we evaluate $y^* = f(\mathbf{x}^*) + \varepsilon$ and include this new observation in our dataset.

Active Bayesian optimization for model search

The critical component of ABO is the inner optimization procedure that searches for suitable models to the objective function: the active Bayesian optimization for model search (ABOMS). Notice that the main challenge is that ABOMS is nested in a Bayesian optimization loop, meaning that both *data* and *models* will change as we perform more outer Bayesian optimization iterations.

Suppose we already gathered some observations $\mathcal{D}$ of the objective function f. Additionally, we use the previously proposed BOMS (Section 3) as the inner model search procedure.

Inside BOMS, we tried different models, gathering observations of the (log) model evidence, $g(\mathcal{M}; \mathcal{D}) = \log p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})$. We denote by $\mathcal{D}_g = \{\mathcal{M}_j, g(\mathcal{M}_j; \mathcal{D})\}$ the observations of the inner Bayesian optimization. After one loop of the outer Bayesian optimization, we obtain new data $\mathcal{D}' = \mathcal{D} \cup \{(\mathbf{x}^*, y^*)\}$. Now, the model evidence of all previously evaluated models $\mathcal{D}_g$ changes since $g(\mathcal{M}_j, \mathcal{D}) \neq g(\mathcal{M}_j, \mathcal{D}')$ for all j. As a result, we would have to retrain all models in $\mathcal{D}_g$ to correctly compare them. Recall that there are good models in $\mathcal{D}_g$ for explaining the objective function f. These models will be passed to the outer Bayesian optimization, where they will be updated — ultimately, we want to provide outstanding suggestions $\mathbf{x}^*$ for where to query f next, thus they need to be retrained. A *large portion* of the tested models in $\mathcal{D}_g$, however, are not appropriated for modeling f; in fact, they can be totally ignored by the outer optimization. Yet these "bad" models can help guide the search toward more-promising regions of model space. How to retain information from previously evaluated models without resorting to exhaustive retraining?

Our answer is to modify BOMS in two ways. First, we place a GP on the *normalized* model evidence, $g(\mathcal{M}; \mathcal{D}) = \log p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}) \,/\, |\mathcal{D}|$, which let us compare models across iterations. Second, we assume that each evidence evaluation is *corrupted by noise*, the variance of which depends on the number of data points used to compute it: the more data we use, the more accurate our estimate, and the lower the noise. More specifically, we use the same GP prior of discussed in the previous chapter, $p(g) = \mathcal{GP}(g; \mu_g, K_g)$, where $\mu_g \colon \mathbb{M} \to \mathbb{R}$ is just a constant mean function and $K_g \colon \mathbb{M}^2 \to \mathbb{R}$ is the "kernel kernel" defined as a squared exponential kernel that uses the (averaged) Hellinger distance between the inputs as oppose to the standard ℓ_2 norm. Our observation model, however, assumes that the observations of the normalized model evidence are corrupted by heterogenous noise:

$$y_g(\mathcal{M}; \mathcal{D}_n) = \frac{g(\mathcal{M}; \mathcal{D}_n)}{n} + \varepsilon\Big(\frac{1}{n}\Big). \tag{6.10}$$

To choose the amount of noise, we observed that, using the chain rule, the marginal likelihood can be written as

$$\log p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}) = \sum_i \log p\big(y_i \mid \mathbf{x}_i, \big\{(\mathbf{x}_j, y_j) \mid j < i\big\}, \mathcal{M}\big), \tag{6.11}$$

which is the sum of the marginal predictive log likelihoods for the points in the $\mathcal{D}$. When we divide $\log p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})$ by $|\mathcal{D}| = n$, we can *interpret* the result as an estimate of the average predictive log marginal likelihood.[15] Therefore, if

$$\log p(\mathbf{y} \mid \mathbf{X}, \mathcal{M})/n \approx \mathbb{E}\big[\log p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}) \mid \mathcal{M}\big],$$

then the variance of this estimate with n measurements is

$$\mathrm{Var}\Big[\log p\big(y_i \mid \mathbf{x}_i, \big\{(\mathbf{x}_j, y_j) \mid j < i\big\}, \mathcal{M}\big)\Big]/n.$$

which shrinks like σ_g^2/n for a small constant σ_g (e.g., 0.5). For large n it goes to 0. This mechanism gracefully allows us to condition on the history of all previously proposed models during the search. By modeling earlier evidence computations as noisier, we avoid recomputing the model evidence of previous models every round, but we still make the search for good models better informed.

Implementation

In practice, several distributions presented above are often intractable for GPs. Now, we discuss how to efficiently approximate these quantities. First, instead of using just a delta

[15] Note that the training data is not independent since we are choosing the locations $\mathbf{x}$, and we are not assuming that $n \to \infty$

approximation to the hyperparameter posterior $p(\theta \mid \mathcal{D}, \mathcal{M})$, e.g. MLE/MAP, we use a Laplace approximation, i.e., we make a second-order Taylor expansion around its mode: $\hat{\theta} = \arg\max_\theta \log p(\theta \mid \mathcal{D}, \mathcal{M})$. This results in a multivariate Gaussian approximation:

$$p(\theta \mid \mathcal{D}, \mathcal{M}) \approx \mathcal{N}(\theta; \hat{\theta}, \Sigma) \text{ where } \Sigma^{-1} = -\nabla^2 \log p(\theta \mid \mathcal{D}, \mathcal{M})\big|_{\theta=\hat{\theta}}.$$

Conveniently, the Laplace approximation also give us a means of approximating the model evidence:

$$\log p(\mathbf{y} \mid \mathbf{X}, \mathcal{M}) \approx \log p(\mathbf{y} \mid \mathbf{X}, \hat{\theta}, \mathcal{M}) \; + \log p(\hat{\theta} \mid \mathcal{M}) - \tfrac{1}{2} \log \det \Sigma^{-1} + \tfrac{d}{2} \log 2\pi,$$

where d is the dimension of θ. The above approximation can be interpreted as rewarding explaining the data well while penalizing model complexity [63, 67].

