FEEDBACK CODING FOR EFFICIENT INTERACTIVE MACHINE LEARNING

A Dissertation
Presented to
The Academic Faculty

By

Gregory Humberto Canal

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

August 2021

FEEDBACK CODING FOR EFFICIENT INTERACTIVE MACHINE LEARNING

Thesis committee:

Dr. Christopher Rozell
Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Yao Xie
Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Mark Davenport
Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Robert Nowak
Electrical and Computer Engineering
University of Wisconsin-Madison

Dr. Matthieu Bloch
Electrical and Computer Engineering
Georgia Institute of Technology

Date approved: June 21, 2021

We know the past but cannot control it. We control the future but cannot know it.

Claude Shannon

For my family, who instilled in me the value of curiosity, learning, and creating, and without whom this work would not be possible.

ACKNOWLEDGMENTS

The greatest part about graduate school has been the opportunity to meet so many extraordinary people along the way. First, I would like to thank my advisor Chris Rozell for his mentorship and guidance throughout graduate school. It has been an incredible experience working with you, and I am extremely grateful for the breadth and depth of an education you have given me both inside and outside the lab, including how to improve as a communicator and storyteller and helping me grow as an independent researcher and person. I would like to thank Mark Davenport for his mentorship, our collaborations, and for serving on my thesis committee; I am very lucky to have had Mark as another amazing mentor throughout graduate school. In the latter part of my PhD, I had the wonderful opportunity to connect and collaborate with Matthieu Bloch, who has shared with me many fascinating discussions and insights about information theory, research, and beyond, and who I am grateful to have on my committee. I would also like to thank Professor Yao Xie and Professor Rob Nowak, who have kindly lent their time and expertise to serve on my committee and provide feedback on my work.

I am incredibly lucky to have had the unique experience of being a member of the "Children of the Norm" cohort, which has been an engaging and lively space for collaborations, feedback, and discussions throughout graduate school. Chris, Mark, Justin Romberg and Eva Dyer led by example in showing each of us students how to not only do solid technical research, but also how to communicate clearly and tell a story in our research and careers, and how to enjoy the best $3 IPA in town every Friday. I would like to especially thank Justin for the fantastic core technical instruction you have given me in my PhD — your courses in digital signal processing, statistical learning, and convex optimization provided an invaluable foundation for my technical graduate education. As part of this cohort, I have had the opportunity to make collaborations and friendships outside of my immediate lab group, and thank (to name a few of the many) Andrew, Andy, Cole, Jihui, Kyle, Liangbei,

Michael, Namrata, Nauman, Ning, Rakshith, Santhosh, Sihan, Sohail, Thinh, and Tomer for their friendship and support.

I am grateful for the collaborations, feedback, friendships, and endless support from all of the awesome members (both past and present) of the SIP Lab that I have had the pleasure to meet and work with: Adam C., Adam W., Ayse, Belén, John, Kion, Kyle, Marissa, Matt, Nick, Pavel, Sankar, Shaoheng, Sippy, Siva, and Stefano; it has been a privilege to be a part of this amazing group. I have learned so much from all of you, and graduate school would not have been the same without you. I would like to acknowledge Marissa in particular, who welcomed me to Georgia Tech as my ECE student mentor and helped me navigate my first year of graduate school.

The Georgia Tech ECE department as a whole is full of amazing people who I could not have earned this degree without. Thank you to all of my professors, including Professor Fekri who captivated me with his course on information theory, and to the ECE administrators including Angel Greenwood, Dr. Daniela Staiculescu, Pat Dixon, Raquel Plaskett, and Tasha Torrence, who have been incredibly kind and helpful throughout my PhD.

Finally, I would like to thank my friends and family in general. Your support, love, patience, and laughter has been more helpful than I could ever express in words, and I am extremely lucky and thankful to have you in my life. To my parents, Humberto and Maureen, my sister Aly and her husband Mike, and my Zeena, your love means everything to me and I could not have come this far without you. My courageous grandma Ellen left her homeland of Ireland for the U.S. when she was only 21 years old, with dreams of building a better life for herself and raising a family with education as a top priority, which she never had the opportunity to pursue herself. She and my Pop Pop, also an immigrant from Ireland, raised my mom with this value in mind. My Mimi and Grandpa Canal similarly raised my dad to always be curious, work hard, and love and support those around him. I owe everything I have to my parents and family that came before me, and am forever grateful for their love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

When training supervised computational systems that learn from data, the most basic scenario consists of the learning algorithm operating on a fixed batch of data, provided in its entirety before training. However, there are a large number of applications ranging from adaptive robotic sensing to human-in-the-loop drug discovery where there lies a *choice* in which data points are selected for labeling, and where this choice can be made "on the fly" after each selected data point is labeled. In such interactive machine learning (IML) scenarios, the quality of interactions with the information source providing data labels has a tremendous impact on the performance of the resulting system. At any point in time during training, certain data points are more informative to label than others due to label redundancy (similar data points provide superfluous labels) or different levels of noise (noisy labels do not improve the learned model). By only labeling informative data points, it is possible for systems trained in an interactive paradigm to learn a generalizable model with far fewer labels than would be required otherwise. This reduced demand for data labels is important in any setting where labels are expensive or time-consuming to acquire, such as when soliciting the knowledge of a human expert.

To measure the informativeness of labeling any particular data point, it is common practice to apply statistical tools from the field of information theory. However, using these tools to directly estimate query informativeness over a large number of possible data points can be computationally expensive, and in many applications such as real-time human-computer interfacing, queries need to be selected with minimal computational overhead. More fundamentally, as we explore in this work there sometimes exist convenient query structures in IML that allow for computational and algorithmic advantages not capitalized on by brute-force information maximization. In this thesis, we identify and model query structures in IML to develop direct information maximization solutions as well as approximations that allow for computationally efficient query selection. A major theme

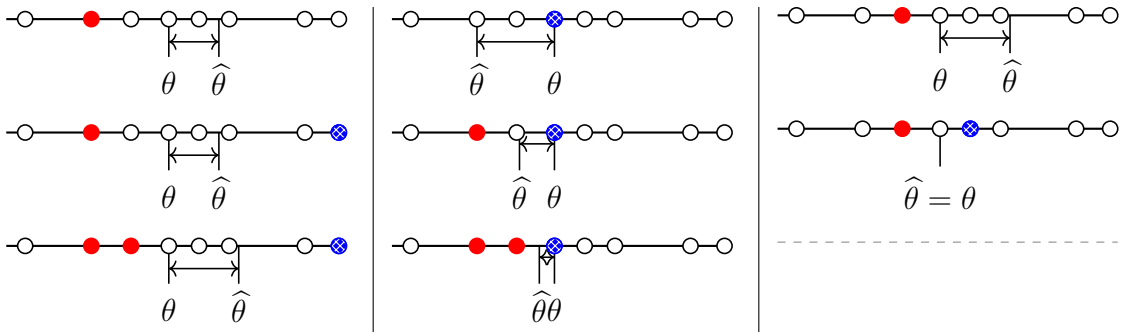
of this work is the utilization of tools and concepts from feedback coding theory, which have only seen limited application to IML.

Specifically, we frame IML as a feedback communications problem and directly apply principles and algorithms from coding theory to design and analyze high-performing IML systems that efficiently utilize interactions between the labeling expert and learner. As a step towards integrating concepts from coding theory into IML, we directly apply a recently developed feedback coding scheme to sequential human-computer interaction systems including control of a robot swarm with a brain-computer interface [1] and interactive object segmentation [2]. We then identify simplifying query structures to develop approximate methods for efficient, informative query selection in interactive ordinal embedding construction [3] and preference learning systems [4, 5]. Finally, we combine the direct application of feedback coding with approximate information maximization to design and analyze a general active learning algorithm, which we study in detail for logistic regression [6].

CHAPTER 1
INTRODUCTION

Recent improvements in computational power, increased availability of large datasets, and algorithmic advances have led to a surge in performance of machine learning systems across a wide variety of settings. For instance, the ImageNet dataset contains over 14 million labeled images across over 21,000 categories [7], and state-of-the-art methods have been able to successfully classify this data with high accuracy [8]. However, is it imperative to not only investigate how machine learning algorithms can be scaled to match larger datasets, but also how one can be more judicious in the selection of training data itself. It is crucial to select data intelligently for labeling in any learning scenario where data labels are expensive, such as in some healthcare applications where a medical expert can only provide a small set of labeled examples [9] or when selecting experiments for drug discovery from a combinatorially large set of chemical possibilities [10, 11]. Even in settings rich with labeled data, one may still wish to select data intelligently to reduce training memory and compute time costs [12]. In cases such as these, it is advantageous to have a labeling expert "in the loop" during training, where the expert and machine learner iteratively collaborate to select data points for training that maximize learning efficiency. In this thesis, we broadly refer to this type of learning as *interactive machine learning* (IML) [13].

As an example of the sample complexity benefits gained from having an expert in-the-loop for data selection, consider the illustrative example in Figure 1.1 of a one-dimensional threshold classifier (adapted from [14]). In this example, an expert assigns a label $y \in \{0, 1\}$ to each requested data point $x \in \mathbb{R}$, each depicted as a circle. Labels are indicated by circle texture, with solid, red $(y = 1)$ or hatched, blue $(y = 0)$ circles for labeled points, and unfilled circles for unlabeled points. A learner predicts data labels with the threshold function $h_\theta(x) = \mathbb{1}_{x \leq \theta}(x)$, parameterized by a threshold $\theta \in \mathbb{R}$.



(a) Passive learning (b) Active learning (c) Machine teaching

Figure 1.1: One-dimensional threshold classification for different example selection strategies, depicted vertically over three sequentially labeled examples for each method.

Suppose that an optimal (or "ground truth") threshold θ exists that perfectly separates the two data classes, and that the learner estimates this threshold with a maximum margin classifier parameterized by $\widehat{\theta}$, which is computed as the bisecting value of the two innermost data points with disagreeing labels. If data points are selected *passively* for labeling as in Figure 1.1a by sampling uniformly at random, then obtaining an absolute deviation $|\widehat{\theta} - \theta|$ less than Δ for some $\Delta > 0$ requires on the order of $O(\Delta^{-1})$ labeled examples. If the learner instead adopts an *active learning* strategy as in Figure 1.1b by intelligently requesting data points for labeling, it only requires $O(\log \Delta^{-1})$ labeled examples to achieve the same degree of error. This can be achieved by employing a bisection search strategy, where at each iteration the learner requests labels for examples that bisect the feasible space of threshold values. However, if the expert adopts a *machine teaching* strategy as in Figure 1.1c by utilizing their knowledge of the ideal separator value θ to both select and label examples, the number of required samples can be drastically reduced. In this case, the expert utilizes the fact that the learner uses a maximum margin classifier: since the learner will estimate $\widehat{\theta}$ as the bisector between the two innermost examples that differ in label, the expert can simply label two examples that are bisected by θ. This insight allows the expert to specify θ with $O(1)$ labeled examples for any arbitrary $\Delta > 0$, demonstrating the power of putting the expert in-the-loop to gain sample complexity benefits in machine learning.

The key factors that differentiate between these strategies are to what degree the example

selection policy can leverage the history of labeled examples (as in active learning) or the ground truth threshold known by the expert (as in machine teaching), as well as differences in the information pathways between the expert and the learner. In passive learning, information about the ground truth threshold is passed to the learner through data labels rather than example locations, since the locations are simply selected at random without regard to the ground truth threshold. In active learning, although the learner leverages the labeling history to select each example location such that assigned labels are as informative as possible, the locations themselves do not directly encode information about the ground truth threshold. By contrast, in machine teaching the example locations themselves directly encode the ground truth threshold value (assuming that these points have opposite labels). In other words, the differences in sample complexity between each example selection policy are related to differences in what type and how much information each labeled example encodes, along with what information is available to the example selection policy, which serves as an *encoder* of the ground truth decision parameters.

Thinking about labeled examples as encoding ground truth parameters is reminiscent of Shannon's mathematical theory of communication, which showed the limitations for how efficiently a telecommunications message could be encoded and communicated with a set of symbols, and mathematically guaranteed the existence of coding algorithms that could achieve this optimal performance [15]. These efforts evolved into the fields of information and coding theory, which not only had the benefit of delineating clear design benchmarks for telecommunications systems in the seventy years that followed Shannon's work, but have also provided a suite of statistical tools that have had profound impacts on fields ranging from psychology [16] to machine learning [17, 18, 19]. Viewing the IML task of judiciously selecting training examples to maximize a machine's learning rate as a coding problem, we can leverage this same set of tools to measure and maximize example information.

As we introduce in Chapter 2, the key quantity in applying information theory to example selection is the information gain provided by any particular example, which is

the mutual information between an example's label and the underlying model parameters being learned[1]. While information gain (and similarly mutual information) has several complementary interpretations, roughly speaking it measures the reduction in uncertainty that labeling an example provides about the ground truth model parameters. While the direct estimation and subsequent maximization of information gain is already a popular strategy for active data selection [20], there are several challenges involved in its estimation[2]. In particular, it is not always the case that information gain can be evaluated analytically, and so samples typically need to be drawn from a probability distribution over the possible model parameters, which are then used in a Monte Carlo estimate of information gain. This approach adds algorithmic complexity and can be computationally expensive, since the accuracy of the information gain estimate scales directly with the number of samples drawn, and generating samples from the parameter distribution may be difficult. Any increase in the computational cost of estimating information gain per candidate example can be significant, since typically a large pool of unlabeled examples is considered for labeling and, as we explore in Chapter 5, for more complex interaction types the number of candidate examples to consider can scale combinatorially with the pool size. It is particularly important for data selection compute times to be kept low in applications involving humans-in-the-loop in order to minimize real-time interaction latency, and is especially important when there is a direct cost associated with a human expert's time spent waiting between interactions.

More fundamentally, the brute-force approach of estimating and maximizing information gain does not fully capitalize on query structures inherent to many IML problems. In this thesis, we fully identify and model these query structures and show in several cases how they lead to either direct information maximization solutions that do not require explicit information gain estimation, or alternate example selection policies that approximate the

[1] In this work, we take a Bayesian approach and assume the existence of a prior distribution over "optimal" or "ground truth" model parameters that generate labels according to the same model class as the learner.

[2] The computational challenges we discuss here are present even when analytical expressions are available for probability densities, and are distinct from the statistical and computational difficulties involved in estimating mutual information from empirical samples (see for example [21]).

action of information maximization while having a cheaper computational cost along with other algorithmic advantages. Specifically, by formalizing the notion of labeled examples as "encoding" the underlying model parameters to be learned, we model interactive learning from first principles as a noisy communications system with feedback, and utilize theoretical and algorithmic tools from feedback channel coding and information theory to analyze and develop example selection policies for efficient IML. We use this model to both deploy existing feedback coding schemes for the design of example selection policies that maximize information gain by construction, and motivate computationally efficient approximations to information maximization. In general, such feedback coding schemes and models have previously only had limited application to designing example selection policies in interactive learning, and we anticipate that more formally bridging feedback coding theory with interactive learning will open up new avenues for efficient example selection and IML analysis.

We approach our coding-theoretic design of IML interaction policies through a sequence of investigations:

- How can existing feedback coding algorithms that are simple and human-implementable be applied to directly select informative interactions in general human-computer interaction (HCI) tasks?
- What challenges arise when applying these algorithms specifically to machine learning tasks, and how can these challenges be addressed in the case studies of interactive similarity and preference learning by gaining insights from each problem's query structure?
- How can we leverage a coding-theoretic modeling and analysis of active machine learning to modify existing optimal coding strategies for the design of general example selection policies?

After an introduction to information theory, feedback coding, and interactive learning in

Chapter 2, we explore these research thrusts across three parts.

In Chapters 3 and 4, we directly apply feedback coding to design an interaction algorithm for two HCI problems in brain-computer interfacing and image segmentation. We begin in Chapter 3 by studying feedback coding for brain-computer interfaces (BCI), which are systems that consist of hardware to measure a human user's brain activity, an interaction algorithm to map the user's mental commands to control signals, and an end effector that the user operates via these control signals. BCIs involve either invasive measurements which allow for high precision control but are generally infeasible, or noninvasive measurements which offer lower quality signals but are more practical to use. In general, BCI systems have not been developed that efficiently, robustly, and scalably perform high-complexity control while retaining the practicality of noninvasive measurements. In this chapter, we leverage a recently developed feedback coding scheme [22, 23] to fill this gap by modeling BCIs as a communications system and deploying a human-implementable interaction algorithm for noninvasive control of a high-complexity robot swarm. We construct a scalable dictionary of robotic behaviors that can be searched simply and efficiently by a BCI user, as we demonstrate through a large-scale user study testing the feasibility of our interaction algorithm, a user test of the full BCI system on (virtual and real) robot swarms, and simulations that verify our results against theoretical models. Our results provide a proof of concept for how a large class of high-complexity effectors (even beyond robotics) can be effectively controlled by a BCI system with low-complexity and noisy inputs.