Next consider the posterior distribution $p(f \mid \mathcal{D}, \mathcal{M})$, which is an integral over the model's hyperparameters. This distribution is intractable, even with our Gaussian approximation to the hyperparameter posterior $p(\theta \mid \mathcal{D}, \mathcal{M}) \approx \mathcal{N}(\theta; \hat{\theta}, \Sigma)$. We use a general approximation technique originally proposed by [65] (Section 4) in the context of Bayesian quadrature. This approach assumes that the posterior mean of $p(f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M})$ is affine in θ around $\hat{\theta}$ and the GP covariance is constant. Let

$$\mu^*(\theta) = \mathbb{E}[f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M}] \quad \text{and} \quad \nu^*(\theta) = \mathrm{Var}[f^* \mid \mathbf{x}^*, \mathcal{D}, \theta, \mathcal{M}]$$

be the posterior predictive mean and variance of f^*. The result of this approximation is that the posterior distribution of f^* is approximated by

$$p(f^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}) \approx \mathcal{N}\big(f^*; \mu^*(\hat{\theta}), \sigma^2_{\text{AFFINE}}\big), \quad \text{where} \quad \sigma^2_{\text{AFFINE}} = \big[\nabla\mu^*(\hat{\theta})\big]^\top \Sigma \big[\nabla\mu^*(\hat{\theta})\big].$$

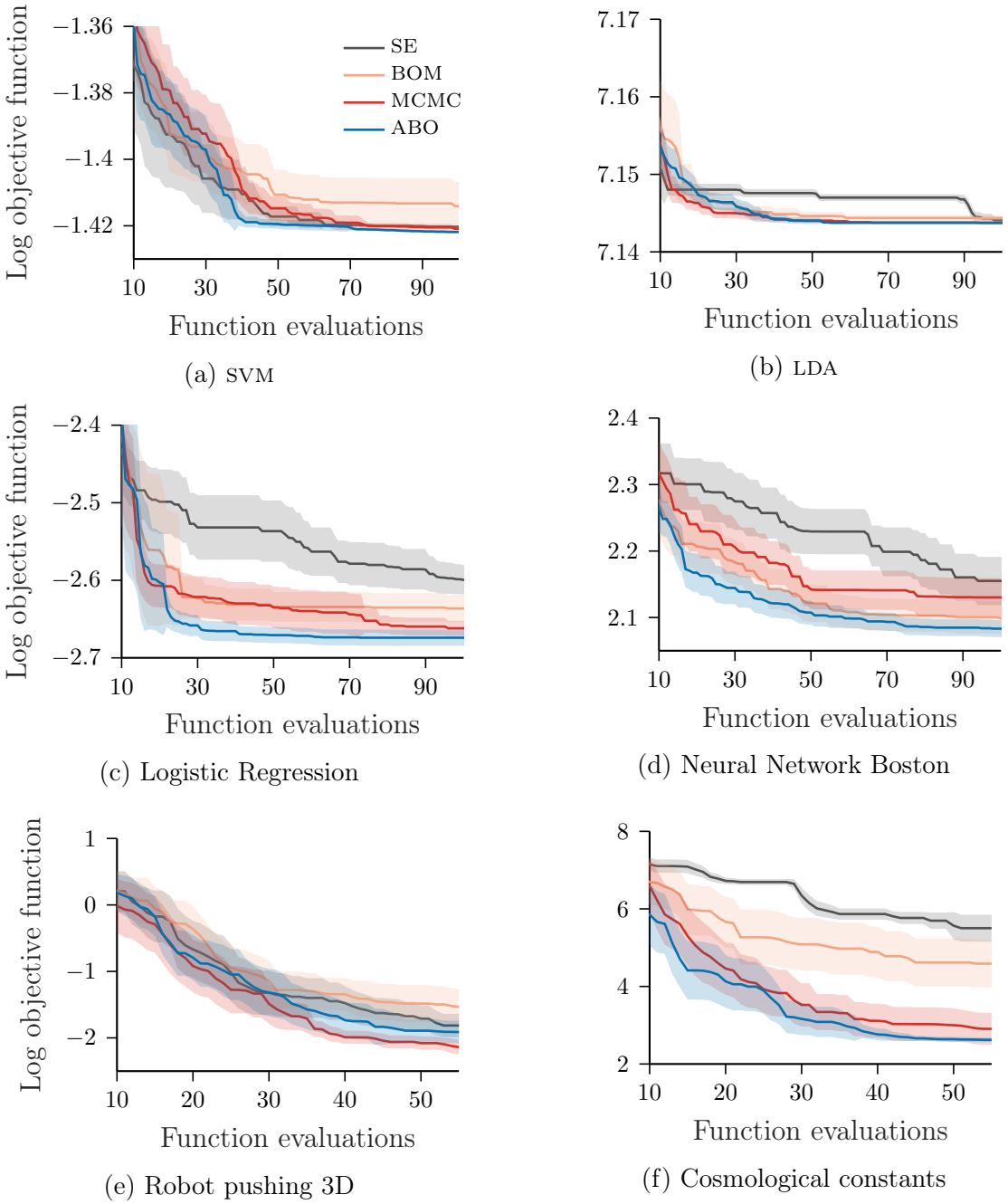
This approach was shown to be a good alternative for propagating the uncertainty in the hyperparameters [65]. Finally, given the Gaussian approximations above (6.12), we use standard techniques to analytically approximate the predictive distribution:

$$p(y^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M}) = \int p(y^* \mid f^*)\, p(f^* \mid \mathbf{x}^*, \mathcal{D}, \mathcal{M})\, \mathrm{d}f^*.$$

Our code and data will be available online: `https://github.com/gustavomalkomes/abo`.

6.5 Empirical Results

We validate our approach against several optimization alternatives and across several domains. Our first baseline is a random strategy that selects twice as many locations as the other methods. We refer to this strategy as RANDOM 2× [48]. We also consider a competitive Bayesian optimization implementation which uses a single non-isotropic squared exponential kernel (SE), expected improvement as the acquisition function and all the approximations described in Section 6.4. Then, we considered two more baselines that represent the uncertainty about the unknown function through a combination of multiple models. One baseline uses the same collection of predefined models throughout its execution; we refer to this approach as the bag of models (BOM). The other is an adaptation of the method proposed in [18], here referred as MCMC, which, similar to BOMS, is allowed to dynamically select more models every iteration. Instead of using the additive class of models proposed in the original work, we adapted their Metropolis–Hastings algorithm to the more-general compositional grammar proposed by Duvenaud et al. [13], which is also used by our method. This choice lets us compare which adaptive strategy performs better in practice. Specifically, given an initial



(a) SVM

(b) LDA

(c) Logistic Regression

(d) Neural Network Boston

(e) Robot pushing 3D

(f) Cosmological constants

Figure 6.2: Averaged minimum observed function value and standard error of all methods for several objective functions. For better visualization, we omit the first 10 function evaluations since they are usually much higher than the final observations.

model $\mathcal{M}$, the MCMC proposal distribution randomly selects a neighboring model $\mathcal{M}'$ from the grammar. Then we compute the acceptance probability as in [18].

All multiple models strategies (BOM, MCMC and BOMS) start with the same selection of models (See Section 6.4) and they aim to maximize the model-marginalized expected improvement (6.8). Both adaptive algorithms (BOMS and MCMC) are allowed to perform five model evidence computations before each function evaluation; BOMS queries five new models and MCMC performs five new proposals. In our experiments, we limited the number of models to 50, always keeping those with the higher model evidence. Model choice and acquisition functions apart, we kept all configurations the same. All methods used L-BFGS to optimize each model's hyperparameters. To avoid bad local minima, we perform two restarts, each begining from a sample of $p(\theta \mid \mathcal{M})$. All the approximations described in Section 6.4 were also used. We maximized the acquisition functions by densely sampling $1000d^2$ points from a d-dimensional low-discrepancy Sobol sequence, and starting MATLAB `fmincon` (a local optimizer) from the sampled point with highest value. Each experiment, was repeated 20 times with five random initial examples, which were the same for all Bayesian optimization methods. RANDOM $2\times$ results were averaged across 1000 repetitions.

Benchmark functions for global optimization.

Our first set of experiments are test functions commonly used as benchmarks for optimization [82]. We adopted a similar setup as previous works [27] but included more test functions. The goal is to find the global minimum of each test function given a limited number of function evaluations. We provide more information about the chosen functions in the supplementary material. The maximum number of function evaluations was limited to 10 times the dimensionality of the function domain being optimized.

Table 6.1: Results for the average gap performance across 20 repetitions for different test functions and methods. RANDOM 2× (R 2×) results are averaged across 1000 experiments. Numbers that are *not* significantly different from the highest average gap for each function are bolded (one-sided paired Wilcoxon signed rank test, 5% significance level).

	function	d	R 2×	SE	BOM	MCMC	ABO
synthetic objectives	Ackley $2d$	2	0.422	0.717	**0.984**	**0.988**	**0.980**
	Beale	2	0.725	0.541	**0.644**	**0.596**	**0.688**
	Branin	2	0.743	**1.000**	0.950	0.996	0.998
	Eggholder	2	0.461	**0.516**	**0.529**	**0.546**	**0.579**
	Six-Hump Camel	2	0.673	0.723	**0.988**	**0.992**	**0.998**
	Drop-Wave	2	**0.458**	**0.496**	**0.421**	**0.447**	**0.481**
	Griewank $2d$	2	0.669	**0.924**	**0.954**	**0.941**	**0.964**
	Rastrigin	2	0.538	0.410	**0.832**	**0.827**	**0.850**
	Rosenbrock	2	0.787	**1.000**	0.999	0.993	0.999
	Shubert	2	0.337	**0.384**	**0.374**	**0.332**	**0.481**
	Hartmann	3	0.682	**1.000**	**0.970**	0.999	**1.000**
	Levy	3	0.669	0.774	**0.913**	0.942	**0.971**
	Rastrigin $4d$	4	0.414	0.261	**0.823**	0.715	**0.821**
	Ackley $5d$	5	0.299	0.736	0.409	**0.886**	0.809
	Griewank $5d$	5	0.605	**0.971**	0.756	**0.974**	0.968
	mean gap		0.566	0.697	0.770	0.812	0.839
	median gap		0.605	0.723	0.832	0.941	0.964
real-world objectives	SVM	3	0.903	0.912	0.840	0.938	**0.956**
	LDA	3	0.939	**0.950**	0.925	**0.950**	**0.950**
	Logistic regression	4	0.928	0.774	0.899	0.936	**0.994**
	Robot pushing $3d$	3	0.815	0.927	0.878	**0.967**	0.935
	Robot pushing $4d$	4	**0.824**	**0.748**	0.619	**0.668**	**0.715**
	Neural network Boston	4	0.491	0.594	0.703	0.640	**0.757**
	Neural network cancer	4	**0.845**	**0.645**	**0.682**	**0.773**	**0.749**
	Cosmological constants	9	0.739	0.848	0.859	0.984	**0.999**
	mean gap		0.810	0.800	0.801	0.857	0.882
	median gap		0.834	0.811	0.850	0.937	0.943

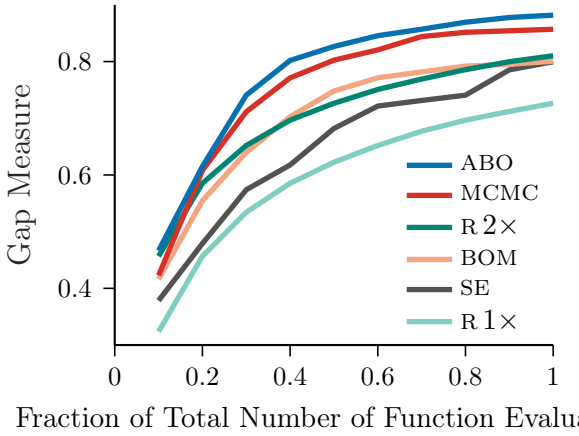


Figure 6.3: Average gap across the eight real-world objective functions vs. fraction of total number of function evaluations. Here, we display the performance of random search (R 1×) for reference.

We report the gap measure [34], defined as $\frac{f(x_{\text{first}})-f(x_{\text{best}})}{f(x_{\text{first}})-f(x_{\text{OPT}})}$, where $f(x_{\text{first}})$ is the minimum function value among the first initial random points, $f(x_{\text{best}})$ is the best value found by the method, and $f(x_{\text{OPT}})$ is the optimum.

Table 6.1 (top) shows the results for different functions and methods. For each test function, we perform a one-sided Wilcoxon signed rank test at the 5% significance level with each method and the one that had the highest average performance. All results that are *not* significantly different than the highest are marked in bold. First, note that RANDOM 2× performs poorly in these synthetic constructed "hard" functions. Then, observe that the overall performance of all multi-model methods is higher than the single GP baseline, with BOMS leading these algorithms with respect to the mean and median gap performance overall. In fact, ABO's performance is comparable to the best method for 11 out of 15 functions.

Real-world optimization functions. To further investigate the importance of model search, we consider a second set of functions used in recent publications. Our goal was

to select a diverse and challenging set of functions which might better demonstrate the performance of these algorithms in a real application. More information about these functions is given in the supplementary material.

We show the gap measure for the second set of experiments in Table 6.1 (bottom) and perform the same statistical test as before. For computing the gap measure in these experiments, when the true global minimum is unknown, we used the minimum observed value across all experiments as a proxy for the optimal value. In Figure 6.3 we show the average gap measure across all eight test functions as a function of the total number of functions evaluations allowed. In Figure 6.2, we show the averaged minimum observed function value and standard error of all methods for 6 out of the 8 functions (see supplementary material for the other two functions).

With more practical objective functions, the importance of model search becomes more clear. ABO either outperforms the other methods (4 out of the 8 datasets) or achieves the lowest objective function. Figure 6.3 also shows that ABO quickly advances on the search for the global minimum — on average, the gap measure is higher than 0.8 after at half of the budget. Interestingly, RANDOM 2× also performs well for 2 out of these 8 datasets, those are the problems in which all methods have a similar performance, suggesting that these functions are easier to optimize than the others.

Naturally, training more models and performing an extra search to dynamically select models require more computation than running a standard single Bayesian optimization. In our implementation, not optimized for speed, the median wall clock across all test functions for *updating* and searching the five new models was 65 and 41 seconds, respectively, for MCMC and ABO. Note that the model update is what dominates this procedure for both methods, with MCMC tending to select more complex models than ABO. In practice, one could perform

this step in parallel with the expensive objective function evaluation, requiring no additional overhead besides the cost of optimizing the model-marginal acquisition function, which can also be adjusted by the user.

6.6 Conclusion

We introduced a novel automated Bayesian optimization approach that uses multiple models to represent its belief about an objective function and subsequently decide where to query next. Our method automatically and efficiently searches for better models as more data is gathered. Empirical results show that the proposed algorithm often outperforms the baselines for several different objective functions across multiple applications. We hope that this work can represent a step towards a fully automated system for Bayesian optimization that can be used by a nonexpert on arbitrary objectives.