In Chapter 4, we consider the problem of interactively specifying an object segment in an image in an efficient and robust manner via binary inputs corrupted by noise. Our method leverages a similar formulation and feedback coding scheme as in Chapter 3 that allows a user to interactively select a segment from an ordered lexicon of segments for a given image. We propose an intuitive lexicon based on ellipses (EllipseLex) and evaluate its ability to specify desired object segments over increasing numbers of inputs at various levels of input noise, and compare it to a baseline algorithm. After evaluating the performance

of each method on the Microsoft Common Objects in Context (MS-COCO) dataset using several metrics, we find that our method exhibits competitive performance when specifying real-world objects in images.

In Chapters 5 and 6, we describe how the algorithms used in Chapters 3 and 4 cannot generally be directly applied to problems in machine learning, and instead explore how specific query structures can still motivate approximate information maximization methods for interactive similarity and preference learning. These chapters explore how many machine learning tasks such as clustering, classification, recommender systems, and dataset search benefit from embedding data points in a space where distances reflect notions of relative similarity as perceived by humans. A common way to construct such an embedding is to request triplet similarity queries to an oracle, comparing two objects with respect to a reference. In Chapter 5 we generalize triplet queries to tuple queries of arbitrary size that ask an oracle to rank multiple objects against a reference, and introduce an efficient and robust adaptive selection method called InfoTuple that uses a novel approach to information gain maximization. We show that the performance of InfoTuple at various tuple sizes exceeds that of the state-of-the-art adaptive triplet selection method on synthetic tests and new human response datasets, and empirically demonstrate the significant gains in efficiency and query consistency achieved by querying larger tuples instead of triplets.

Once such an ordinal embedding is constructed, in Chapter 6 we consider the task of estimating a user's preference vector w from paired comparisons of the form "does user w prefer item p or item q?," where both the user and items are embedded in the same low-dimensional Euclidean space with distances that reflect user and item similarities. Such observations arise in numerous settings, including psychometrics and psychology experiments, search tasks, advertising, and recommender systems. In such tasks, queries can be extremely costly and subject to varying levels of response noise; thus, we aim to *actively* choose pairs that are most informative given the results of previous comparisons. We provide new theoretical insights into the benefits and challenges of greedy information maximization

in this setting, and develop two novel strategies that maximize lower bounds on information gain and are simpler to analyze and compute respectively. We use simulated responses from a real-world dataset to validate our strategies through their similar performance to greedy information maximization, and their superior preference estimation over state-of-the-art selection methods as well as random queries. We also consider a time-varying extension of this problem, in which w evolves according to some unknown dynamics model. In this extension, we consider the task of actively selecting informative paired comparisons between landmark points to jointly estimate the state trajectory and identify the true dynamics model from a finite set of candidate models.

In Chapter 7, we propose a new feedback coding scheme specific to IML problems to address the challenges encountered in Chapters 5 and 6 by proposing a generic approximation to the feedback coding algorithm utilized in Chapters 3 and 4. We focus specifically on how the iterative selection of examples for labeling in active machine learning (where the task of example selection lies with the learner, as opposed to machine teaching) is conceptually similar to feedback channel coding: in both tasks, the objective is to seek a minimal sequence of actions to encode information in the presence of noise. While this high-level overlap has been previously noted, there remain open questions on how to best formulate active learning as a communications system to leverage existing analysis and algorithms in feedback coding. In this chapter, we formally identify and leverage the structural commonalities between the two problems, including the characterization of encoder and noisy channel components, to design a new algorithm. Specifically, we develop an optimal transport-based feedback coding scheme called *Approximate Posterior Matching* (APM) for the task of active example selection and explore its application to Bayesian logistic regression, a popular model in active learning. We evaluate APM on a variety of datasets and demonstrate learning performance comparable to existing active learning methods, at a reduced computational cost. These results demonstrate the potential of directly deploying concepts from feedback channel coding to design efficient active learning strategies.

We conclude in Chapter 8 with a summary of our findings and discussion of future directions. This thesis is the product of a series of fruitful and exciting collaborations; at the start of each chapter, we list in a footnote the contributions of each project collaborator.

CHAPTER 2

BACKGROUND

In this chapter, after introducing our mathematical notation we briefly describe core problems, quantities, and concepts in information theory, coding theory, and interactive learning.

2.1 Mathematical Preliminaries

When introducing scalars, vectors, or matrices, we immediately define their domains (e.g., $x \in \mathbb{R}^d$ or $x \in \mathbb{R}$) rather than distinguishing between these quantities with boldface or other notation. We denote random variables and vectors by uppercase letters (e.g., X) and observed instantiations of random variables and vectors with lowercase letters (e.g., x). In a slight abuse of terminology, in general we forgo the distinction between random variables and vectors, and refer generally to both as "variables," with dimensionality being clear from context (e.g., random variable $X \in \mathbb{R}^d$). We denote sequences of variables (random or deterministic) with subscripts and superscripts as $Y_j^k = \{Y_i\}_{i=j}^k = \{Y_j, Y_{j+1}, \dots Y_{k-1}, Y_k\}$. When $j = 1$, we simply write Y^k to indicate $\{Y_i\}_{i=1}^k = \{Y_1, Y_2, \dots Y_{k-1}, Y_k\}$.

If A is a finite set of discrete elements, then $|A|$ denotes the cardinality of A, i.e., the number of elements in A. $|\Sigma|$ is also used to denote the determinant of square matrix Σ, and we assume that the distinction between cardinality and determinant will be clear from context. If A is an uncountable set $A \subset \mathbb{R}^d$, then Vol denotes the volume of A:

$$\mathrm{Vol}(A) = \int_{\mathbb{R}^d} \mathbb{1}_A(x)\, dx = \int_A dx,$$

where $\mathbb{1}_A(x)$ denotes the indicator function on set A, i.e., $\mathbb{1}_A(x) = 1$ if $x \in A$ and $\mathbb{1}_A(x) = 0$ otherwise. We use $\log$ to denote the natural logarithm unless stated otherwise, in which case the base will always be given explicitly (e.g., $\log_2$).

The probability of an event E is denoted by $\mathrm{P}(E)$. We use the same notation for both probability mass functions (p.m.f.) of discrete random variables and probability density functions (p.d.f.) of continuous random variables, and leave the distinction to context. For two random variables X and Y, we notate their joint density as $p_{X,Y}(x, y)$, their marginal densities as $p_X(x)$ and $p_Y(x)$, and their conditional densities as $p_{Y|X}(y \mid x)$ and $p_{X|Y}(x \mid y)$. If the distribution arguments are clear from context, or if we wish to refer to a distribution as its own entity, we will sometimes omit the arguments from notation and refer to $p_{X,Y}$, p_X, p_Y, $p_{Y|X}$ and $p_{X|Y}$. Conversely, we sometimes instead omit the subscript, and include only the arguments, i.e., $p(x, y)$, $p(x)$, $p(y)$, $p(y \mid x)$, and $p(x \mid y)$, in which case the relevant random variables are determined from the arguments. We use similar conventions to denote the expectation of a random variable, with $\mathbb{E}_X[X]$ being equivalent to $\mathbb{E}[X]$ if the notation is clear in context. When taking the expected value of a function $f(X)$ over X, we will sometimes explicitly highlight the variate $x \sim p_X$ in writing expectations, as $\mathbb{E}_{x \sim p_X}[f(x)]$.

One important class of distributions for continuous random variables are those that are *log-concave*, meaning that their probability density functions $p(w)$ satisfy $p(\alpha w_1 + (1-\alpha)w_2) \geq p(w_1)^{\alpha} p(w_2)^{1-\alpha}$ for any $w_1, w_2 \in \mathbb{R}^d$ and $0 < \alpha < 1$. More generally, any function p that satisfies this property is also referred to as log-concave. The class of log-concave distributions is broad, and includes many common distributions such as the normal, exponential, and uniform distributions. Log-concave distributions and functions have several convenient properties, including that log-concavity of distributions is preserved under marginalization [24], and log-concavity of functions is preserved under multiplication [25].

The latter property is particularly useful when studying log-concave distributions in Bayesian inference: for a hidden random variable θ with prior distribution p_θ, and M independent observations $Y^M = \{Y_1, Y_2, \ldots Y_M\}$ each distributed according to $p_{Y_i|\theta}$, we

can use Bayes' rules to write the posterior distribution for θ after observing Y^M:

$$p_{\theta|Y^M}(\theta \mid Y^M) = \frac{p_\theta(\theta) \prod_{i=1}^M p(Y_i \mid \theta)}{p(Y^M)}. \tag{2.1}$$

Since $p(Y^M)$ is simply a normalizing constant that does not depend on θ, we observe from eq. (2.1) that the posterior density of θ after observing Y^M is proportional to a product of the prior and observation likelihoods. If both the prior and likelihoods are log-concave with respect to θ, then the product property of log-concave functions implies that $p_{\theta|Y^M}$ is also log-concave.

2.2 Information Theory, Entropy, and Mutual Information

Information theory is a field of study that has revolutionized statistical technologies over the past seventy years, having profound impacts on fields from telecommunications to machine learning. The field was born in the 1940s out of the pursuit for a fundamental understanding of how information (e.g., messages, data) could be stored and transmitted efficiently, in either the presence or absence of corrupting noise. While this *coding* problem had been contemplated previously, it was the seminal work and ideas of Claude Shannon in his *Mathematical Theory of Communication* that formally defined the statistical notion of information and used the set of resulting tools to both prove fundamental limits on coding performance and mathematically guarantee the existence of optimal codes to represent and transmit data in the presence of noise [26, 15]. This work set the foundation for information and coding theory in the years that followed, providing a set of tools and approaches for studying the storage and transmission of information as well as delineating clear performance benchmarks for engineers to strive for when designing new coding systems. Following in the footsteps of a long line of work applying these tools to signal processing and machine learning, in this thesis we further expand their application to problems in IML.

The first quantity typically studied in information theory[1] is the *entropy* of a discrete random variable $X \in \mathcal{X}$ with p.m.f. $p_X(x)$, defined as

$$H(X) = \mathbb{E}_{x \sim p_X}\left[\log_2 \frac{1}{p_X(x)}\right] = \sum_{x \in \mathcal{X}} p_X(x) \log_2 \frac{1}{p_X(x)}, \tag{2.2}$$

where by convention $0 \log 0 = 0$. Entropy has been widely adopted in information theory, coding theory, and other fields as a standard measure of uncertainty of a random variable. Intuitively, the reciprocal of the p.m.f. is a measure of the "surprise" in observing a particular random variate, with less (resp. more) likely events being more (resp. less) surprising to observe. The logarithm ensures that "surprise" is additive when observing independent random variables, which is a convenient mathematical property for such a measure. If uncertainty is interpreted to mean the expected level of surprise in observing a random variable, then the definition of entropy in eq. (2.2) follows naturally. We use the base 2 logarithm in entropy and all information-theoretic quantities that follow, such that the resulting units of uncertainty and information are in *bits*.

When the distribution of X is conditioned on fixed observations of a discrete random variable $Y \in \mathcal{Y}$, we can evaluate entropy with the conditional p.m.f. $p_{X|Y}(X \mid y)$ instead:

$$H(X \mid Y = y) = \mathbb{E}_{x \sim p_{X|y}}\left[\log_2 \frac{1}{p_{X|Y}(X \mid y)}\right] = \sum_{x \in \mathcal{X}} p_{X|Y}(x \mid y) \log_2 \frac{1}{p_{X|Y}(x \mid y)},$$

which we call the *fixed conditional entropy*[2]. For shorthand, we will sometimes denote this as $H(X \mid y)$, where conditioning on a lowercase variate indicates conditioning on a fixed observation. Calculating the entropy of X according to the p.m.f. $p_{X|Y}$ represents a measure of uncertainty of X upon observing a fixed instance of Y. When we average this measure of uncertainty over the marginal distribution of Y given by $p_Y(y)$, we have the *conditional entropy* of X conditioned on Y, which represents an aggregate measure of uncertainty of X

[1] The material from this section is largely drawn from [27].

[2] It would be more standard to call this quantity the discrete entropy of X conditioned on a fixed observation of Y, but we adopt this terminology for conciseness.

upon observing Y:

$$\begin{aligned} H(X \mid Y) &= \mathbb{E}_{Y \sim p_Y}[H(X \mid Y = y)] \\ &= \sum_{y \in \mathcal{Y}} p_Y(y) H(X \mid Y = y) \\ &= \sum_{y \in \mathcal{Y}} p_Y(y) \left[\sum_{x \in \mathcal{X}} p_{X|Y}(x \mid y) \log_2 \frac{1}{p_{X|Y}(x \mid y)} \right] \\ &= \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} p_{X,Y}(x, y) \log_2 \frac{1}{p_{X|Y}(x \mid y)}, \end{aligned} \tag{2.3}$$

where $p_{X,Y}(x, y) = p_Y(y) p_{X|Y}(x \mid y)$ is the joint p.m.f. of X and Y.

We can also define similar quantities for continuous random variables. If X has a p.d.f. given by $p_X(x)$, the *differential entropy* of X is defined as

$$h(X) = \mathbb{E}_{x \sim p_X} \left[\log_2 \frac{1}{p_X(x)} \right] = \int_{\mathcal{X}} p_X(x) \log_2 \frac{1}{p_X(x)} \, dx.$$

While differential entropy is not equivalent to discrete entropy in the limit, in some cases it can still be interpreted as a measure of uncertainty in the sense of measuring the volume of a distribution's support. For instance, if $X \in \mathbb{R}^d$ is uniformly distributed over a set $A \subset \mathbb{R}^d$, and the volume of A is well defined, then differential entropy corresponds directly to this volume:

$$h(X) = \int_A \frac{1}{\mathrm{Vol}(A)} \log_2(\mathrm{Vol}(A)) \, dx = \log_2(\mathrm{Vol}(A)).$$

Similarly, when $X \in \mathbb{R}^d$ is a multivariate Gaussian distribution with covariance matrix Σ, then $h(X) = \frac{1}{2} \log_2 \left[(2\pi e)^d |\Sigma| \right]$. In this case, differential entropy corresponds geometrically to the volume of an ellipsoid defined by the eigenvectors of Σ.

In a similar manner to the discrete case, we can define the fixed conditional differential entropy of continuous random variable $X \in \mathcal{X}$ when conditioned on continuous random

variable $Y \in \mathcal{Y}$, where in this case $\mathcal{X}$ and $\mathcal{Y}$ are continuous sets:

$$h(X \mid Y = y) = \mathbb{E}_{x \sim p_{X|y}}\left[\log_2 \frac{1}{p_{X|Y}(X \mid y)}\right] = \int_{\mathcal{X}} p_{X|Y}(x \mid y) \log_2 \frac{1}{p_{X|Y}(x \mid y)}\, dx.$$

We can also define the corresponding *conditional differential entropy* given by

$$\begin{aligned} h(X \mid Y) &= \mathbb{E}_Y[h(X \mid Y = y)] \\ &= \int_{\mathcal{Y}} p_Y(y) \left[\int_{\mathcal{X}} p_{X|Y}(x \mid y) \log_2 \frac{1}{p_{X|Y}(x \mid y)}\, dx\right] dy \\ &= \int_{\mathcal{Y}} \int_{\mathcal{X}} p_{X,Y}(x, y) \log_2 \frac{1}{p_{X|Y}(x \mid y)}\, dx\, dy. \end{aligned} \tag{2.4}$$

As in the discrete case, we sometimes write $h(X \mid y)$ as a shorthand for $h(X \mid Y = y)$. More generally, we can define conditional entropies between mixtures of discrete and continuous random variables by simply altering the domain of the outer expectation in either eq. (2.3) or eq. (2.4). Specifically, for continuous random variable X and discrete random variable Y (with p.d.f.'s $p_{X|Y}$ and p_X and p.m.f.'s $p_{Y|X}$ and p_Y defined accordingly), we have

$$H(Y \mid X) = \mathbb{E}_{x \sim X}[H(Y \mid x)] \qquad h(X \mid Y) = \mathbb{E}_{y \sim p_Y}[h(X \mid y)].$$

With entropy and conditional entropy defined for discrete and continuous random variables, we now introduce *mutual information*, which is arguably the central quantity in information theory. With a slight abuse of integral notation (substituting summations as needed for discrete random variables), the mutual information between random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ with joint distribution $p_{X,Y}(x, y)$ and marginal distributions $p_X(x)$ and $p_Y(y)$ is defined as

$$I(X; Y) = \int_{\mathcal{Y}} \int_{\mathcal{X}} p_{X,Y}(x, y) \log_2 \frac{p_{X,Y}(x, y)}{p_X(x) p_Y(y)}\, dx\, dy. \tag{2.5}$$

While there are many motivations, interpretations, and representations of mutual information,

here we focus on a difference of entropies formulation. In particular, in this thesis we are mainly concerned with the mutual information between one discrete, and one continuous random variable. It is immediately clear from eq. (2.5) that mutual information is symmetric with respect to X and Y, and so without loss of generality we assume X is continuous, and Y is discrete. By expanding terms and using the fact that $p_{X,Y}(x, y) = p_X(x)p_{Y|X}(x \mid y)$, we can rewrite eq. (2.5) as

$$I(X;Y) = H(Y) - H(Y \mid X). \tag{2.6}$$

This form of mutual information leads to a classic interpretation: the first term of eq. (2.6) measures the uncertainty of random variable Y, and the second term measures the resulting uncertainty of Y after having observed X. Thus, mutual information corresponds to the *reduction in uncertainty* about Y, upon observing X.

Similarly, we can expand eq. (2.5) by instead factoring $p_{X,Y}(x, y) = p_Y(y)p_{X|Y}(x \mid y)$, resulting in

$$I(X;Y) = h(X) - h(X \mid Y). \tag{2.7}$$

While differential entropy on its own cannot be interpreted in the same manner as discrete entropy, the *difference* of differential entropies in eq. (2.7) is strongly related to the discrete case since in the limit it is equivalent to approximating X as a discrete random variable and measuring the corresponding reduction in discrete entropy. When differential entropy corresponds to volume (as in the uniform or Gaussian case), eq. (2.7) can also be interpreted as the expected reduction in the support volume of X, upon observing Y. We explore this interpretation more formally in the context of active estimation of user preferences in Chapter 6.

In many problems in coding theory and interactive learning, the joint distribution between two random variables X and Y is most naturally represented as the product of marginal $p_X(x)$ and conditional distribution $p_{Y|X}(y \mid x)$. For this reason, we also introduce notation

to represent mutual information as an explicit function of marginal distribution p_X and conditional distribution $p_{Y|X}$ given by $I(p_X, p_{Y|X}) \coloneqq I(X;Y)$. Finally, as we explore in Section 2.3.1 it is sometimes the case that the distributions of X and Y are conditioned on a third variable $Z \in \mathcal{Z}$, and in a similar manner to conditional entropy we can define *fixed conditional mutual information* as

$$I(X;Y \mid Z=z) = \int_{\mathcal{Y}} \int_{\mathcal{X}} p_{X,Y|Z}(x,y \mid z) \log_2 \frac{p_{X,Y|Z}(x,y \mid z)}{p_{X|Z}(x \mid z) p_{Y|Z}(y \mid z)} \, dx \, dy,$$

and *conditional mutual information* as

$$\begin{aligned} I(X;Y \mid Z) &= \mathbb{E}_{z \sim p_Z}[I(X;Y \mid Z=z)] \\ &= \int_{\mathcal{Z}} p_Z(z) \int_{\mathcal{Y}} \int_{\mathcal{X}} p_{X,Y|Z}(x,y \mid z) \log_2 \frac{p_{X,Y|Z}(x,y \mid z)}{p_{X|Z}(x \mid z) p_{Y|Z}(y \mid z)} \, dx \, dy \, dz \\ &= \int_{\mathcal{Z}} \int_{\mathcal{Y}} \int_{\mathcal{X}} p_{X,Y,Z}(x,y,z) \log_2 \frac{p_{X,Y|Z}(x,y \mid z)}{p_{X|Z}(x \mid z) p_{Y|Z}(y \mid z)} \, dx \, dy \, dz. \end{aligned}$$

We sometimes use $I(X;Y \mid z)$ to indicate fixed conditional mutual information, where the lowercase variate z indicates conditioning on a fixed observation. It is easy to show that $I(X;Y \mid z) = H(X \mid z) - H(X \mid Y, z)$ and similarly $I(X;Y \mid Z) = H(X \mid Z) - H(X \mid Y, Z)$, where the notation $H(X \mid Y, z)$ simply indicates a conditional entropy $H(X \mid Y)$ where the joint distribution over X, Y is conditioned on observing $Z = z$, i.e., $p_{X,Y|Z}(x, y \mid z)$. Conditioning on multiple random variables as in $H(X \mid Y, Z)$ can be understood by defining $Y' = (Y, Z)$ and computing $H(X \mid Y') = \mathbb{E}_{y' \sim p_{Y'}}[H(X \mid y')] = \mathbb{E}_{y,z \sim p_{Y,Z}}[H(X \mid y, z)]$.

2.3 Channel Coding Theory

With notions of statistical uncertainty and information defined, we can directly apply these quantities to the fundamental problem of information theory: efficiently and robustly encoding a message with a set of symbols. In a typical telecommunications framework, an

information source has a *message* that it wishes to send across some medium, or *channel*, to a recipient. To accomplish this, the message must first be translated to an *encoding*, or a representation compatible with transmission over the channel. This abstraction is applicable to almost any scenario where information is transferred from one point to another; when we as humans converse with a listener, we translate the words in our minds into sound waves traveling through air, and when we wish to capture an image with a camera and send it across the internet, the image must be first stored in a binary representation which is then transmitted via a digital signal. Similarly in IML, when a human expert wishes to teach a concept to a machine, they must do so by encoding their knowledge through a set of labeled training examples. Once the message is appropriately encoded, it is then transmitted across the channel and received by the recipient. Assuming that the recipient understands how to translate, or *decode*, the transmitted encoding back into its original format, it can then understand the message sent by the source.

Often times, the information source wishes to transmit their message by using as compact of an encoding as possible in order to minimize the complexity of the translation process from message to encoding and back, as well as to minimize the usage of the transmission channel, which in many real-world scenarios has an associated cost for each use (e.g., memory, energy, bandwidth). However, during the process of transmission it is possible for the message encoding to be corrupted such that it exits the channel at the receiver in a corrupted form. We generally refer to this corruption as *noise*, and call a channel with corruption a *noisy channel*. Therefore, the source wishes to use an encoding process that not only minimizes the complexity and cost of transmission, but also somehow ensures that the receiver will be able to decode the message, even if it is corrupted along the way.

These two goals of compactness and robustness to noise are often in conflict with one other, since the primary mechanism to build robustness into a code is to intentionally add redundancy into the encoding scheme. Intuitively, encoding a message with some degree of redundancy mitigates the risk of overall message corruption, since by definition the

message can be recovered with only a subset of the transmitted code remaining intact. A classic example of redundant coding that illustrates the tradeoff between compactness and robustness is *repetition coding*, where a message is first represented as a sequence of zeros or ones — or *bits*, not to be confused with the information unit described previously — and at transmission time each bit is repeated K times. The receiver then processes each block of K bits by taking a majority vote and inferring the original bit. Even if some repetition bits are corrupted during transmission, each original bit will be inferred correctly (and the source message decoded corrected) as long as a majority of repetition bits are left uncorrupted in each block. Because the degree of noise protection scales with K, this means that increased robustness comes at the cost of transmitting a signal that is K times longer than the original encoding.

To make these concepts more precise, suppose that the source's goal is to transmit a message θ belonging to set S by mapping the message to a sequence of n symbols $L^n = \{L_1, L_2, \ldots, L_n\}$ each belonging to a symbol alphabet $\mathcal{A}$. In the problem of *source coding*, a recipient has direct access to L^n and decodes this symbol set into a decoded message $\widehat{\theta}_n \in S$, with the hope that $\widehat{\theta}_n = \theta$. In this case, an efficient source coding scheme will map the message into as few symbols as possibly while simultaneously preserving the content of the message θ. More realistically, in many settings the symbols L^n are passed through a channel and corrupted by noise, resulting in a noisy set of corresponding symbols $Y^n = \{Y_1, Y_2, \ldots, Y_n\}$ belonging to alphabet $\mathcal{Y}$. This noisy transmission process is typically modeled as sampling each Y_i from a fixed probability distribution $p_{Y|L}$ conditioned on L_i; in communications terms, each L_i is a *channel input* symbol passing through a *noisy channel* $p_{Y|L}$, resulting in a corresponding *channel output* symbol Y_i, and the total set of outputs Y^n is then decoded into a potentially erroneous $\widehat{\theta}_n$. Since the statistics of Y_i only depend on the input statistics of L_i, the channel $p_{Y|L}$ is considered a *memoryless* channel. When both $\mathcal{A}$ and $\mathcal{Y}$ are discrete sets, $p_{Y|L}$ is called a *discrete memoryless channel*. In this *channel coding* scenario, redundancy must be added by the encoder (in the spirit of repetition coding) such

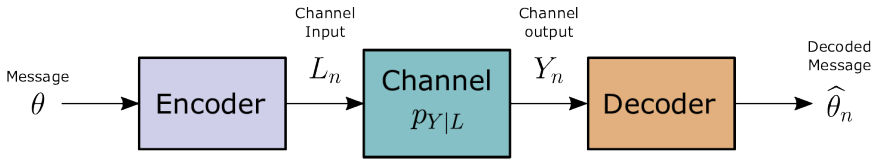


Figure 2.1: Channel coding in a communications system.

that the decoder can robustly infer the sent message from the channel outputs, even in the presence of channel noise (full communications system depicted in Figure 2.1).

An important quantity in this framework is the *information gain* across the channel, which is the mutual information $I(L;Y)$ between the channel input and channel output. Recalling from Section 2.2 the reduction in uncertainty interpretation of mutual information, information gain measures the decrease in uncertainty about channel input L upon receiving Y, i.e.,[3] $I(L;Y) = H(L) - H(L \mid Y)$. In a well-designed coding scheme, this reduction in uncertainty should be maximized. To see this, note that if L were known to the decoder before observing Y, then receiving Y does not tell the decoder anything about the message it didn't already know. Mathematically, this means that $H(L)$ must be large for the transmission and receipt of Y to be of any use to the decoder. Even if transmitting L does in fact have the potential to be informative to the decoder (i.e., large $H(L)$), the received output Y is only useful to the decoder if transmission over the channel does not detrimentally corrupt the input. If this level of corruption is low, then roughly speaking the channel input is recoverable from the channel output and so the uncertainty about L is low upon receiving Y, i.e., $H(L \mid Y)$ is small. Combining these insights, a large information gain $H(L) - H(L \mid Y)$ corresponds to informative transmission over the channel.

Since $p_{Y|L}$ is fixed for any particular channel, the information gain depends solely on the probability distribution of channel inputs p_L utilized by a particular coding scheme, which is known as the *channel input distribution*. A natural question to ask is, what is the maximum amount of information that can be gained across a given channel, and under what input distribution p_L does this maximization occur? The maximum information

[3]For the sake of exposition we use discrete random variables in this discussion, but the results here generalize to continuous random variables.

gain is called the *channel capacity* and is denoted by $C \coloneqq \max_{p_L} I(p_L, p_{Y|L})$, and the maximizing distribution is called the *capacity-achieving distribution* which we denote by $p_L^* \coloneqq \arg\max_{p_L} I(p_L, p_{Y|L})$. Arguably the most fundamental and important result in channel coding theory concerns the relationship between channel capacity and the performance limits of any possible coding scheme. This classic result is Shannon's Noisy Channel Coding Theorem, which we summarize below. In our statement of the theorem, we define a coding scheme as *achieving* a rate $R > 0$ if $|S| = 2^{Rn}$ for n transmitted symbols and $P(\widehat{\theta}_n \neq \theta) \to 0$ as $n \to \infty$. Intuitively, the achievable rate of a code measures the number of possible messages that can be reliably transmitted with only n channel uses, with higher rates indicating more efficient codes. We say that a coding scheme is *capacity-achieving* or *optimal* if it achieves the channel capacity C.

Theorem 2.3.1 ([15])**.** *For every rate $R < C$ there exists a coding scheme that achieves R. Conversely, for every rate $R > C$ no such scheme exists. More generally, for coding schemes with information gain $I(L;Y)$, no rate $R > I(L;Y)$ is achievable.*

This theorem laid the groundwork for all of channel coding theory, since it showed that communication is possible with arbitrarily small error across a noisy channel while simultaneously defining its efficiency limitations, which sent telecommunications engineers on a search for capacity-achieving coding schemes while preventing them from attempting to design any schemes that achieved a rate higher than C. Furthermore, the theorem's converse motivates coding schemes that maximize the information gain $I(L;Y)$ in general, since this quantity upper bounds the achievable rates of a given code.

2.3.1 Coding with Noiseless Feedback

Since the early years of information theory, one particular problem of interest has been channel coding in the presence of *feedback* [28, 29, 30]. In this setting, the encoder chooses symbols iteratively (i.e., one after another in a sequence) and after each channel output is received, a signal (such as an acknowledgment of receipt) is transmitted back from the

receiver to the encoder. This feedback signal then functions as side information during the next iteration of encoding, and better informs the encoder's choice of the subsequent channel input. In the case where all received channel outputs are noiselessly available to the encoder, the channel is said to have *noiseless feedback* (depicted in Figure 2.2). Intuitively, noiseless feedback allows the encoder to fully observe what information is still "missing" at the decoder so that it can better select its encoded symbols [31]. At any particular iteration n, all probability distributions at the encoder and decoder are conditioned on observing y^{n-1}, so that the information gain across the channel becomes $I(L_n; Y_n \mid y^{n-1}) = I(p_{L_n|y^{n-1}}, p_{Y|L})$ and the posterior distribution of the message (the message being unknown to the decoder) is $p_{\theta|y^{n-1}}$.

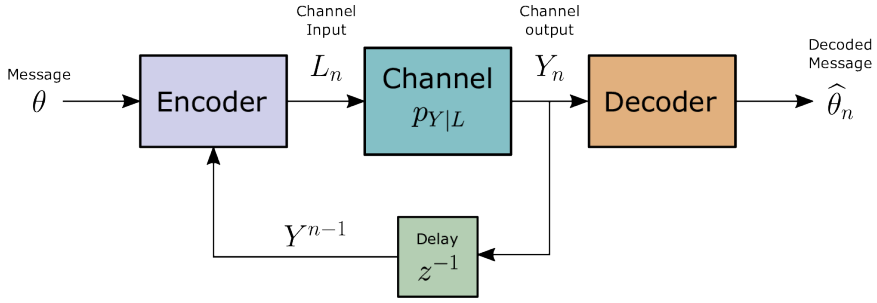


Figure 2.2: Channel coding with feedback.

Without feedback, capacity-achieving coding schemes typically involve complex forward error-correcting codes that require elaborate computational mechanisms to implement. However, in the presence of noiseless feedback the existence of *simple* capacity-achieving codes have been shown that involve only basic operations on the transmitted message and previous channel outputs [29, 30]; we will show in Chapters 3 and 4 how this simplicity leads to feedback coding schemes that are implementable by humans to convey their intentions in human-computer interaction. Furthermore, although noiseless feedback does not increase the channel capacity of discrete memoryless channels [28], it has been shown to increase the rate of decoding error decay in some cases [30] and can increase capacity for certain channels with memory [27].

2.3.2 Posterior Matching

In recent years, a capacity-achieving coding scheme known as *posterior matching* (PM) has been proposed as a general approach to feedback coding [22]. The mechanism behind PM is a simple, yet powerful idea: at each iteration, an encoder mapping is constructed such that each channel input is distributed as the capacity-achieving distribution and is statistically independent of all previous channel outputs. These conditions ensure that the overall information conveyed about the message over the channel (i.e., $I(\theta; Y^n)$) is maximized at nC, which is the maximum possible reduction in uncertainty about the message after having received a length-n sequence of channel outputs at the decoder [32]. For message $\theta \in [0, 1]$, channel inputs $L_i \in \mathbb{R}$, and channel outputs y^{n-1} observable at the encoder, the PM encoding mapping is given by

$$g_n(\cdot, y^{n-1}) = F_L^{-1} \circ F_{\theta|y^{n-1}}(\cdot \mid y^{n-1}) \qquad L_n = g_n(\theta, y^{n-1}), \tag{2.8}$$

where F_L^{-1} is the inverse c.d.f. of the channel's capacity-achieving distribution, and $F_{\theta|y^{n-1}}$ is the c.d.f. of the message posterior $p_{\theta|y^{n-1}}$. The intuition behind PM relies on properties of transforming a random variable through c.d.f.'s: transforming θ by $F_{\theta|y^{n-1}}(\cdot \mid y^{n-1})$ results in a uniformly distributed random variable, which when subsequently transformed through F_L^{-1} is shaped to the capacity-achieving distribution. In essence, at every channel input the message posterior distribution is "matched" to the capacity-achieving distribution in order to maximize information transfer across the channel.