Chapter 7

Discussion and Future Directions

In this dissertation, we developed four novel active learning algorithms offering automated solutions for real-world problems in science, medicine, and engineering. Our algorithms were shown to achieve superior empirical performance than previously proposed approaches. Additionally, they are easy to use and do not require extensive machine-learning expertise to be used in applications.

In our first contribution, we investigated *active search*—an active learning paradigm that, among other applications, can be used to facilitate drug discovery. We gave a theoretical hardness result for active search and proposed a novel sample-efficient, computational-tractable policy for this problem. In more details, we proved that no polynomial-time algorithm can approximate the expected utility of the optimal policy within a constant approximation ratio. We then proposed a novel method, known as efficient nonmyopic search (ENS), for active search. Our method approximates the Bayesian optimal policy by computing, conditioned on the location of the next candidate point, how many additional targets are expected at termination if the remaining budget is spent simultaneously. By always considering the remaining budget, and approximately accounting for the impact of our current choice in future observations, our approach displays a marvelous trade-off between exploration and

exploitation. We also derived an effective pruning strategy that can significantly improve ENS' efficiency in practical applications. We conducted an extensive set of experiments that demonstrated the superior overall performance of our approach on various domains. More recently, we have extended ENS to batch active learning, where multiple points are selected simultaneously [39]. We have demonstrated that active search is, in general, a hard problem. However, one exciting future direction is to understand, under what conditions (*e.g.*, assumptions about the structure of problem instances) we can find efficient algorithms with theoretical guarantees. We could also investigate different strategies for using our approximation to the Bayesian optimal policy, for example, combining ENS with an A^* algorithm could result in a more efficient policy.

Our second contribution was to introduce a novel information-theoretic approach for active model selection, named *Bayesian active model selection*. Specifically, our approach is based on the mutual information between the output variable and the model class. Different from previous approaches, our method does not require extensive model retraining when evaluating each prospective next location, resulting in an efficient active model selection policy. Further, we provided an effective and efficient analytic approximation to our policy that can be used for automatically learning the model class of Gaussian processes with arbitrary observation likelihoods. Then, we successfully apply this technique to active structure discovery and to rapid screening for noise-induced hearing loss, showing that our approach is sample-efficient. Bayesian active model selection is a general framework that can be extended to many exciting applications. Vision tests and other psychometric tests could be automated under the same principles, allowing fast screening of large populations with relatively low-cost. Another promising direction is to investigate non-myopic policies for this problem, as well as strategies for querying multiple points.

Our third contribution was to introduce a novel search mechanism for model selection. Here we creatively framed model selection as an active optimization problem on the model space. We then adapted standard Bayesian optimization principles to solve this non-Euclidean black-box optimization problem. Our methodology is general and is suitable for any class or probabilistic models. In this work, however, we focused on the rich and common used class of Gaussian process models. We proposed a novel covariance function that captures the similarity between models. Empirically, we have demonstrated that modeling the model evidence (or marginal likelihood) with a GP in model space is capable of predicting the quality of unobserved models with enough fidelity to effectively explore model space via Bayesian optimization. Finally, we compared our approach with a previously proposed state-of-the-art technique and observed that our approach is sample-efficient. In a high level, we have shown that adapting Bayesian optimization to other non-Euclidean input spaces is promising. Several other domains could be investigated. The space of architectures for neural networks, for example, would be a natural next step.

Our last contribution is a novel automated active learning framework. In particular, we apply this approach to Bayesian optimization, creating an automated Bayesian optimization. Our algorithm dynamically builds a collection of models to explain the latent objective function. As more observation are gathered, our approach refines the collection of models using Bayesian optimization in the model space—similarly to our previous contribution. To efficiently perform model search in the active learning regime, we assume that each model evidence was corrupted by noise, the variance of which depends on the number of data points used to compute it. Our active model search procedure is completely general and could be applied to any active learning algorithm. Finally, empirical results show that the our method outperforms the baselines for several objective functions across multiple applications. In a future work, we could demonstrate the effectiveness of our automated active learning

to regression and classification tasks. Another interesting direction is to explore strategies for coupling our two-level Bayesian optimization loops. Instead of virtually running two independent Bayesian optimization algorithms, we could make the inner (model selection) Bayesian optimization search for relevant models for the outer task; as opposed to searching for all models that are relevant for explaining the current data well, as we have proposed. This would make the algorithm less general and more complicated (at least naively it would require extensive retraining!) but it could result in a more sample-efficient solution.

References

[1] ASM Alloy Center Database. URL `http://mio.asminternational.org/ac/`.

[2] A. Ali, R. Caruana, and A. Kapoor. Active Learning with Model Selection. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (*AAAI *2014)*, pages 1673–1679, 2014.

[3] P. Auer. Using Confidence Bounds for Exploitation–Exploration Trade-offs. *Journal of Machine Learning Research*, 3:397–422, 2002.

[4] F. R. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *Conference on Neural Information Processing Systems (*NIPS *2008)*, 2008.

[5] T. C. Bailey, Y. Chen, Y. Mao, C. Lu, G. Hackmann, S. T. Micek, K. M. Heard, K. M. Faulkner, and M. H. Kollef. A Trial of a Real-Time Alert for Clinical Deterioration in Patients Hospitalized on General Medical Wards. *Journal of Hospital Medicine*, 8(5): 236–242, 2013.

[6] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *Conference on Neural Information Processing Systems (*NIPS *2011)*. 2011.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

[8] E. Brochu, V. M. Cora, and N. De Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[9] R. Carhart and J. Jerger. Preferred Method for Clinical Determination of Pure-Tone Thresholds. *Journal of Speech and Hearing Disorders*, 24(4):330–345, 1959.

[10] D. R. Cavagnaro, J. I. Myung, M. A. Pitt, and J. V. Kujala. Adaptive design optimization: A mutual information-based approach to model discrimination in cognitive science. *Neural computation*, 22(4):887–905, 2010.

[11] M. Don, J. J. Eggermont, and D. E. Brackmann. Reconstruction of the audiogram using brain stem responses and high-pass noise masking. *Annals of Otology, Rhinology and Laryngology.*, 88(3 Part 2, Supplement 57):1–20, 1979.

[12] D. Duvenaud. *Automatic Model Construction with Gaussian Processes*. PhD thesis, Computational and Biological Learning Laboratory, University of Cambridge, 2014.

[13] D. Duvenaud, J. R. Lloyd, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Structure Discovery in Nonparametric Regression through Compositional Kernel Search. In *Proceedings of the 30th International Conference on Machine Learning (*ICML *2013)*, pages 1166–1174, 2013.

[14] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens. Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19:355–369, 2007.

[15] P. I. Frazier, W. B. Poweel, and S. Dayanik. A Knowledge-Gradient Policy for Sequential Information Collection. SIAM *Journal on Control and Optimization*, 47(5):2410–2439, 2008.

[16] J. R. Gardner, G. Malkomes, R. Garnett, K. Q. Weinberger, D. Barbour, and J. P. Cunningham. Bayesian active model selection with an application to automated audiometry. In *Conference on Neural Information Processing Systems (NIPS)*, pages 2386–2394, 2015.