One remarkable aspect about PM is that it generalizes several existing feedback coding schemes [22]; when transmitting over a binary symmetric channel (see next section) PM simplifies to the one-bit feedback scheme introduced by Horstein [29]. When transmitting over an additive white Gaussian noise channel, PM distills to the intuitive Schalkwijk-Kailath scheme, which transmits the error of a minimum mean square error estimate as a means of refining the decoder state [30]. As we discuss in Chapter 7, PM has also been

extended to feedback communication with multidimensional message, channel input, and channel output spaces [31].

2.3.3 Posterior Matching over a Binary Symmetric Channel

One posterior matching application of particular interest is the case of feedback coding over a *binary symmetric channel* (BSC). A BSC with *crossover probability* $0 \leq \varepsilon \leq 0.5$ is a binary input ($L \in \{0, 1\}$) binary output ($Y \in \{0, 1\}$) channel defined by the following transition probability (diagrammed in Figure 2.3a):

$$p(Y = y \mid L = \ell) = \begin{cases} 1 - \varepsilon & y = \ell \\ \varepsilon & y \neq \ell \end{cases}.$$

It is easy to show in this case that PM (eq. (2.8)) distills to the following scheme (diagrammed in Figure 2.3b):

$$L_n = \begin{cases} 0 & \theta < \text{median}(p_{\theta|y^{n-1}}) \\ 1 & \theta \geq \text{median}(p_{\theta|y^{n-1}}) \end{cases}. \tag{2.9}$$

In this simple coding rule, at each coding iteration the encoder indicates to the decoder if the message is less or greater than the median of the current message posterior distribution, after which the decoder updates its message posterior. If the posterior median is interpreted as a "guess" of the message, this strategy is akin to a noisy "twenty questions" style game between the encoder and decoder, where the encoder uses binary answers to refine the decoder's guess [33, 34]. In fact, since only the posterior median is required to implement this rule (rather than requiring knowledge of the full message posterior $p_{\theta|y^{n-1}}$), it is sufficient for the decoder to compute the median at the receiver and feed back this guess directly, rather than relaying the entire history of channel outputs. Another interpretation of PM over a BSC is related to the idea of binary search; at each iteration, the encoder effectively cuts the message posterior distribution in half, essentially executing a noise-tolerant version of

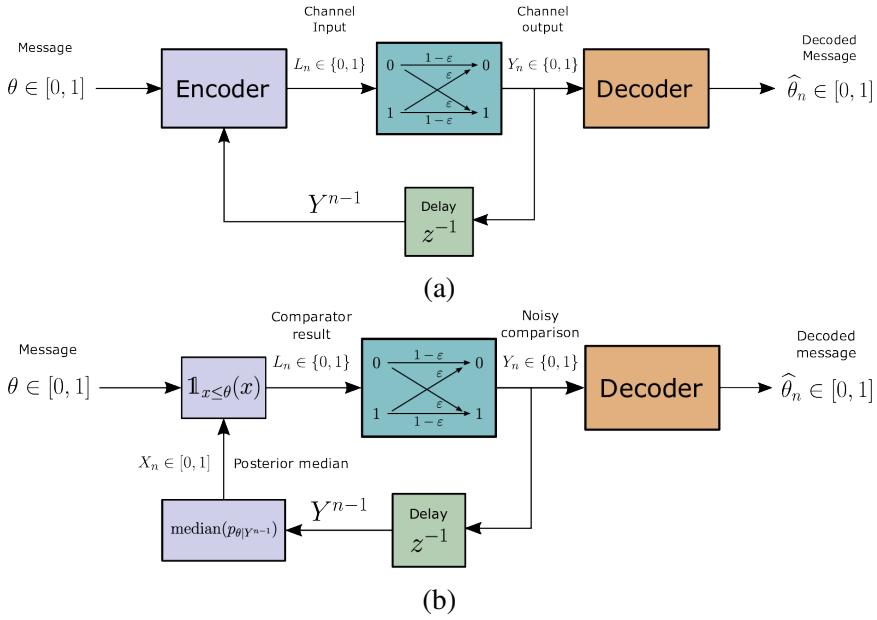


Figure 2.3: (a) Feedback coding over a BSC. (b) In posterior matching over a BSC, the encoder transmits the result of a comparator between the message and posterior median, which functions as an Bayesian estimate of the message.

binary search over the unit interval. For this reason, the algorithm is also known as the *probabilistic bisection algorithm* [35]. As we will discuss in the next section, the active bisection strategy in Figure 1.1b is closely related to PM over a BSC.

PM over a BSC has a long history in the coding-theoretic, active hypothesis testing, and adaptive sampling literature. The probabilistic bisection algorithm was originally proposed by Horstein, who studied it in the context described here of channel coding over a BSC with noiseless feedback [29]. Since Horstein's original work, the algorithm has been analyzed in depth [36] as well as extended to bisection in general hypothesis spaces [37]. It has also be specialized to the case of communicating a message from a discrete set of points rather than from the entire unit interval [38]. Following prior literature [35], we refer to this discretized version as the Burnashev-Zigangirov (BZ) algorithm. The BZ algorithm is conceptually similar to the PM rule in eq. (2.9), with slight modifications that take into account the discrete nature of the message set [38]. As we will describe, the BZ algorithm plays a critical role in Chapters 3 and 4. Due to the conceptual similarities with PM and

the fact that any differences come down to technicalities during bisection and posterior updating, for simplicity we sometimes refer to BZ interchangeably with PM and make it clear in implementation details whether the discretized version is being used.

2.4 Interactive Machine Learning

Conceptually, the statistical notions of uncertainty formalized by information and coding theory weave naturally into the question of learning from data. When a label is assigned to a data point and provided to a learner that seeks a hypothesis within a structured model class, the learner gains evidence for certain plausible structures in the data. That is to say, upon receiving a labeled data point the learner's uncertainty is reduced about the optimal model within their learning class. By measuring this reduction in uncertainty — which is a reduction that should be maximized at every iteration of learning — we can measure the "value" (or "utility") that labeling any particular data point provides to the learning process.

If various data points have different amounts of "value" to the learning process, it may be the case that one can train a learner with a *small* amount of *high-value* labeled data (in terms of reduction in model uncertainty) rather than training on a large dataset labeled passively, i.e., examples selected for labeling uniformly at random with varying levels of informativeness. Furthermore, it may also be the case that the value of labeling any particular data point *changes* over the course of learning; due to underlying structure between data points, labeling certain data may eliminate the need to label other data points that provide redundant information, or may give the learner new information that allows them to identify points that are likely to be noisy and therefore should not be requested for labeling. The fact that data points have different utility levels that may change throughout learning suggests that in order to train a learner efficiently by only labeling a small amount of high utility data, data should be selected both *judiciously* (only select informative data points for labeling) and *iteratively* (solicit labels for data "on the fly" to adapt to changing levels of informativeness). We refer to this combination of judicious, iterative selection of training data as *interactive*

selection.

More broadly, we use the term *interactive machine learning* in this thesis to refer to any scenario where a machine learns to make decisions (including classification, estimation, image segmentation, robotic control, or general effector control) by receiving supervised information (e.g., data labels, control signals) from an expert (or *oracle*), and where each training point is selected judiciously and iteratively by either the learner, the expert, or by some consensus between the two. Although the oracle is typically a human expert who supervises interactions, it may also consist of more abstract oracles such as the natural world (which implicitly provides labels when it is measured) or another machine. In this thesis, we consider two subfields of interactive machine learning, which we introduced briefly in Figure 1.1: in *active learning*, the learner has agency over which examples are selected for labeling, and in *machine teaching* the oracle has agency over example selection.

In active learning, data points are selected sequentially by a learner for labeling to train a model with as few labeled examples as possible, with a variety of approaches ranging from uncertainty sampling to querying by committee [39]. Minimizing the number of labeled examples is critical in any active learning scenario where labels are expensive to obtain, such as in healthcare applications where a medical expert must hand-label each training example [9], or where only a limited number of examples can be evaluated, such as in drug discovery [10] or adaptive sensing by mobile robots [40]. The study of active example or measurement selection dates back to the work of Lindley [41] and Chernoff [42] on the design of experiments for sequential hypothesis testing. These approaches have been subsequently extended to general Bayesian techniques for maximizing information in statistical experiments and estimation [20, 43]. Modern active learning methods vary considerably in their approach to example selection, ranging from coreset construction [44, 45] and adversarial learning of informative examples [46] to ensemble measures of example utility [47] and Bayesian information acquisition methods [48, 49].

In *machine teaching*, the oracle is engaged even more directly in the learning process

by functioning as a teacher who has an explicit role in selecting examples or interactions with the learner, rather than serving as a passive labeler who only responds to labeling requests by the machine [14]. While much of this thesis is specific to the scenario of active learning, as we discuss in Chapter 8 the extension of this work to the design of interaction policies in machine teaching is an exciting avenue for future work. In our formulation of active learning and machine teaching problems, we specifically focus on a Bayesian learning scenario where the oracle's knowledge is encapsulated in a set of parameters that during labeling is fixed and known exactly by the oracle, but a priori is modeled probabilistically with a prior distribution over a parameter space. Upon observing newly labeled data points, the learner updates a posterior distribution over this parameter space.

The fundamental idea in this thesis is that interactive learning shares many technical parallels with channel coding with feedback. The oracle has knowledge of ground truth parameters (e.g., decision surface parameters) that they wish to teach a machine learning algorithm, but they cannot communicate these parameters directly and instead must act on them by providing labels for individual data examples. In the language of coding theory, the oracle's knowledge plays the role of the "message" which is encoded to the learner through a sequence of interactions. Because the expert may be inconsistent in their labeling or the data features and model class may not be rich enough to model the oracle's true behavior, this labeling process is inherently noisy and can be modeled as a noisy communications channel. Each noisy interaction (e.g., labeled examples with noisy labels) plays the role of a "channel output" that is observable through feedback to select the next interaction. Both feedback channel coding and interactive learning seek to minimize the number of encoder actions, leverage a history of noisy observations to select the next most informative action, must account for observation noise, and should operate in a computationally efficient manner. By noting this overlap and capitalizing on the intuition that the value of a data label comes from the reduction in uncertainty it provides about the oracle's knowledge, one can translate the problem of example selection in interactive learning to a problem of channel coding, and

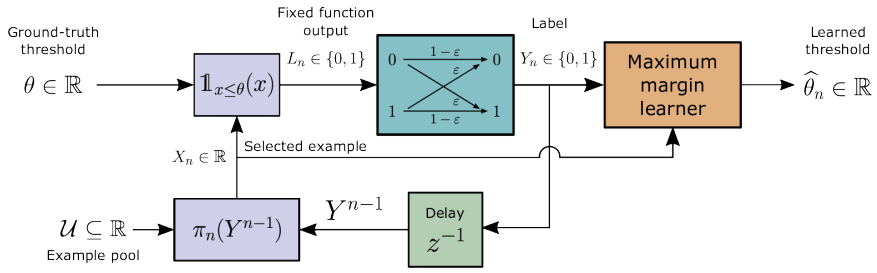


Figure 2.4: One-dimensional threshold classification as channel coding with feedback.

subsequently utilize tools from information and coding theory to both select and measure the informativeness of examples.

While related feedback coding formulations and insights have been proposed in the literature, in this work we focus on how identifying coding structures in each interactive problem setting leads to direct application of existing coding strategies (in particular, PM) or approximations to information maximization that leverage insights from the query structure. A key component of this process is the identification of the encoder structure in the equivalent interactive learning communications system. As an example, recall from the one-dimensional threshold example in Figure 1.1 that the expert interacted with the learner through a fixed function $h_\theta(x) = \mathbb{1}_{x \leq \theta}(x)$. In this example, the assigned label Y was taken directly as the output of this function, but one might imagine imposing a noise distribution $p_{Y|L}$ on the labels where $L = h_\theta(x)$. In this case, the noiseless label L functions as an "encoder output" which is then input into a "channel" given by $p_{Y|L}$ that corrupts the true label with noise, resulting in "channel output" Y. In a Bernoulli noise model where L is flipped with probability ε, we can model the channel by a BSC as in Figure 2.4. By comparing this system to Figure 2.3b, the resemblance to PM over a BSC is clear; this correspondence motivates the selection of the posterior median of $p_{\theta|y^{n-1}}$ — or an example in a finite pool $\mathcal{U}$ that closely approximates this point — as an active learning policy π_n to select the next example X_n for labeling. In fact, this approach is one way to motivate the bisection strategy used for active example selection in Figure 1.1b.

By identifying similar structures in Chapters 3 and 4, we directly deploy PM over a BSC

for intelligent interaction selection in HCI tasks. We then identify other comparison-based query structures in Chapters 5 and 6 for more general IML tasks, and discuss how PM is problematic to directly apply for example selection in these settings. Instead, we develop alternate strategies to approximately maximize information gain in a computationally efficient manner, utilizing these identified query structures. Finally, in Chapter 7 we formalize how the interaction mechanism of an expert acting upon ground truth parameters with a fixed function (i.e., encoder) passed through a fixed noise distribution (i.e., noisy channel) is common for many general active learning problems of interest such as logistic regression. We present an extension to PM that is compatible with these general active learning scenarios; specifically, we explore the idea of finding an encoder mapping that finds the *closest* input distribution to the capacity-achieving distribution, rather than p_L^* itself. We defer a detailed review of similar approaches in the literature to the related work section of each respective chapter, so that any related prior work is presented and discussed in context.

CHAPTER 3

INTERACTIVE BRAIN-COMPUTER INTERFACING FOR HIGH-COMPLEXITY EFFECTOR CONTROL

In this chapter and the next, we explore the application of PM to informative interaction selection in various human-computer interaction (HCI) tasks including brain-computer interfacing (BCI) for robot swarm control and interactive object segmentation, as a step towards understanding how coding principles can be applied to IML in general. In both BCI swarm control and interactive object segmentation we utilize a noisy one-bit input as a system interaction mechanism, which is common in HCI settings. Mathematically, we model the human user as having some intended behavior θ which they wish to communicate to a machine using a sequence of one-bit inputs, each denoted by $L_i \in \{0, 1\}$. Since there may be errors stemming from either incorrect user decisions or from the input mechanism itself, we model these interactions as being susceptible to a Bernoulli error with flipping probability $0 \leq \varepsilon \leq 0.5$, resulting in a noisy bit $Y_i \in \{0, 1\}$. As depicted in Figure 3.1a we can represent this interaction mechanism as a feedback communications system over a BSC with crossover probability ε, where the intent θ serves as a message being communicated to a computer that attempts to recover it from the sequence of noisy binary inputs.

As we will describe in detail, in both of our HCI applications we can model the human intent (e.g., swarm configuration or image segment) as a real number on the unit interval $\theta \in [0, 1]$. Furthermore, in both applications we introduce the ability for a human operator to compare any two behaviors in a way that mathematically translates to a comparator function $h_\theta(x) = \mathbb{1}_{x \leq \theta}(x)$. To do so, the set of possible intended behaviors must be constructed in such a way that a human operator can reliably state if one behavior comes before or after another behavior according to a total ordering. With this ordering, PM over a BSC can then be directly applied to design an efficient one-bit interaction mechanism where the computer

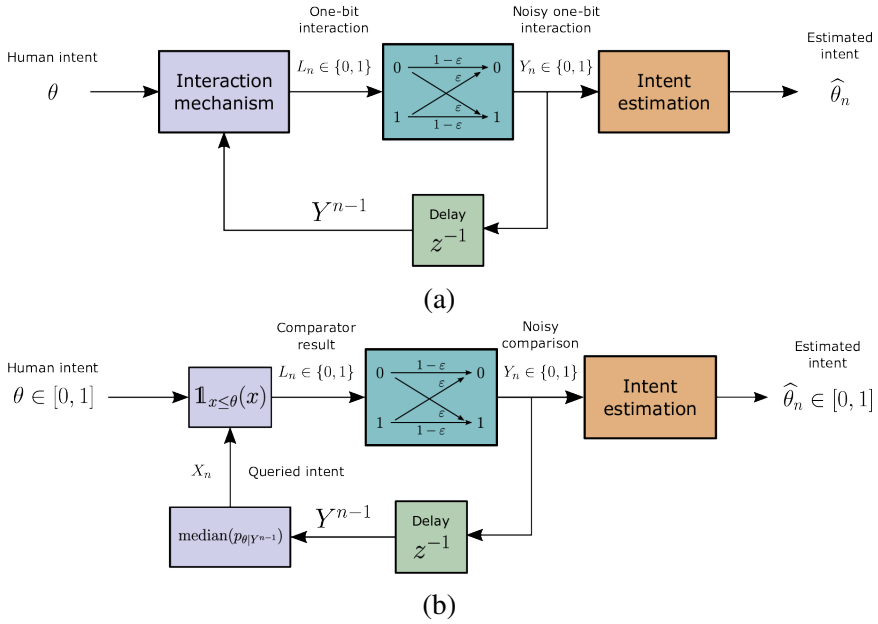


Figure 3.1: (a) One-bit interaction in human-computer interfaces as a coding problem. (b) If a total order exists for the set of possible human intents, then we can directly apply PM over a BSC to design a one-bit HCI interaction mechanism.

presents a guessed intent $X_n \in [0,1]$ to the human, who indicates (possibly with errors) if their intent precedes or succeeds the guess in the behavior ordering. Specifically, the computer guesses the behavior corresponding to $X_n = \text{median}(p_{\theta|Y^{n-1}})$ (Figure 3.1b).

PM and feedback coding models similar to Figure 3.1 have previously been directly applied for interaction selection in various areas of HCI including brain-computer interfacing [23], map localization [50], and aircraft path planning [51, 52]. However, as we demonstrate in this chapter and the next, there are significant differences between these previous efforts and our use of PM for general effector control. In particular, our approach greatly expands the complexity of effectors controllable by one-bit HCIs beyond that of previous work by designing richer sets of intended behaviors along with scalable rules for determining a total ordering.

3.1 Tradeoffs in Brain-Computer Interfacing

A BCI[1] is a system that allows a human operator to use only mental commands in controlling end effectors that interact with the world around them [53]. Such a system consists of a measurement device to record the human user's brain activity in the form of electrical signals, which are then processed into commands that drive a system end effector. This direct link between brain and effector provides a means for paralyzed users to circumvent muscular pathways and interact with everyday devices [54] as well as an augmented interface for healthy users. Although BCIs with invasive neural measurements have had experimental success in controlling high-complexity effectors (e.g., robotic arms [55, 56, 57, 58]) with many degrees of freedom, such BCIs are only available in research settings and require a surgical procedure for electrode implantation. BCIs with noninvasive measurements (e.g., scalp electrode recordings via an electroencephalogram (EEG)) are more widely implementable due to their relative ease of use and lower cost, but are limited to controlling comparatively simpler effectors (e.g., basic wheelchair control [59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70], cursor control [71, 72, 73, 74, 75, 76]) with few degrees of freedom due to lower signal-to-noise ratios. There are several tradeoffs involved in the design of BCIs, including whether measurements are taken invasively or noninvasively, how many mental commands are needed to drive the effector to a desired behavior, how scalable the system is to effectors of varying complexity, and how robust the system is to user error and noise in measurement processing.

In recent years there has been an emerging interest in improving these tradeoffs for neurotechnology in commercial and clinical applications, with aims to both broaden intended uses and engineer higher quality BCI devices [77, 78, 79, 80]. Despite this increased interest, there remains a large gap between the complexity of potential end effectors and the

[1]This chapter is in collaboration with Dr. Yancy Diaz-Mercado, Dr. Magnus Egerstedt, and Dr. Christopher Rozell. GC developed and tested the signal processing and dictionary search pipeline along with the associated experiments and analysis. YDM implemented the low-level swarm control algorithms and supervised experiments on the Robotarium. CR and ME supervised the project. This work has been submitted as a journal article, where GC is the lead author [1].

capabilities of interaction algorithms that map the user's mental commands from noninvasive interfaces to control signals. There are several specifications required for a noninvasive interaction algorithm to meet this need. First and foremost, any such algorithm must be implementable by humans through mental commands easily learned with training. Furthermore, such interaction algorithms must be scalable so that they remain tractable with a minimal increase in user overhead when controlling more complex effectors. Similarly, increases in effector complexity should not result in a need for increased measurement capabilities (e.g., additional EEG features). Because even the most advanced BCI measurements are susceptible to errors, an interaction algorithm must be robust to such errors. Finally, due to the wide range of applications that can benefit from BCIs, an ideal interaction algorithm should be designed for general use and be easily adaptable to a variety of specific tasks.

Currently, interaction algorithms for noninvasive BCIs fall into two broad categories that only achieve a subset of these specifications. In the first category, the user selects discrete effector behaviors from a finite set of options displayed on an interface, such as choosing waypoints for a motorized wheelchair [70] or selecting letters on a virtual keyboard [81, 82, 83, 84]. Although this type of interaction is easy to use, it scales poorly since it becomes increasingly tedious for a user to select their desired behavior as the number of options (i.e., the precision) increases. In the second category, continuous features from measured brain activity are directly mapped to continuous control over effector action spaces with arbitrary precision (e.g., robotic arm control [85, 86], quadcopter control [87], cursor control in up to three-dimensional space [71, 72, 73, 74, 75, 76]). Unlike discrete selection, continuous control allows a user to navigate an effector's action space with arbitrary precision in a scalable method. However, this type of interaction is severely limited in that each additional effector degree of freedom requires an independent, continuous measurement feature, which scales poorly and typically limits an effector to a maximum of three degrees of freedom for EEG-based BCIs.

Our new, robust interaction algorithm reaps the benefits of both discrete selection and

continuous control while addressing the disadvantages of each. The key innovation of our information-theoretic approach is that each new input is used in conjunction with closed-loop feedback to the user to efficiently refine the entire effector state simultaneously through a sequence of simple and tractable decisions. We test our approach on human control of a mobile robot swarm (a large collection of robots, as depicted in Figure 3.2c), where a human operator issues high-level, global commands which are executed by the swarm in a distributed fashion (individual robot depicted in Figure 3.2d).

Robot swarm control serves as an ideal testbed for our approach, since robot swarms are high-complexity cyber-physical systems that can be naturally parameterized beyond three degrees of freedom and have been previously tested in a BCI setting [88]. Part of what makes robot swarm control complex is the necessity to coordinate the individual robot motion to avoid collisions while attempting to achieve their objectives, e.g., reach a target formation. Robot swarms typically consist of weak robots which possess limited computation, sensing, and communication capabilities. Thus, in order to achieve the desired behavior, the control must rely on local sensing information and scale well in complexity with the number of robots in the swarm. These local rules result in the desired global emergent behavior. When humans are involved, the swarm formation must be achieved quickly and be cohesive enough to provide the human operator with clear visual feedback to aid in the decision-making.

Over the last couple of decades, there have been many developments in large classes of coordination algorithms and abstractions that support the required mapping from low-complexity, high-level commands to highly complex coordinated swarm behaviors [89]. Recent advances in coverage control [90, 91] provide an excellent approach to perform this mapping for formation control. The algorithms allow for a human operator to broadcast reference swarm spatial densities and boundaries in the robot domain that encode desired formations. The robots in the domain can then coordinate their motion with nearby robots to robustly achieve the commanded density distributions in real time in a scalable, distributed manner. In this chapter, we show through an array of human trials and simulations that

refining the entire state space is an effective approach for BCI swarm control, thereby demonstrating the potential and flexibility of our method for controlling high-complexity end effectors with low-complexity inputs.

3.2 Refining End Effector Behavior

To understand our interaction algorithm at a high-level, first consider the task of finding a word in an English dictionary. Anecdotally, one would commonly find that the user repeatedly bisects the remaining pages depending on whether their desired word comes before or after the current page. Our interaction algorithm is analogous to this efficient search procedure: the BCI user selects an effector behavior from an ordered dictionary of candidate behaviors through a sequence of bisections. Specifically, suppose that the BCI user learns a lexicographical ordering rule for the set of effector behaviors, which determines a total order of behaviors organized as a dictionary. At each round of interaction, the effector presents to the user the behavior that bisects the remainder of the dictionary. The user indicates to the effector (via a binary mental command) if their desired behavior precedes or succeeds the candidate behavior, and the dictionary scope is narrowed based on their reply. Rather than strict elimination of half of the dictionary at each step, the algorithm uses a probabilistic weighting over the dictionary to account for possible noise in the user's input (see Section A.3.1). Eventually, the user will have provided enough refinements for the end effector to correctly converge to the user's desired behavior. Importantly, this procedure does not involve the adjustment of individual effector parameters, but instead only requires the user to decide on the precedence of their desired behavior with respect to the current one. Although each dictionary bisection affects every effector parameter, the user only has to make a simple binary decision at each round, regardless of the number of effector parameters; this is distinct from a brute-force approach where the user adjusts each parameter individually.

While this interaction algorithm is intuitively satisfying, it is also endowed with rigorous

performance guarantees that become apparent when the entire interface is framed as a feedback communications system: the human user acts as a "transmitter" by encoding their desired effector behavior (the "message") through a sequence of binary BCI inputs ("codes"). These inputs are sequentially decoded by the end effector to refine a new estimate of the user's desired behavior, which is fully observable to the user as "noiseless feedback" and informs the choice of their next input. Because there is some chance that the user's binary input will be misclassified or that the user will make a decision error, the sequence of classification results can be modeled as outputs of a noisy binary symmetric channel (BSC) with a crossover probability equal to the misclassification probability. When framed as such a communications system, our interaction algorithm is mathematically equivalent to the posterior matching coding scheme [22]. Posterior matching is an optimal capacity-achieving code [27], meaning that this interaction algorithm communicates the user's desired behavior to the effector with as few binary inputs as possible for a given error rate.

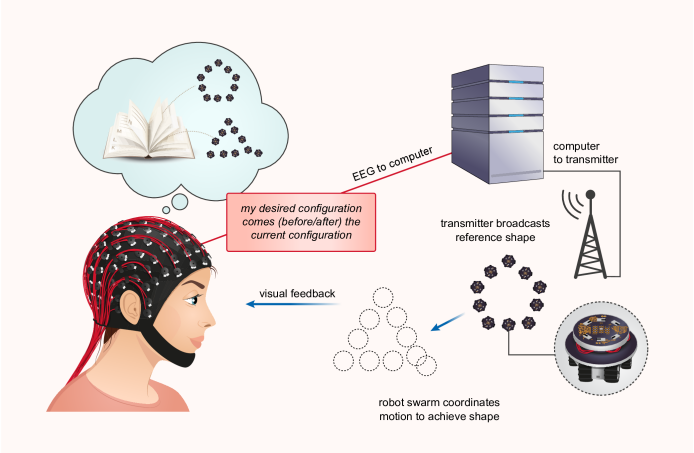
In previous work, posterior matching has been used as an interaction algorithm in noninvasive BCIs for tasks such as text entry or vehicle path planning [23, 52, 50, 51]. In these cases, a dictionary of ordered effector behaviors can be formed by constructing each dictionary element, or *string*, as a concatenation of *characters* from a fixed *alphabet*. For example, in text entry and path planning a string is constructed as a concatenation of English language letters and arc segments, respectively. In either of these cases, the precedence between two strings can be determined by identifying the first character that differs between the strings (referred to here as the *critical character*), and assigning precedence to the string whose critical character comes earliest in the character alphabet (e.g., 'a' precedes 'z,' arcs angled left precede arcs angled right). We refer to such dictionaries as *homogeneous* since in each case a single alphabet is used for all character positions in the behavior string. Tasks such as text entry or path planning can be adequately modeled by homogeneous dictionaries, since each additional effector parameter (e.g., letter or arc segment) is of the same type.

Unlike the tasks described above, many high-complexity effectors cannot be described

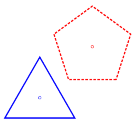
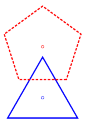
with homogeneous dictionaries by concatenating characters from a single alphabet. For example, in robot swarm control, each swarm configuration is characterized by varied parameters describing position, shape, and size. To model these high-complexity effectors we design a *heterogeneous* dictionary, where a different alphabet is used for each character position in the behavior string. To our knowledge, posterior matching has not been deployed as an interaction algorithm using heterogeneous dictionaries, and it was previously unknown if BCI users can successfully learn and apply a heterogeneous dictionary to posterior matching control of a high-complexity effector. As we detail below, we demonstrate in a large-scale interface study that people can learn such a heterogeneous dictionary with little training and make pairwise string comparisons with high proficiency.

While one might conceive of a variety of heterogeneous dictionaries to describe swarm configurations, here we adopt a dictionary of regular polygons as a proof of concept. Each polygon string is parameterized by characters including horizontal position, vertical position, number of sides, and size, with distinct alphabets for each character position (Figure 3.2b). To search this polygon dictionary with posterior matching, the BCI user issues hand motor imagery (MI) inputs detected via EEG measurements to indicate if the desired behavior comes before or after the currently demonstrated behavior in the dictionary. MI tasks are a well-studied and popular binary input modality where the user mentally visualizes wrist flexions of either their left or right hand and the resulting changes in EEG frequencies are detected by a binary classifier [92, 93]. To refine the swarm, the user determines the first character where their desired configuration differs from the current configuration and issues a left-hand (right-hand) MI input if their desired polygon preceded (succeeds) the current polygon at the critical character. As the complexity of the dictionary increases, the sequential scan to find the critical character may take marginally more time, but the decision by the user is ultimately based only on a simple evaluation of that character (despite each user input potentially updating all characters). Note that this approach is not limited to EEG-based MI, and is compatible with any binary input mechanism including inputs

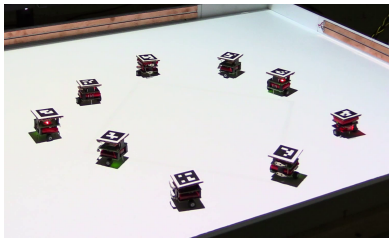
detected by invasive BCIs. We refer to this combination of a heterogeneous swarm dictionary with binary input posterior matching as SCINET: Swarm Control via Interactive Neural Teleoperation (illustrated in Figure 3.2a).



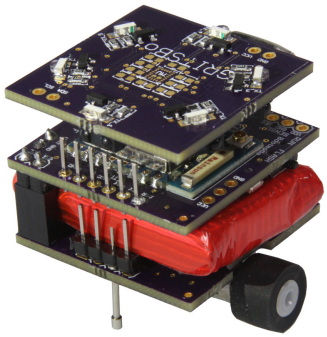
(a)

Alphabet description	**1st character:** horizontal position	**2nd character:** vertical position	**3rd character:** number of sides	**4th character:** size
Alphabet ordering	**left** precedes **right**	**below** precedes **above**	**fewer** precedes **more**	**smaller** precedes **larger**
Example				

(b)



(c)



(d)

Figure 3.2: **Refining effector behavior through configuration sorting.** Effector behavior is determined through iterative refinement from the BCI user. **a,** In the example of robot swarm configuration refinement, the BCI user indicates through a mental command (e.g., binary motor imagery) if their desired configuration comes before or after the current configuration in the swarm dictionary (see **b**). A computer decodes the input by classifying scalp electrode recordings from the user, and updates a posterior distribution over the configuration dictionary. The median of the updated distribution is selected as a new configuration guess and transmitted to the swarm through a global update. Each individual robot then adjusts its position locally so that the overall configuration conforms to the new guess in a distributed manner. **b,** In the swarm configuration dictionary, character alphabets are defined in order from first to last as follows, with alphabet precedence in parentheses: horizontal position of the configuration center (centers to the left preceding centers to the right); vertical position of the configuration center (centers below preceding centers above); number of sides (fewer sides preceding more sides); configuration size as the radius from the center to each vertex (smaller radii preceding larger radii). Each example panel depicts a pair of strings whose critical character corresponds to the panel column, with blue (solid) configurations preceding red (dashed) configurations in the overall dictionary ordering. **c,** Example of a robot swarm coordinating to form a globally specified configuration by using only local information. **d,** Close-up view of an individual mobile robot used in our demonstrations. A robot swarm (as in **c**) consists of several such robots collectively performing global actions in a distributed manner.

3.3 Dictionary Sorting Proficiency

Although in theory SCINET is capable of controlling an arbitrary number of degrees of freedom (i.e., string characters), this scalability is limited in practice by the ability and ease by which the BCI operator can sort strings according to the swarm dictionary ordering. A typical human user should be able to quickly learn the swarm dictionary and subsequently sort any pair of strings, with high proficiency when the critical character is located at any position in the string. To evaluate these user capabilities in an isolated manner from the rest of the BCI system, we conducted a user study where participants ($n = 150$) used a point-and-click interface to select between configurations on a screen (study details in Section A.2). Each participant was first presented with a set of graphical and text instructions explaining the polygon dictionary ordering and how to use it to sort a given string pair. Each participant was then presented with 150 randomly selected shape pairs from the dictionary (Figure 3.3b), and asked to indicate which shape precedes the other in the dictionary ordering. We provided each participant with a visual aid to use as a reference during the task (Figure 3.3a); such an aid could also be presented to a BCI operator in a practical setting.