[17] J. R. Gardner, X. Song, K. Q. Weinberger, D. Barbour, and J. P. Cunningham. Psychophysical Detection Testing with Bayesian Active Learning. In *Proceedings of the 31th Conference on Uncertainty in Artificial Intelligence (*UAI *2015)*, 2015.

[18] J. R. Gardner, C. Guo, K. Q. Weinberger, R. Garnett, and R. Grosse. Discovering and exploiting additive structure for Bayesian optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1311–1319, 2017.

[19] M. R. Garey. Optimal binary identification procedures. *SIAM Journal on Applied Mathematics*, 23(2):173–186, 1972.

[20] M. R. Garey and R. L. Graham. Performance bounds on the splitting algorithm for binary testing. *Acta Informatica*, 3(4):347–355, 1974.

[21] R. Garnett, M. A. Osborne, and S. J. Roberts. Bayesian Optimization for Sensor Set Selection. In *Proceedings of the 9th* ACM/IEEE *International Conference on Information Processing in Sensor Networks (*IPSN *2010)*, pages 209–219, 2010.

[22] R. Garnett, Y. Krishnamurthy, D. Wang, J. Schneider, and R. Mann. Bayesian optimal Active Search on Graphs. In *9th Workshop on Mining and Learning with Graphs*, 2011.

[23] R. Garnett, Y. Krishnamurthy, X. Xiong, J. G. Schneider, and R. P. Mann. Bayesian Optimal Active Search and Surveying. In *Proceedings of the 29th International Conference on Machine Learning (*ICML*)*, 2012.

[24] R. Garnett, M. A. Osborne, and P. Hennig. Active Learning of Linear Embeddings for Gaussian Processes. In *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (*UAI *2014*, pages 230–239, 2014.

[25] R. Garnett, T. Gärtner, M. Vogt, and J. Bajorath. Introducing the 'active search' method for iterative virtual screening. *Journal of Computer-Aided Molecular Design*, 29 (4):305–314, 2015.

[26] D. Geman and B. Jedynak. An active testing model for tracking roads in satellite images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(1):1–14, 1996.

[27] J. González, M. A. Osborne, and N. D. Lawrence. GLASSES: relieving the myopia of Bayesian optimisation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.

[28] R. Grosse, R. Salakhutdinov, W. Freeman, and J. Tenenbaum. Exploiting compositionality to explore a large space of model structures. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (*UAI *2014)*, 2012.

[29] Y. Guo and R. Greiner. Optimistic active-learning using mutual information. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (*IJCAI*)*, volume 7, pages 823–829, 2007.

[30] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[31] J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani. Predictive Entropy Search for Efficient Global Optimization of Black-box Functions. In *NIPS 2014*, pages 918–926, 2014.

[32] N. Houlsby, F. Huszar, Z. Ghahramani, and J. M. Hernández-Lobato. Collaborative Gaussian Processes for Preference Learning. In *NIPS*, pages 2096–2104, 2012.

[33] N. Houlsby, J. M. Hernández-Lobato, and Z. Ghahramani. Cold-start Active Learning with Robust Ordinal Matrix Factorization. In *Proceedings of the 31nd International Conference on Machine Learning (*ICML *2014)*, pages 766–774, 2014.

[34] D. Huang, T. T. Allen, W. I. Notz, and N. Zeng. Global optimization of stochastic black-box systems via sequential kriging meta-models. *Journal of Global optimization*, 34:441–466, 2006.

[35] D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.

[36] W. Hughson and H. Westlake. Manual for program outline for rehabilitation of aural casualties both military and civilian. *Transactions of the American Academy of Ophthalmology and Otolaryngology*, 48(Supplement):1–15, 1944.

[37] S. Jasial, Y. Hu, M. Vogt, and J. Bajorath. Activity-relevant similarity values for fingerprints and implications for similarity searching. *F1000Research*, 5(Chem Inf Sci): 591, 2016.

[38] B. Jedynak, P. I. Frazier, and R. Sznitman. Twenty questions with noise: Bayes optimal policies for entropy loss. *Journal of Applied Probability*, 49(1):114–136, 2012.

[39] S. Jiang, G. Malkomes, M. Abbott, B. Moseley, and R. Garnett. Efficient nonmyopic batch active search. In *Advances in Neural Information Processing Systems 31 (*NIPS*)*, pages 1099–1109. 2018.

[40] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

[41] Y. Kawazoe, J.-Z. Yu, A.-P. Tsai, and T. Masumoto, editors. *Nonequilibrium Phase Diagrams of Ternary Amorphous Alloys*, volume 37A of *Condensed Matter*. Springer–Verlag, 1997.

[42] I. Kononenko. Machine Learning for Medical Diagnosis: History, State of the Art and Perspective. *Artificial Intelligence in Medicine*, 23(1):89–109, 2001.

[43] J. Kuha. AIC and BIC: Comparisons of Assumptions and Performance. *Sociological Methods and Research*, 33(2):188–229, 2004.

[44] J. Kulick, R. Lieck, and M. Toussaint. Active Learning of Hyperparameters: An Expected Cross Entropy Criterion for Active Model Selection. *CoRR*, abs/1409.7552, 2014.

[45] H. J. Kushner. A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86:97–106, 1964.

[46] M. Kuss and C. E. Rasmussen. Assessing Approximate Inference for Binary Gaussian Process Classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005.

[47] D. D. Lewis and W. A. Gale. A Sequential Algorithm for Training Text Classifiers. In *Proceedings of the 17th Annual International* ACM SIGIR *Conference on Research and Development in Information Retrieval*, pages 3–12, 1994.

[48] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018. URL `http://jmlr.org/papers/v18/16-558.html`.

[49] C. K. Ling, K. H. Low, and P. Jaillet. Gaussian process planning with Lipschitz continuous reward functions: Towards unifying Bayesian optimization, active learning, and beyond. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 1860–1866, 2016.

[50] T. Liu, Y. Lin, X. Wen, R. N. Jorissen, and M. K. Gilson. BindingDB: A web-accessible database of experimentally determined protein–ligand binding affinities. *Nucleic Acids Research*, 35(suppl 1):D198–D201, 2007.

[51] J. R. Lloyd, D. Duvenaud, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Automatic Construction and Natural-Language Description of Nonparametric Regression Models. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI 2014)*, pages 1242–1250, 2014.

[52] D. W. Loveland. Performance bounds for binary testing with arbitrary weights. *Acta Informatica*, 22(1):101–114, 1985.

[53] Y. Ma, R. Garnett, and J. Schneider. Active Area Search via Bayesian Quadrature. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, number 33 in Proceedings of Machine Learning Research, pages 595–603, 2014.

[54] Y. Ma, T. Huang, and J. Schneider. Active Search and Bandits on Graphs using Sigma-Optimality. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*, pages 542–551, 2015.

[55] Y. Ma, D. J. Sutherland, R. Garnett, and J. Schneider. Active Pointillistic Pattern Search. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, number 38 in Proceedings of Machine Learning Research, pages 672–680, 2015.