Overall, participants were able to sort shape pairs with high accuracy. When evaluating sorting accuracy over all pairs of strings (Figure 3.3c), most subjects sorted with nearly perfect accuracy (median 99.3% accuracy). Furthermore, response accuracy does not appear to decrease in the aggregate as the position of the critical character appears later in the string (median 100% accuracy for all characters, see Figure 3.3d). When evaluating critical character performance for each individual participant, we also find that most participants exhibit non-decreasing or only modestly decreasing performance as character depth increases (Supplementary Figure A.8). Although a user's capacity for learning and memorizing a dictionary ordering may create a performance bottleneck, this can be mitigated by providing users with a mnemonic aid to assist in recalling the ordering, as was done in our study. These results suggest that users can rapidly learn and apply string sorting in our heterogeneous

dictionary, and that adding more characters (i.e., effector parameters) does not hinder a user's ability to effectively compare pairs of strings across multiple parameters simultaneously.

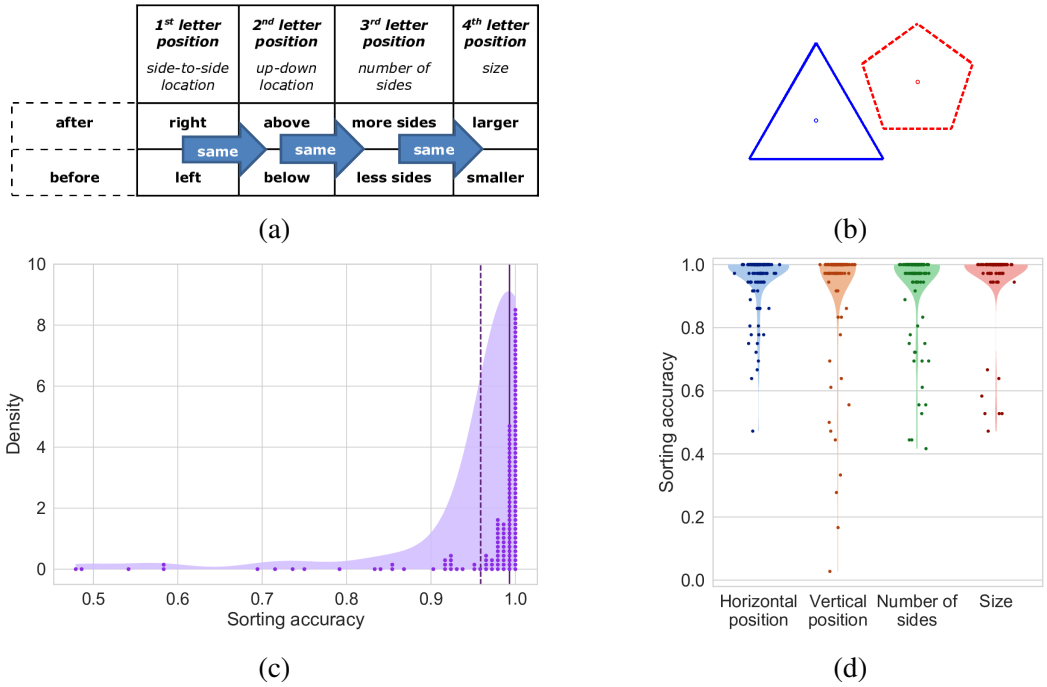


Figure 3.3: **Evaluating configuration sorting proficiency in a user study. a,** Mnemonic aid for dictionary ordering recall using plain language, provided to each user study participant ($n = 150$). **b,** Example shape pair presented to each participant; in this case, the correct sorting is that the blue (solid) triangle precedes the red (dashed) pentagon, since horizontal position is the critical character and the triangle's center is located further to the left. **c,** Estimated distribution of overall dictionary sorting accuracy across all participants, where each participant is represented by a dot. The vast majority of participants were able to correctly sort configurations with high accuracy, with a mean accuracy (dashed vertical) of 95.8% and a median accuracy (solid vertical) of 99.3%. **d,** Dictionary sorting accuracy for each critical character. Each column shows accuracy across all participants (represented as dots, with added horizontal jitter for visual clarity) when calculated only for queried shape pairs with the respective critical character. As the depth of the critical character position increases, sorting ability does not decrease, as exhibited by a median accuracy of 100% for all characters. These results support the scalability of sorting heterogeneous dictionary strings as an interaction mechanism, since accuracy does not decrease as additional characters are added to the dictionary.

3.4 Full System Evaluation

Beyond the interaction algorithm, there are a number of additional factors which can affect SCINET performance in the full system. Namely, the user must not only compare the current swarm configuration against their target string in the dictionary ordering, but must then issue a binary input via a mental command and subsequently observe the real-time changes in the swarm's behavior. Due to practical effects such as user fatigue, the user's error in issuing inputs may stray from the theoretical BSC assumed by the posterior matching algorithm. Since posterior matching assumes a fixed BSC crossover probability, it is unclear if non-ideal input statistics will result in poor system performance, and if such effects can be modeled. To evaluate SCINET in practice, we measure accuracy of a physical SCINET implementation against a simulation model that accounts for these practical effects.

As a pilot demonstration, GC trained an EEG MI classifier and used the rules of posterior matching to control a simulated robot swarm (presented visually on a monitor) (Figure 3.4b). In a series of repeat trials, target configurations were presented and MI commands were issued to steer the swarm towards the specified configuration (details in Section A.2). As one might anticipate, over the course of issuing a sequence of MI inputs, the error rate of user inputs (calculated with respect to the correct input according to the rules of posterior matching) varied as additional commands were issued (Figure 3.4c). In theory, this error can be attributed to both the user error of issuing the incorrect posterior matching input, as well as classification error due to the MI detection algorithm classifying the input incorrectly. We conclude from the previous dictionary sorting user study that the former error source is small (estimated at 4.2%, see Figure 3.3c), and therefore the increasing net input errors are likely due to degrading MI signal feature separation (Supplementary Figure A.12). This effect is possibly due to user fatigue in issuing a large number of inputs with minimal training, and resulted in an overall input error of 21.8%. Given previous work on MI inputs, this error rate is likely to be significantly improved with higher-fidelity interfaces and more extensive

user training [94].

Nonetheless, an overall configuration selection accuracy of 75.7% was achieved (Figure 3.4d), calculated as the fraction of trials where the swarm converged perfectly to the specified target with zero error; this greatly exceeds the accuracy of 1.67% that would be obtained by chance selection alone. Furthermore, we can account for these observed results with a simple model on the non-stationary input statistics. We fit a piecewise polynomial to the empirical crossover probability (Figure 3.4c, see Section A.2) and use this profile to generate input errors in a posterior matching simulation that assumes a fixed crossover probability. This simulation model obtains a similar configuration accuracy (74.3%) to that observed in practice (Figure 3.4d). Additionally, this model matches the observed behavior even when evaluating trials based on their required numbers of inputs to converge, which is a distinguishing element between trials since longer convergence is associated with increasing input errors and, therefore, with decreased performance.

GC also demonstrated SCINET's capability to be implemented in a (non-virtual) cyber-physical system by successfully steering a physical robot swarm in multiple trials as a proof-of-principle to complement the virtual simulations of swarm behavior (Figure 3.4a). Taken together, these results collectively demonstrate that SCINET can achieve reasonable configuration accuracy despite the presence of non-stationary input errors, and that performance can be captured by a simple model. Additionally, the availability of a simulator that closely matches observed empirical behavior allows us to explore the performance of SCINET with more general dictionaries.

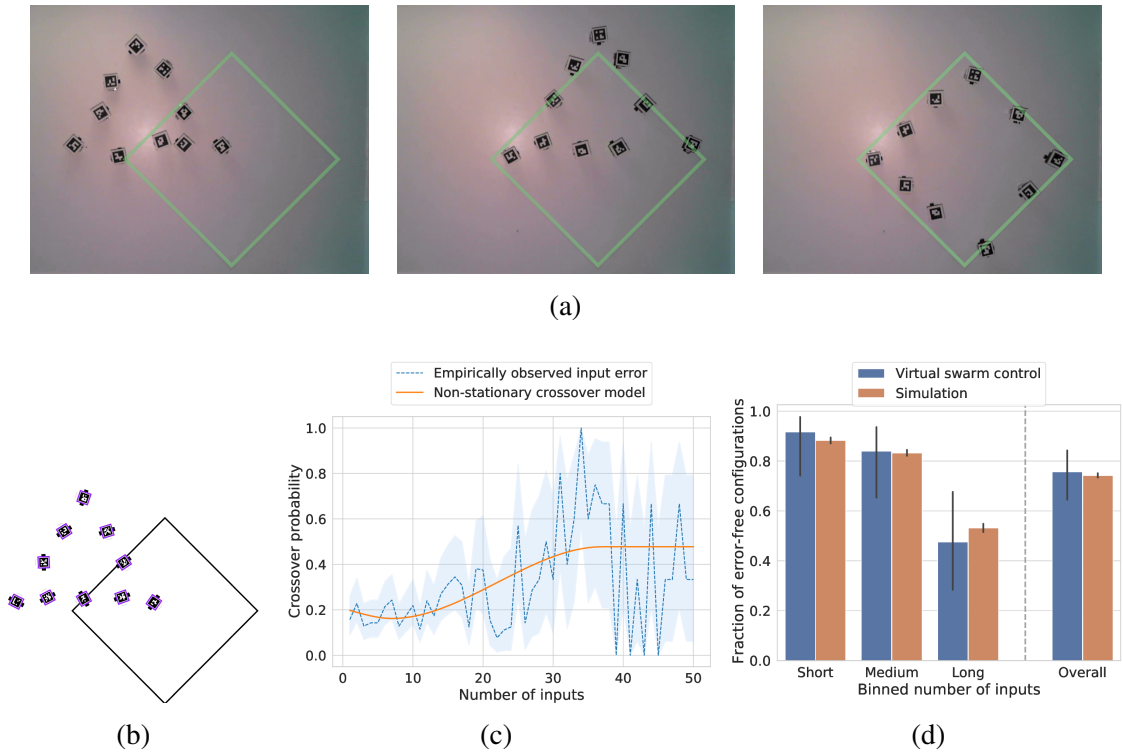


Figure 3.4: **End-to-end testing of full system with EEG inputs and swarm control.** SCINET was tested as a full system by GC for controlling both physical (**a**) and virtual (**b**) robot swarms. **a,** When controlling physical robots, the target configuration is presented to the user as an illuminated shape on the robot arena (accentuated here for visibility). **b,** During virtual swarm control, the BCI user views a monitor that presents a target configuration (depicted as a shape outline) alongside the swarm's current configuration. The virtual robots simulate realistic robot motion, and readjust their positions dynamically after each user input. **c,** Non-stationary crossover probability versus number of inputs in virtual swarm control trials ($n = 70$). At each input, we estimate the empirical crossover probability (blue, dotted line, with 95% Wilson confidence interval [95]) as the fraction of trials where that input was decoded incorrectly with respect to the target configuration. A modified cubic function was fit to this changing crossover probability (orange, solid line) and used to generate errors in a realistic SCINET simulation (see Section A.2). **d,** Comparison of experimental and simulation configuration accuracy as a function of number of inputs until convergence, where the non-stationary crossover profile from **c** was used for simulating input errors. Results are binned into short, medium, and long trials by selecting bin edges at the 1/3 and 2/3 percentiles of virtual swarm convergence times, such that each bin includes approximately the same number of trials. Configuration accuracy within each bin is computed as the fraction of trials ($n = 10,000$) converging successfully, where error bars depict the 95% Wilson confidence interval. The overall configuration accuracy across all trials (regardless of number of inputs to converge) is also depicted. Experimental accuracy (binned and overall) closely matches that of the simulated model, suggesting that posterior matching (which assumes a fixed crossover probability) with input errors generated by the profile in **c** is an appropriate model for the observed experimental behavior. This suggests that the end-to-end system is performing as expected (once the input statistics are accounted for), and that it is reasonable to use this simulator to further explore system behavior.

3.5 Generalizing Performance Tradeoffs

Ultimately, the accuracy and number of controllable degrees of freedom (and hence the dictionary size) in SCINET is determined by the error rate of the input mechanism and budget on the allowable number of inputs; increasing the controlled degrees of freedom requires additional inputs to refine effector behavior. To more fully explore this tradeoff, we use different input error profiles and dictionary sizes (corresponding to a variety of end-effector degrees of freedom) to simulate posterior matching as well as a baseline interaction algorithm (called *stepwise search*) that resembles discrete menu selection in existing BCIs. In stepwise search, each binary input updates the swarm's guessed configuration by moving to the next string in the dictionary, in the direction indicated by the user's input.[2] Note that the number of steps needed for convergence in stepwise search scales linearly with the size of the dictionary.

In the data collected from a simple interface (with input characteristics reported in Figure 3.4c), our proposed interaction mechanism can work well in some scenarios despite the relatively high overall error rate and the non-stationary error profile that nears chance probability (50%) as the number of inputs increases. However, large dictionaries providing more resolution will suffer a performance bottleneck with this non-stationary error profile because they require more inputs for convergence. While developing high-performance input mechanisms is not the focus of this work, we evaluate SCINET system performance with realistic improved input mechanisms by simulating both posterior matching and stepwise search for a fixed crossover probability of 10% (comparable to input errors seen in prior work [94]) and a variety of dictionary sizes (expressed as an equivalent number of degrees of freedom by subdividing each dictionary with an average alphabet size from our physical system — see Section A.2). The *information transfer rate* (ITR) [53] (specified in bits per trial) is shown in Figure 3.5a, demonstrating that SCINET, with this simulated input

[2]This algorithm is similar to the *fixed offset* [96] and *sequential-select* [51] policies explored in previous work on PM-based BCIs.

mechanism, can achieve increasingly high information rates with larger dictionaries. The fraction of error-free configurations (i.e., perfectly achieving the desired configuration) also approaches 100% (Figure 3.5b), even with large dictionaries and non-zero error rates in the user input. Finally, to study the rate of convergence of the estimated configuration to the target in the dictionary (which is not reflected in the fraction of error-free configurations), we also measure the absolute deviation of the estimated configuration from the target and observe that error decays quickly regardless of dictionary size (Figure 3.5c, see Section A.2).

In all metrics, posterior matching vastly outperforms discrete menu selection through a stepwise search approach. While larger dictionaries require more inputs to refine a configuration to a desired level of accuracy, posterior matching with a fixed crossover probability still achieves high performance for large dictionaries in a modest number of inputs. We note that with a fixed input profile, posterior matching can successfully control upwards of 6 separate degrees of freedom, which (to our knowledge) exceeds the current capabilities of noninvasive continuous control BCIs. We also plot the same performance metrics for simulations where input errors are generated according to the non-stationary profile observed in our physical experiments (Figures 3.5d to 3.5f). Even with this adverse input characteristic, posterior matching greatly outperforms stepwise search across all metrics. Although performance degrades for larger dictionary sizes, these larger dictionary sizes correspond to estimated degrees of freedom that lie beyond the control capabilities of typical noninvasive BCIs.