[56] D. J. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[57] D. J. C. Mackay. *Bayesian Methods for Adaptive Models*. PhD thesis, Pasadena, CA, USA, 1992. UMI Order No. GAX92-32200.

[58] G. Malkomes, C. Schaff, and R. Garnett. Bayesian optimization for automated model selection. In *Conference on Neural Information Processing Systems (NIPS)*, 2016.

[59] D. McBride and S. Williams. Audiometric notch as a sign of noise induced hearing loss. *Occupational Environmental Medicine*, 58(1):46–51, 2001.

[60] T. P. Minka. Expectation Propagation for Approximate Bayesian Inference. In *UAI*, pages 362–369, 2001.

[61] T. Mitchell. *Machine Learning*. McGraw-Hill International Editions. McGraw-Hill, 1997. ISBN 9780071154673. URL `https://books.google.com/books?id=EoYBngEACAAJ`.

[62] J. Močkus. *On Bayesian methods for seeking the extremum*, pages 400–404. Springer, 1974.

[63] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[64] D. I. Nelson, R. Y. Nelson, M. Concha-Barrientos, and M. Fingerhut. The global burden of occupational noise-induced hearing loss. *American Journal of Industrial Medicine*, 48 (6):446–458, 2005.

[65] M. A. Osborne, D. Duvenaud, R. Garnett, C. E. Rasmussen, S. J. Roberts, and Z. Ghahramani. Active learning of model evidence using Bayesian quadrature. In *Conference on Neural Information Processing Systems (NIPS)*, 2012.

[66] J. J. Pfeiffer III, J. Neville, and P. N. Bennett. Active Exploration in Networks: Using Probabilistic Relationships for Learning and Inference. In *Proceedings of the 23rd* ACM *International Conference on Information and Knowledge Management*, pages 639–648, 2014.

[67] A. E. Raftery. Approximate Bayes Factors and Accounting for Model Uncertainty in Generalised Linear Models. *Biometrika*, 83(2):251–266, 1996.

[68] C. E. Rasmussen and H. Nickisch. Documentation for GPML Matlab Code. `http://www.gaussianprocess.org/gpml/code/matlab/doc/`, 2015.

[69] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[70] R. Sabbadin, J. Lang, and N. Ravoanjanahary. Purely Epistemic Markov Decision Processes. In *Proceedings of the 22nd* AAAI *Conference on Artificial Intelligence*, pages 1057–1062, 2007.

[71] M. Sandelius. On an optimal search procedure. 1961.

[72] S. Saria, A. K. Rajani, J. Gould, D. L. Koller, and A. A. Penn. Integration of Early Physiological Responses Predicts Later Illness Severity in Preterm Infants. *Science Translational Medicine*, 2(48):48ra65, 2010.

[73] G. Schwarz. Estimating the Dimension of a Model. *Annals of Statistics*, 6(2):461–464, 1978.

[74] B. Settles. *Active Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool, 2012.

[75] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *COLT 92*, pages 287–294, New York, NY, USA, 1992. ACM. ISBN 0-89791-497-X.

[76] Shaltyanis. One method of multiextremum optimization. *Automatic Control and Computer Sciences*, 5(3):33–38, 1971.

[77] J. Shargorodsky, S. G. Curhan, G. C. Curhan, and R. Eavey. Change in Prevalence of Hearing Loss in US Adolescents. *Journal of the American Medical Association*, 304(7): 772–778, 2010.

[78] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in Neural Information Processing Systems 25 (*NIPS *2012)*, pages 2951–2959, 2012.

[79] N. Srinivas, A. Krause, S. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning (ICML)*, 2010.

[80] N. Srinivas, A. Krause, S. Kakade, and M. W. Seeger. Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1015–1022, 2010.

[81] T. Sterling and J. J. Irwin. ZINC 15 – Ligand Discovery for Everyone. *Journal of Chemical Information and Modeling*, 55(11):2324–2337, 2015.

[82] S. Surjanovic and D. Bingham. Optimization test functions and datasets, 2017. URL `http://www.sfu.ca/~ssurjano/optimization.html`.

[83] R. Sznitman, A. Lucchi, P. I. Frazier, B. M. Jedynak, and P. Fua. An optimal policy for target localization with application to electron microscopy. In *International Conference on Machine Learning (*ICML*)*, number CONF, 2013.

[84] T. Tsiligkaridis, B. M. Sadler, and A. O. Hero. A collaborative 20 questions model for target search with human-machine interaction. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6516–6520. IEEE, 2013.

[85] H. P. Vanchinathan, A. Marfurt, C.-A. Robelin, D. Kossmann, and A. Krause. Discovering Valuable Items from Massive Data. In *Proceedings of the 21st* ACM SIGKDD *International Conference on Knowledge Discovery and Data Mining*, pages 1195–1204, 2015.

[86] K. Wagstaff. Machine Learning that Matters. In *Proceedings of the 29th International Conference on Machine Learning (ICML 2012)*, 2012.

[87] X. Wang, R. Garnett, and J. Schneider. Active Search on Graphs. In *Proceedings of the 19th* ACM SIGKDD *International Conference on Knowledge Discovery and Data Mining*, pages 731–738, 2013.

[88] L. Ward, A. Agrawal, A. Choudhary, and C. Wolverton. A general-purpose machine learning framework for predicting properties of inorganic materials. *npj Computational Materials*, 2:16028, 2016.

[89] C. Williams and C. Rasmussen. Gaussian processes for regression. In *NIPS*, 1996.

[90] A. G. Wilson, E. Gilboa, A. Nehorai, and J. P. Cunningham. Fast Kernel Learning for Multidimensional Pattern Extrapolation. In *NIPS*, pages 3626–3634, 2014.

[91] Y. Yang and M. Loog. Active learning using uncertainty information. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2646–2651. IEEE, 2016.

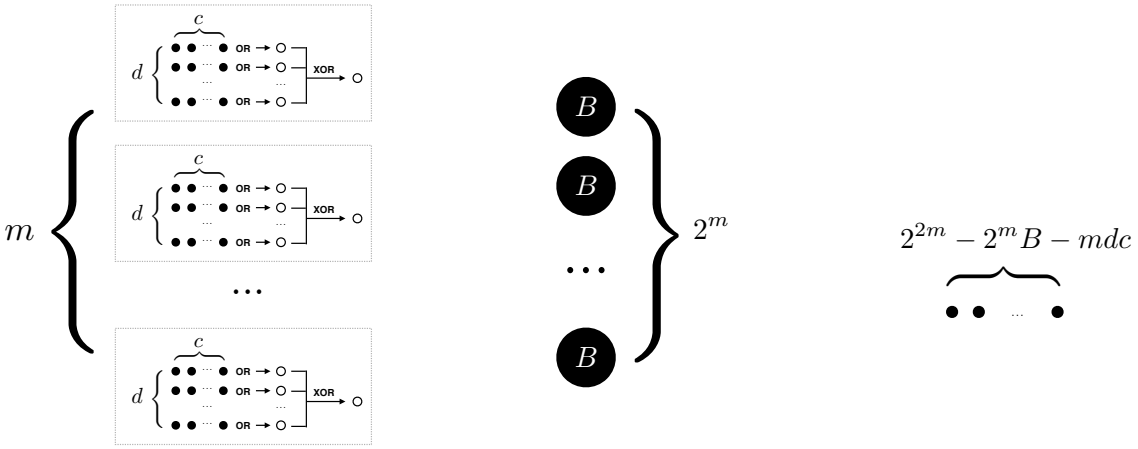
Appendix A

Hardness of active search

We present the proof of Theorem 1. We assume that active search policies have access to the correct marginal probabilities $f(x; \mathcal{D}) = \Pr(y = 1 \mid x, \mathcal{D})$, for any given point x and labeled data $\mathcal{D}$, which may include "ficticious" observations. Further, the computational cost will be analyzed as the number of calls to f, i.e., $f(x; D)$ has unit cost. Note that the optimal policy operates in such a computational model, with exponentially many calls (in terms of $|\mathcal{X}|$) to the marginal probability function f.

Our proof technique consists of constructing an explicit active search instance where a small "secret" set of points S encodes the location of a larger "hidden" group of positive points. A particular feasibly exponential-cost policy identifies this small set first, and then obtains a large reward by collecting the revealed targets. We will show that an algorithm with limited computational power (i.e., polynomial in $n = |\mathcal{X}|$) will not be able to identify the set S. As a consequence, its performance will be arbitrarily worse than an optimal solution as the size of the instance increases.

The crux of the proof is to construct a class of instances $\mathcal{H}$ that we present next. Figure A.1 shows a schematic representation of an example instance $\mathcal{I} \in \mathcal{H}$. The instances in $\mathcal{H}$



(a) Secret set S of mdc isolated points.

(b) $2^m B$ points from clumps C_j.

(c) Isolated and independent points R.

Figure A.1: An instance of active search where any efficient algorithm can be arbitrarily worse than an optimal policy.

differ from each other by a permutation of the labels. An instance has $n = |\mathcal{X}| = 2^{2m}$ points, where m is a parameter of the instance. The search budget is defined to be $B = m^2$. The points in each instance can be categorized as follows.

"Clumps." These points are partitioned into 2^m groups, which we will call "clumps," each of size B. All points in a clump share the same label. Additionally, exactly one of the clumps comprises all positive points; the remaining points are all negative. The clump containing the positive points is chosen uniformly at random; therefore, the prior marginal probability for any point x_c in this category is $f(x_c; \emptyset) = p_c = 2^{-m}$. We denote the clump of all positive points C_*, where $*$ can be regarded as a m-bit integral index $1 \leq * \leq 2^m$. Figure A.1(b) illustrates these points.

"Isolated points." The remaining points share the property that observing any *single* one of them does not change the marginal probabilities of any other point. These points are illustrated in Figures A.1(a) (black dots) and A.1(c). The marginal probabilities for any point

x_b in this category is defined to be $f(x_b; \emptyset) = p_b = 1 - \sqrt[c]{1/2}$, where we define $c = \sqrt{m}/2$ for convenience. These points can be further classified into two categories:

- A "secret set," denoted by S; see Figure A.1(a) (black dots). These points encode which of the clumps C_* contains the positive points, using a scheme we describe below. For ease of exposition, we partition the set S into m subsets $S_1, \ldots, S_m$, each of size dc, where we define $d = \sqrt{m}$. Thus $|S| = mdc = {}^{m^2}/_2 = {}^{B}/_2$; the size of this secret set is exactly half the budget.

  The key of this construction is that each subset S_i encodes one bit b_i of information about which clump C_* contains the positive points, using a simple encoding scheme: the binary representation of the positive clump C_*'s index is $* = b_1 b_2 \ldots b_m$. Each bit is encoded with a two-step mechanism. First, each S_i is partitioned into d groups of c points, each group encoding a "meta" bit of information $b_{ij}, 1 \leq i \leq m, 1 \leq j \leq d$, by a logical OR. These meta-bits, not in the problem instance, are illustrated by the white dots in A.1(a). Finally, the meta-bits associated with S_i encode the bit b_i via a logical XOR,[16] $b_i = b_{i1} \oplus \cdots \oplus b_{id}$.

- Independent points. The remaining $2^{2m} - 2^m B - mdc$ points are totally independent from the others; revealing them conveys no information for any other point. We denote this set of points by R.

Observation 1. *At least d points from S_i need to be observed in order to infer one bit b_i of information about the positive clump.*

A virtual bit b_i is computed by the XOR operation of the d associated meta-bits $b_i = b_{i1} \oplus \cdots \oplus b_{id}$. Notice that each b_{ij} has same marginal probability of being positive, i.e.,

[16] Sum of the bits modulo 2.

$\forall i, j, \Pr(b_{ij} = 1) = (1 - p_b)^c = 1/2$. We also have $\forall i, \Pr(b_i = 1) = 1/2$. It is necessary to observe all d meta-bits b_{ij} from the same group S_i to infer the bit b_i, since observing a fraction of the inputs of an XOR does not change the marginal belief about its outcome. So observing $d - 1$ or fewer points from S conveys no information about the positive clump. Equivalently, $\forall x, \Pr(y = 1 \mid x, \mathcal{D}) = \Pr(y = 1 \mid x)$ if $|\mathcal{D} \cap S| \leq d - 1$.

Observation 2. *Observing any number of clump points does not change the marginal probability of any point in the secret set S.*

We need to make sure that no external information can help to identify the secret set S. Notice that the knowledge of b_i does not change the marginal probability of any b_{ij}; hence, no point x in S will have a different probability after observing b_i. This means that observing points outside S does not help distinguish S from the remaining isolated points R.

Now we formally restate Theorem 1 and provide its proof. The theorem implies that a polynomial time algorithm cannot achieve a constant approximation ratio.

Theorem 2. *Any (possibly randomized) policy for active search that performs* $o\left(n^{\sqrt{\frac{1}{2}\log n}}\right)$ *inference calls* $f(x, \mathcal{D})$*, with* $|\mathcal{D}| \leq B$*, has approximation ratio with respect to the expected utility of the optimal policy of* $\mathcal{O}\left(\frac{1}{\sqrt{\log n}}\right)$ *where* $|\mathcal{X}| = n$ *is the number of points, and* B *is the budget.*[17]

Proof. Consider a random instance $\mathcal{I} \in \mathcal{H}$ and fix a policy $\mathcal{A}$. Let α be the total number of inference calls performed by $\mathcal{A}$ throughout its execution. At the ith inference call $\Pr(y = 1 \mid x, \mathcal{D}_i)$, $\mathcal{A}$ might use an arbitrary training set $\mathcal{D}_i$ of size at most B. We will show that $\mathcal{A}$ has a very small probability of collecting a large reward on $\mathcal{I}$.