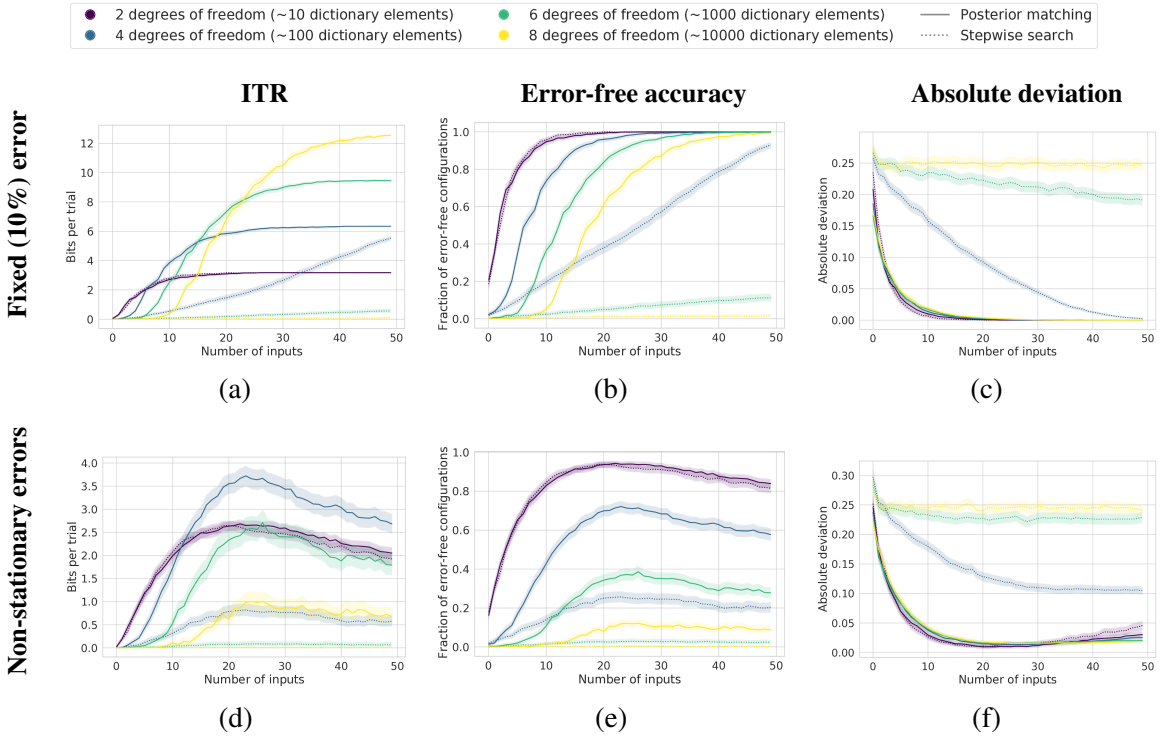


Figure 3.5: **Performance as a function of number of inputs and dictionary size.** We simulated the proposed posterior matching approach (solid lines) and *stepwise search* akin to discrete menu selection (dotted lines) over various dictionary sizes corresponding to different (estimated) end effector degrees of freedom (see Section A.2). We evaluate both algorithms over two input error profiles: a fixed 10% crossover probability similar to prior reported decoding performance (**a-c**), and more adverse non-stationary errors generated according a model of our physical experiments (**d-f**). **a,d,** Information transfer rate (ITR) [53], measuring the amount of information specified by a trial's inputs with respect to dictionary size. Error bars are calculated as the ITR of the corresponding accuracy limits in **b** and **e**. **b,e,** Fraction of estimated configurations that are a perfect match with the target configuration, with 95% Wilson confidence intervals. **c,f** Absolute deviation (dictionary distance) between estimated and target configurations, where error bars depict 95% bootstrap confidence intervals over 10,000 samples (separate resampling for every number of inputs). By all metrics (**a-f**), posterior matching greatly outperforms stepwise search, with differences in performance becoming more drastic for larger dictionary sizes. Additionally, posterior matching obtains high configuration accuracies at high (> 4) estimated degrees of freedom with a modest number of inputs.

3.6 Discussion

The results in this chapter demonstrate how our interaction algorithm significantly expands the capabilities of low-complexity BCIs to efficiently, robustly, and scalably control high complexity effectors, while requiring no more than currently available signal acquisition hardware already in widespread development and use. The success of human users in learning and sorting a heterogeneous shape dictionary supports the use of pairwise string sorting as a simple-to-use and tractable interface design that scales well with the complexity of the end effector system. When tested in a physical system, SCINET can perform well despite the presence of non-stationary input errors, validating the deployment of posterior matching control over a heterogeneous dictionary in a practical setting. By extending our experimental results to a range of dictionary sizes and input mechanism fidelities through realistic simulations, we find that posterior matching both outperforms a baseline algorithm comparable to discrete menu selection and exhibits the ability to control a large number of estimated degrees of freedom with only a modest number of inputs.

While posterior matching with a heterogeneous dictionary was implemented here for the control of robot swarms, the general technique is applicable to any setting where each effector parameter can be assigned its own ordered alphabet. Importantly, our approach has the flexibility for a system designer to select a dictionary size based on their effector's behavioral specifications such as allowable number of user inputs, minimum configuration accuracy, and number of effector parameters (i.e., degrees of freedom). Once the designer decides on a fixed number of dictionary elements, they can then distribute this fixed number of elements among their degrees of freedom in a customized manner by tuning the size of each character's alphabet, allowing for variable resolutions between parameters. More generally, by iteratively refining effector behavior through a sequence of low-complexity inputs rather than requiring a single high-fidelity measurement to instantaneously extract a total system state from the BCI user, SCINET complements years of research devoted to

improving the input mechanisms of BCIs by instead fundamentally redesigning how inputs are utilized.

CHAPTER 4

INTERACTIVE OBJECT SEGMENTATION WITH NOISY BINARY INPUTS

4.1 Interactive Image Segmentation

In this chapter,[1] we take a similar approach as in SCINET to design a one-bit interactive object segmentation system that is simple, robust, and human-implementable. Interactive image segmentation is a task where users specify regions of interest in an image by providing limited, low-complexity inputs with dynamic feedback from the segmentation program. Such low-complexity inputs are commonplace in many human-computer interfaces since they can be easier to use than traditional mechanisms and are sometimes necessary due to user restrictions. For instance, low-complexity inputs are beneficial for users who may wish to interact with a computer system in a hands-free manner such as in sterilized operating rooms [97] or to prevent repetitive strain injuries [98]. Furthermore, users suffering from paralysis or neurological diseases sometimes require alternate means of communication that provide low-complexity inputs and do not rely on traditional muscular mechanisms [53]. For example, such input devices might involve brain-computer interfaces [99], input switches activated by gross motor movements [100], or command entry through non-traditional motor commands such as the tongue drive [101]. From a mathematical perspective, using such an HCI for interactive segmentation is a challenging communications problem since such a system should allow users to indicate segments with high specificity, utilize provided information in an efficient manner, and remain robust to noise in the input (e.g. interface noise, user errors, etc.). In typical communications systems, messages can be conveyed via noisy symbols from low-complexity alphabets with negligible recovery error by employing

[1] This chapter is in collaboration with Sivabalan Manivasagam, Shaoheng Liang, and Dr. Christopher Rozell. GC supervised the low-level details of the project including algorithm and experiment design, and conducted the data analysis. SM and SL assisted in algorithm design and implemented the experiments. CR supervised the project. GC was the lead author of this work, which appears in [2].

complex error correcting codes [27]. However, in the interactive segmentation setting where the 'message' is a segment and the 'encoder' is a human, the utilized encoding scheme must instead be simple and human-implementable.

Approaches to interactive image segmentation systems with these desired traits have varied in the type of input provided and how it is processed [102, 103, 104, 105, 106, 107, 108, 109, 110]. However, there are few examples of approaches developed for the important special case of noisy binary inputs that correspond to many alternative or augmented communication systems. The first segmentation method to use binary inputs is a variant of a twenty questions style game, which we refer to as "N-Questions" and use as a baseline algorithm [111]. In this setting, the user is sequentially queried for a binary response indicating whether a specified pixel is within their desired region, from which the segmentation is updated and the next question presented. While N-Questions performs well over several datasets that emphasize selection of arbitrary regions in a given image, it is not as well suited to the task of specific object segmentation (as opposed to general image segmentation). In the context of real-world HCI systems, arbitrary segmentations that merge multiple objects into composite foreground regions may have limited utility compared to specifying discrete objects for meaningful interaction. Other algorithms using such binary membership queries for specifying arbitrary segmentations include methods based on transductive inference with pixel [112] or superpixel [113] queries, and an extension of N-Questions to voxel queries in three-dimensional space [97].

The main contribution of this chapter is to propose a novel interactive image segmentation algorithm titled EllipseLex that is specifically designed for segmenting real-world objects using noisy binary inputs using the principles of feedback information theory. Rather than creating segmentations directly from user inputs (e.g. bounding boxes, foreground/background seeds), our method builds on previous work in brain-computer interfaces [23, 52] to allow a user to efficiently choose an explicit segment from a large repertoire of possible segments designed to provide a broad approximation of the object space. This set

of possible segments consists of ellipses in various positions, rotations, sizes, and aspect ratios, since ellipses can serve well as bounding boxes around common objects while simultaneously qualifying as inputs into post-processing steps such as the popular GrabCut segmentation algorithm [106] to refine the segment border as demonstrated in Figure 4.1. By assigning an order to this set (or *lexicon*) of ellipses, an algorithm derived from new results in feedback information theory can be used to specify the desired ellipse in an optimal manner and makes our method, to the best of our knowledge, the first binary-input segmentation algorithm to directly model and compensate for input noise. The empirical results of this work demonstrate improved performance of the proposed lexicon at efficient object segmentation with simulated users under a variety of experimental conditions.

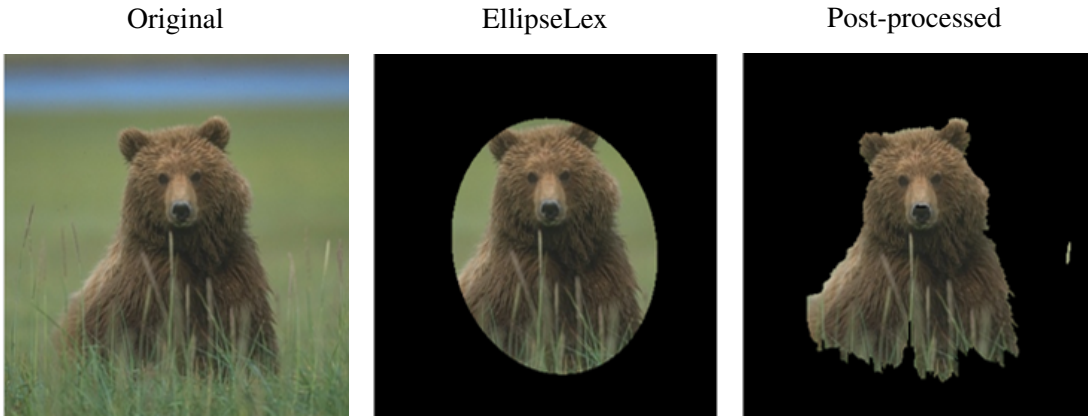


Figure 4.1: **EllipseLex with post-processing.** *source image (left), result of EllipseLex on bear segment (center), EllipseLex with GrabCut [106] post-processing (right)*

4.2 Methods

At its essence, the task of specifying an ellipse (denoted by $\mathbf{z}^*$) in an ordered lexicon using only noisy binary inputs can be abstracted as a communications system between a human user and a computer, where the user encodes this ellipse using a sequence of inputs[2] $X_k \in \{0, 1\}$. Each X_k passes through a BSC such that the output symbol $Y_k \in \{0, 1\}$

[2]In this chapter, X is used to indicate channel inputs rather than L, in order to remain consistent with the chapter's original publication [2].

experiences a bit flip with some crossover probability $p \in [0, 0.5)$. From this output, the computer guesses, or decodes, an ellipse $\hat{\mathbf{z}}_k$, which is then provided to the user as noiseless visual feedback. With this setup, we can directly apply PM[3] as an interaction algorithm, with guaranteed convergence to the desired ellipse $\mathbf{z}^*$ even in the presence of channel noise [22]. Intuitively, the algorithm can be described as a guessing game between the user and computer; at each time step, the computer guesses the median of the posterior distribution over the segments (conditioned on previous inputs) as the user's selected ellipse, and the user then issues a binary input to inform the computer if their desired ellipse (which should approximate their desired object) comes before or after the guessed ellipse according to the ordering of the lexicon. From this input (which may be corrupted by noise), the computer applies a Bayesian update to the segment posterior and presents an updated guess to the user, thereby repeating the cycle. As the user and computer iterate in this manner, the posterior distribution converges to a single mass at the user's desired segment [32].

To apply this coding technique, a finite, ordered lexicon of ellipses must be constructed: an ellipse *word* in this lexicon is described by the tuple $\mathbf{z} = (y, x, \theta, a, r)$ denoting the ellipse's vertical position, horizontal position, angle from the horizon, half-length of the major axis, and aspect ratio of the minor axis to major axis. This lexicon is ordered by comparing the first letter (i.e. $y, x, \theta, a,$ or r) that differs between two ellipse words in question, in the same manner that words are alphabetized in English. Just as *a* precedes z in the English alphabet, each ellipse letter's *alphabet* has a precedence rule stated in the upper section of Figure 4.2. Each alphabet's order of precedence was selected based on the authors' preferences, but any individual alphabet's ordering could be reversed based on a user's preferences, without any changes in the algorithm's performance. Our algorithm in its entirety, referred to as "EllipseLex," is presented in Algorithm 1. At time step k, ellipse mask $\hat{\mathbf{z}}_{k-1}$ is presented to the user as feedback, which they observe to choose the subsequent binary input as illustrated by the examples in Figure 4.2.

[3]Since we are technically searching over a discretized lexicon of ellipses, the BZ algorithm is used here rather than the probabilistic bisection algorithm.

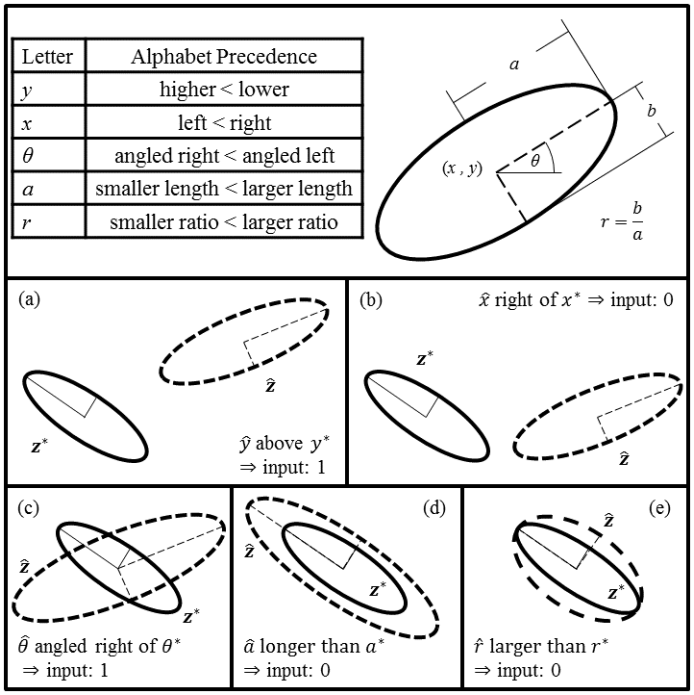


Figure 4.2: **Description and examples of ellipse lexicon.** *The user finds the first letter that differs between the target ($\mathbf{z}^*$) and guess ($\hat{\mathbf{z}}$) by moving down the rows of the 'Alphabet Precedence' table, and then issues an input according to Algorithm 1 as demonstrated in boxes (a) through (e). For instance, in (d) the vertical and horizontal positions as well as the angle are aligned, but the guess's major axis is longer than that of the target, so the target precedes the guess in the lexicon and therefore a 0 should be issued. This type of refinement continues until a guess is produced that matches the target ellipse.*

4.3 Results

We compare the performance of EllipseLex to that of N-Questions using simulated ideal users with fixed levels of input noise. In particular, we examine the convergence of guessed segments to ground truth objects with respect to number of inputs, ground truth segment size, object class, and noise level (i.e. BSC crossover probability). The injected input noise used here lies within the typical range of crossover probabilities in binary-input HCI systems [94] and simulates the compounded effect of input mechanism error with human errors or bias. To quantitatively measure a segmentation method's ability to specify a ground truth region we use the popular F1 score, which lies in the interval $[0, 1]$, with low scores corresponding to poor segmentation performance and high scores indicating exceptional

Algorithm 1: EllipseLex

Input: target ellipse mask $\mathbf{z}^* = (y^*, x^*, \theta^*, a^*, r^*)$
  $\hat{\mathbf{z}}_0 \leftarrow$ initial ellipse guess
  for $k \leftarrow 1$ **to** K **do**
    if $\hat{y}_{k-1} \neq y^*$ **then**
      compare vertical position y
    else if $\hat{x}_{k-1} \neq x^*$ **then**
      compare horizontal position x
    else if $\hat{\theta}_{k-1} \neq \theta^*$ **then**
      compare angle θ
    else if $\hat{a}_{k-1} \neq a^*$ **then**
      compare major axis length a
    else
      compare aspect ratio r
    end if
    let ρ represent the letter to be compared
    if $\rho^* \geq \hat{\rho}_{k-1}$ **then**
      $X_k = 1$ input 1
    else if $\rho^* < \hat{\rho}_{k-1}$ **then**
      $X_k = 0$ input 0
    end if
    $Y_k = BSC(X_k, p)$ BSC with crossover p
    $\hat{\mathbf{z}}_k = PM(Y_k)$ update posterior, return median (Section A.1.1)
  end for
Output: $\hat{\mathbf{z}}_K$

performance [114, 111, 109, 110]. We tested our method on the MS-COCO validation dataset, which contains over 280,000 segmentations for 80 object classes (e.g. dog, chair, cup) in both indoor and outdoor settings [115], allowing for a robust testing of each method's performance in object segmentation. Figure 4.3 illustrates example segments produced by EllipseLex in comparison to N-Questions.