Before analyzing the algorithm $\mathcal{A}$, we present a lower bound on the performance of an optimal policy. Consider the following policy with unlimited computational power: In the first iteration, compute the marginal probability of an arbitrary fixed clump point, conditioning on observing every possible subset of the isolated points of size d with labels all equal to 1. This set of $\mathcal{O}(n^d) = \mathcal{O}(n^{\sqrt{\frac{1}{2}\log n}})$ inference calls will reveal the location of the secret set S: exactly those points will modify the probabilities of the fixed clump point. Now the policy spends the first half of its budget querying the points in S (recall $|S| = B/2$). These outcomes reveal the hidden clump of positives C_*. The policy now spends the second half of the budget querying (collecting) these positive points. The expected performance of this strategy is $B/2 + p_b \, B/2 > B/2$. Certainly this is a lower bound on the optimal performance; hence

$$\text{OPT} > \frac{B}{2}. \tag{A.1}$$

[17]Note we used the little-o notation for the number of inference calls, and the big-O notation of the approximation ratio.

Now consider the algorithm $\mathcal{A}$ at the ith inference. By Observations 1 and 2, $\mathcal{A}$ cannot differentiate between the points in S and those in R unless $|\mathcal{D}_i \cap S| \geq d$. Suppose that before the ith inference, the algorithm has no information about S. Then the chance of $\mathcal{A}$ choosing a $\mathcal{D}_i$ such that $|\mathcal{D}_i \cap S| \geq d$ is no better than that of a random selection from n' points, where $n' = n - 2^m B$ is the number of isolated points. We can upper bound the probability of $\mathcal{A}$ choosing a dataset $\mathcal{D}_i$ such that $|\mathcal{D}_i \cap S| \geq d$, by counting how many subsets would contain at least d points from S, among all subsets of the n' points of size at most B:

$$\Pr\big(|\mathcal{D}_i \cap S| \geq d\big) \leq \frac{\binom{B/2}{d}\binom{n'-d}{B-d}}{\binom{n'}{B}}. \tag{A.2}$$

We only consider the isolated points because an algorithm $\mathcal{A}$ that only queries the isolated points has a higher probability of hitting the secret set. Also note that technically the denomenator in (A.2) should be $\binom{n'}{B} - i + 1$ since one would not choose the same subsets $\mathcal{D}_j, j < i$ as those before the ith inference. But asymptotically $i \leq \alpha$ (assuming $\alpha = \mathcal{O}(2^n)$ for now) is of much lower order than $\binom{n'}{B}$, therefore $\binom{n'}{B} - \alpha + 1 = \Theta\Big(\binom{n'}{B}\Big)$. Denote p_h as the probability of algorithm $\mathcal{A}$ ever "hiting" the secret set after α inferences; then p_h can be union-bounded:

$$\begin{aligned} p_h &\leq \frac{\alpha\binom{B/2}{d}\binom{n'-d}{B-d}}{\binom{n'}{B}} \\ &< \frac{\alpha\left(\frac{B}{2}\right)^d B^d}{(n')^d} \\ &= \frac{\alpha}{\left(\frac{2n'}{B^2}\right)^d} \end{aligned}$$

Note $B = m^2 = \frac{1}{4}\log^2 n$ and $n' = n - 2^m B = n - \sqrt{n}\frac{1}{4}\log^2 n = \Theta(n)$, so

$$\begin{aligned}\left(\frac{2n'}{B^2}\right)^d &= \Theta\left(\left(\frac{2n}{B^2}\right)^d\right) \\ &= \Theta\left(\left(\frac{n}{\frac{1}{32}\log^4 n}\right)^{\sqrt{\frac{1}{2}\log n}}\right) \\ &= \Theta\left(n^{\sqrt{\frac{1}{2}\log n}}\right).\end{aligned}$$

We have now derived

$$p_h < \frac{\alpha}{\Theta(n^{\sqrt{\frac{1}{2}\log n}})}.$$

So for any $\alpha = \mathcal{O}\big(n^{\sqrt{\frac{1}{2}\log n}-\varepsilon}\big)$, where ε is a positive constant, we have

$$p_h < \mathcal{O}\left(\frac{1}{n^{\varepsilon}}\right). \tag{A.3}$$

If $\mathcal{A}$ ever hits the secret set S, for simplicity, we will assume that it achieves maximal performance B. If $\mathcal{A}$ never finds the secret set, we can further consider the following two cases. If the algorithm queries points $x \in R$, no marginal probabilities are changed; if a point $x \in C_j$ is queried, for any clump j, only the marginal probabilities of the clumps are changed. The expected performance in these two cases can be upper bounded by pretending that the algorithm had a larger budget of size $2B$; in which half the budget (i.e., B) is spent on querying points in R, and the other half on querying the clump points. The expected number of targets found after B queries on R is

$$Bp_b = B(1 - \sqrt[c]{1/2}) = B\left(1 - 2^{-\frac{2}{\sqrt{m}}}\right). \tag{A.4}$$

The expected number of targets found after B queries on the clump points is

$$\begin{aligned}\frac{B}{2^m} + \left(1 - \frac{1}{2^m}\right)\left(\frac{B-1}{2^m - 1} + \left(1 - \frac{1}{2^m - 1}\right)(\cdots)\right) \\ = \frac{B(B+1)}{2^{m+1}}.\end{aligned} \tag{A.5}$$

Combining (A.4) and (A.5), we get that the expected performance in the case when $\mathcal{A}$ never hits S can be upper bounded by

$$\frac{B(B+1)}{2^{m+1}} + B\big(1 - 2^{-\frac{2}{\sqrt{m}}}\big).$$

The overall expected performance of $\mathcal{A}$ can be upper bounded by

$$E_{\mathcal{A}} < Bp_h + \frac{B(B+1)}{2^{m+1}} + B\big(1 - 2^{-\frac{2}{\sqrt{m}}}\big),$$

where we have used the trivial upper bound $1 > (1 - p_h)$. Finally, combining with the lower bound of OPT in (A.1), the approximation ratio can be upper bounded by

$$\begin{aligned}\frac{E_{\mathcal{A}}}{\text{OPT}} &< \frac{Bp_h + \frac{B(B+1)}{2^{m+1}} + B\big(1 - 2^{-\frac{2}{\sqrt{m}}}\big)}{B/2} \\ &= 2p_h + \frac{B+1}{2^m} + 2\big(1 - 2^{-\frac{2}{\sqrt{m}}}\big) \\ &= \mathcal{O}\left(\frac{1}{n^\varepsilon} + \frac{\log^2 n}{4\sqrt{n}} + 2\Big(1 - 2^{-\frac{2\sqrt{2}}{\sqrt{\log n}}}\Big)\right) \\ &= \mathcal{O}\left(\frac{1}{\sqrt{\log n}}\right).\end{aligned}$$

for any $\alpha = \mathcal{O}\big(n^{\sqrt{\frac{1}{2}\log n} - \varepsilon}\big) = o\big(n^{\sqrt{\frac{1}{2}\log n}}\big)$. Note that it is easy to verify that $2\big(1 - 2^{-\frac{2\sqrt{2}}{\sqrt{\log n}}}\big) = \Theta\left(\frac{1}{\sqrt{\log n}}\right)$ with L'Hôpital's rule. □