To test these algorithms in simulation, ground truth regions from each image serve as proxies for a human user's intended objects. In N-Questions, this ground truth is used directly as an oracle to answer queries regarding pixel inclusion in the object segment. However, for the sake of simulation EllipseLex operates on a target ellipse mask $\mathbf{z}^*$, which is selected from the lexicon to approximate a given ground truth segment through the use of a heuristic as follows: first, the centroid of a given ground truth segment is calculated as

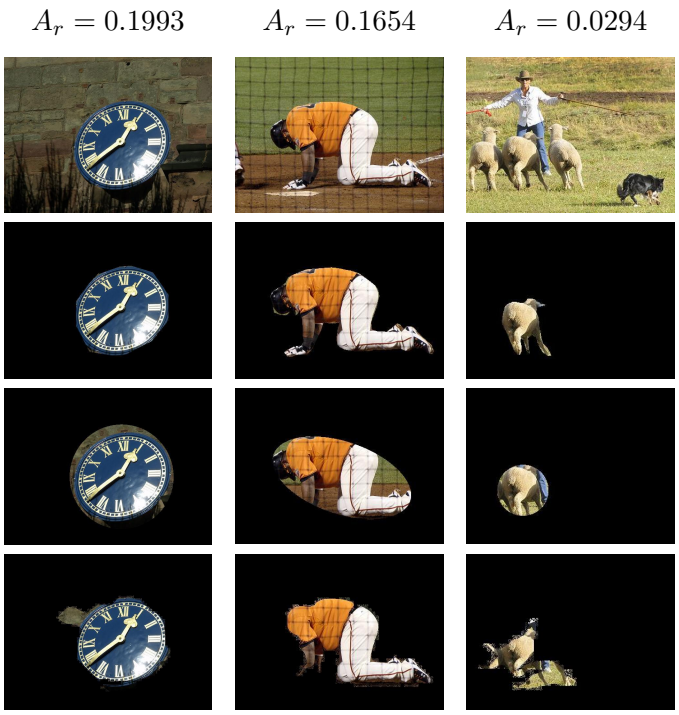


Figure 4.3: **Comparison of segmentation methods.** *Row 1: source image [115]. Row 2: ground truth segmentation. Row 3: EllipseLex mask after 30 noiseless inputs. Row 4: N-Questions mask after 30 noiseless inputs. The relative area (A_r) of each ground truth segment to its source image is displayed above each image column.*

(x_c, y_c). Then, the vertical and horizontal coordinates of the target ellipse are estimated as

$$y^* = \arg\min_{y \in \Sigma_y} |y - y_c| \quad x^* = \arg\min_{x \in \Sigma_x} |x - x_c|, \tag{4.1}$$

where Σ_y and Σ_x denote the vertical and horizontal position alphabets, respectively. After y^* and x^* are selected, $\theta, a,$ and r are set by conducting a brute-force search over their respective alphabets $(\Sigma_\theta, \Sigma_a, \Sigma_r)$ and selecting the ellipse that maximizes F1 score with respect to the given ground truth. More precisely, for ground truth segment s,

$$(\theta^*, a^*, r^*) = \arg\max_{\theta \in \Sigma_\theta, a \in \Sigma_a, r \in \Sigma_r} \mathrm{F1}_s(y^*, x^*, \theta, a, r), \tag{4.2}$$

where $\mathrm{F1}_s(y, x, \theta, a, r)$ calculates the F1 score for an ellipse mask parameterized by (y, x, θ, a, r) with respect to a ground truth segment s. In practice, a human user steers the guessed ellipse

Table 4.1: **Mean F1 score vs. number of inputs.** *At each given input, the F1 score is calculated for the guessed segment produced by EllipseLex-Low, EllipseLex-High, and N-Questions, and averaged over all trials respectively for each method. Results are shown for 0%, 5%, and 10% crossover probabilities in the input. The maximum of each column at each noise level is displayed in bold, and all standard errors of the mean are less than 0.001.*

Method	Noise	$K = 10$	$K = 20$	$K = 30$
EllipseLex-Low	0%	**0.2050**	**0.5927**	0.5949
EllipseLex-High	0%	0.1886	0.4400	**0.7487**
N-Questions	0%	0.1462	0.2112	0.2569
EllipseLex-Low	5%	**0.1720**	**0.4472**	**0.5685**
EllipseLex-High	5%	0.1353	0.2482	0.5518
N-Questions	5%	0.1269	0.1747	0.2064
EllipseLex-Low	10%	**0.1379**	**0.3021**	**0.4616**
EllipseLex-High	10%	0.0970	0.1766	0.3344
N-Questions	10%	0.1145	0.1476	0.1686

to fit their desired segment instead of performing intermediate estimation of a target ellipse. We tested two lexicons corresponding to finer or coarser alphabet sizes (m_y, m_x, m_θ, m_a, m_r) of (15, 20, 10, 20, 10) and (100, 100, 20, 20, 10), referred to as "EllipseLex-Low" and "EllipseLex-High" respectively.

In Table 4.1, mean F1 score from all ground truth regions in MS-COCO is depicted at select numbers of inputs for all methods across all noise levels. Overall, the EllipseLex methods outperform N-Questions for increasing numbers of inputs over all noise levels. A tradeoff between low and high resolution lexicons is evident, with EllipseLex-Low exhibiting faster F1 growth and higher noise resilience in comparison to EllipseLex-High over all noise levels. This is due to the fact that larger lexicon resolutions require more inputs for convergence under noise according to the PM algorithm. However, in the noiseless case EllipseLex-High achieves a significantly higher final F1 score, suggesting that the choice of lexicon parameterization should be determined with respect to the number of desired inputs as well as the noise level.

Next, the F1 score achieved after 30 inputs is compared across methods and noise

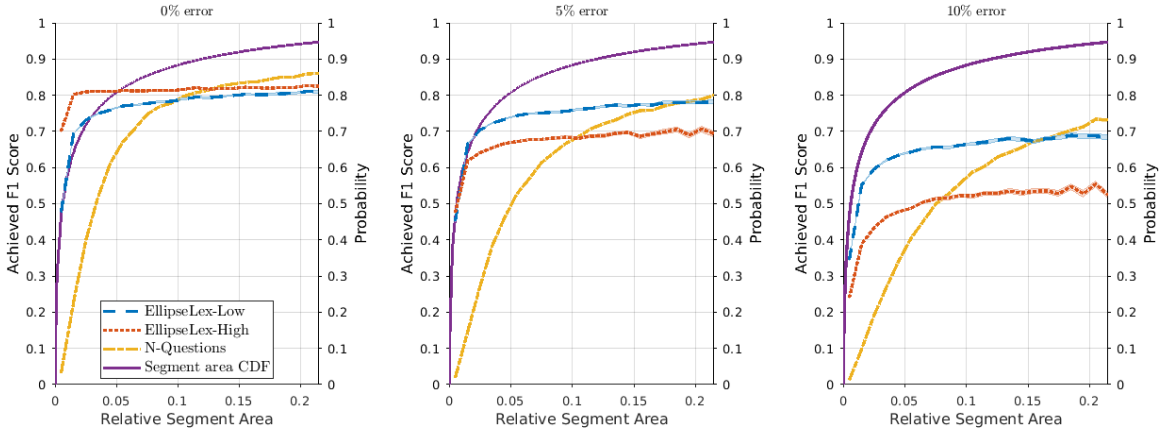


Figure 4.4: **Final F1 score vs. relative segment area.** *Average achieved F1 score after 30 inputs is plotted against relative segment area for all three methods (left axis) along with the empirical cumulative distribution function of relative segment areas (right axis). Relative segment area is calculated as the area of a ground truth segment divided by the area of its source image. Averages here are taken over trials of segments in the same relative segment area bin, with bin sizes of 0.01. Only relative segment areas below the 95th percentile are displayed. Error bars consisting of ± 1 standard error are displayed.*

levels with respect to the relative area of each ground truth segment (segment area / image area). As depicted in Figure 4.4, the EllipseLex methods outperform N-Questions for segments with relative areas of 0.1 and less, a set which comprises approximately 90% of the segments in MS-COCO. While N-Questions demonstrates superior performance for segments with relative area more than approximately 0.15, these segment sizes are rare for real-world objects in the MS-COCO database and the EllipseLex methods do not degrade in performance with increases in area. These trends are evident across all noise levels.

Finally, when final F1 score is plotted against the 80 object classes in MS-COCO in Figure 4.5, the EllipseLex methods appear to achieve improved performance. While all three methods achieve comparable performance for some classes (e.g. “bed”), the EllipseLex methods significantly outperform N-Questions even in the worst performing object classes (e.g. “skis”). This supports the notion that EllipseLex methods are well suited to object selection across multiple everyday object classes. Note that EllipseLex-High achieves higher F1 scores than EllipseLex-Low uniformly across all classes.

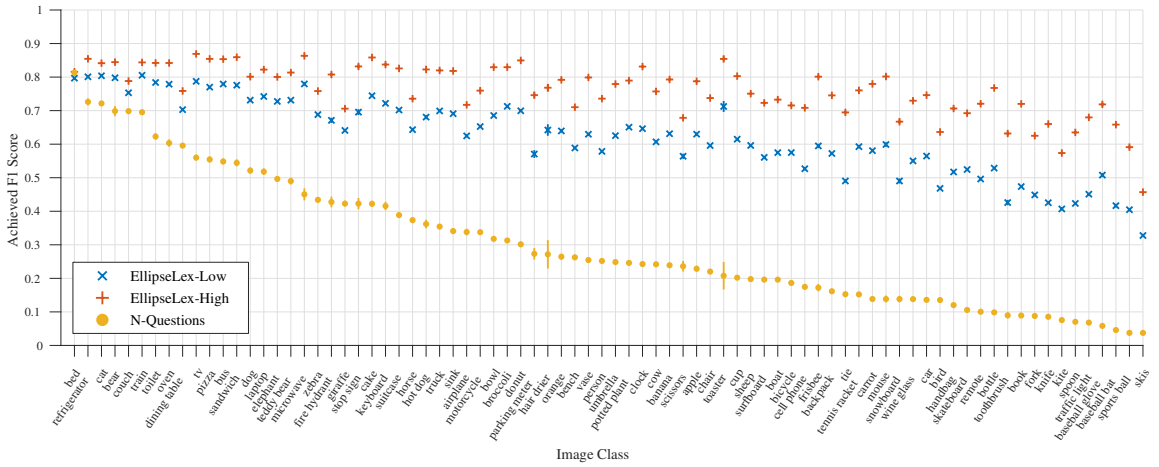


Figure 4.5: **Final F1 score vs. object class.** *Average final F1 score after 30 inputs is plotted against ground truth object class sorted in descending order of N-Questions performance, for the noiseless case only. Averages here are taken over trials with ground truth segments in the same object class. Error bars consisting of ± 1 standard error are displayed.*

4.4 Discussion

The results in this work demonstrate the potential for finite, ordered lexicons of ellipse masks for specifying object segments using only noisy binary inputs according to the PM algorithm. Our method performs well in noise levels that fall in the typical range of crossover probabilities in binary-input HCI systems, indicating an ability to rapidly and precisely specify object segments in a manner resilient to input errors. This is plausibly due to the explicit noise modeling in posterior matching as well as optimal convergence guarantees provided by feedback information theory. The promising performance of EllipseLex overall as well as by class and segment size on a large dataset that includes many classes of real-world objects in natural contexts suggests that our method is conducive to real-world interactive object segmentation in images. Furthermore, EllipseLex has the advantage that after an ellipse mask is produced it can be input into any post-processing stage that accepts a bounding box to produce segmentations by utilizing inherent structure in natural images.

Because this work is the first attempt to create a lexicon of segments that is robustly navigated with noisy binary inputs in an information-efficient manner, this work is intended

to test its performance in simulation to verify its "best case" properties, as is standard in previous literature studying segmentation with binary inputs [111, 112, 113, 97]. Since EllipseLex performs competitively against a baseline method on this task in realistic object segmentation scenarios, a follow-up study involving human subjects is merited to test human ability in ordering guessed ellipses with respect to target segments, evaluate possible user biases, and assess overall human operation of EllipseLex.

CHAPTER 5

ACTIVE ORDINAL QUERYING FOR TUPLEWISE SIMILARITY LEARNING

In Chapters 3 and 4, the problem of one-bit interaction design for HCIs fit conveniently into the framework of feedback communication over a BSC via posterior matching, since both the interactive BCI and image segmentation tasks are endowed with structures amenable to lexicographical sorting by a human. However, general problems in IML do not typically contain the appropriate query structures that allow for direct application of PM. In particular, it is relatively uncommon for IML problems to be adequately modeled as encoding a scalar parameter on the unit interval with a comparator function; in Chapter 7 we address this challenge explicitly in the case of general active learning problems. Instead, in this chapter and the next we show in the case of similarity and preference learning how there may still exist convenient IML query structures that allow for simplifying approximations to be made for the design of computationally efficient query selection strategies that approximate the action of information gain maximization.

5.1 Relative Similarity Learning

Similarity learning[1] is the process of assigning point coordinates to objects in a dataset such that distances between objects in the learned space are consistent with notions of similarity as perceived by humans (see for example Figure 5.1). While these objects usually exist in some high-dimensional space (e.g., images, audio), very often the semantic information humans attribute to these objects lies in a low-dimensional space (e.g., items, words). Once this low-dimensional embedding is learned, existing intelligent algorithms [116, 4] can be used to search the dataset with query complexity scaling in the embedding dimension,

[1]This chapter is in collaboration with Stefano Fenu and Dr. Christopher Rozell. GC and SF contributed equally on most project components, and coauthored the corresponding publication in [3]. CR supervised the project.

Figure 5.1: Low-dimensional similarity embedding of images.[2]

allowing large datasets to be searched quickly in applications such as task selection for robot learning from demonstration [117], object recognition [118], or image retrieval [119].

To construct such an embedding for a given set of objects, queries that capture the similarity statistics between the objects in question must be made to human experts. While there exist several types of similarity queries that can be made (e.g., relative attributes between objects [120]), we focus on relative similarity queries posed to an oracle comparing objects with respect to a "head" (i.e., reference) object. Relative similarity queries are useful because they gather object similarity information using only object identifiers rather than predetermined features or attributes, allowing similarity learning methods to be applied to any collection of uniquely identifiable objects. In contrast, if a head object were not specified, an oracle would need to use a feature-based criterion for ranking the object set, which is not viable in many applications of interest (e.g., learning human preferences).

Such relative similarity queries typically come in the form of triplet comparisons (i.e., "is object a more similar to object b or c?") [121, 122, 123]. *In our first main contribution, we extend these queries to larger rank orderings of tuples of k objects to gather more information at once for similarity learning*. This query type takes the form "rank objects b_1 through b_{k-1} according to their similarity to object a." To the best of our knowledge, this

[2]Image provided by courtesy of Stefano Fenu.

(a) (b)

Figure 5.2: In (a), it is ambiguous which item should be chosen as more similar to the head object, since both comparison items are similar in distinct ways. In (b), adding one more comparison item can add context to disambiguate this choice.

study is the first attempt to leverage this generalized query type in similarity learning. The use of this query type is motivated by the fact that comparing multiple objects simultaneously provides increased context for a human expert [124], which can increase labeling consistency without a significant increase in human effort per query [125] and has demonstrated benefits in settings such as rank learning [126]. In technical terms, tuplewise queries capture joint dependence between objects that isn't captured in triplet comparisons (which are often incorrectly modeled as independent queries). To illustrate this point, consider the difference between the triplet query and tuple query presented in Figure 5.2. In the triplet query, multiple attributes could be used to rank a given query, increasing the ambiguity about which item should be chosen as more similar to the reference. Adding an item to the tuple can provide additional context about the entire dataset to the oracle, clarifying which criterion should be used to rank the tuple and thereby making the query less ambiguous.

While tuple queries are appealing, their use presents two major challenges. First, in a dataset of N objects queried with tuples of size k there are $N\binom{N-1}{k-1}$ possible tuples. Labeling these individual tuples is prohibitively time consuming for large datasets. Even if uniformly random query selection is used to downsample this set, there is evidence that such a strategy is still punitively expensive [127]. Requesting an exhaustive number of queries is also

inefficient from an information standpoint, since there is redundancy in the set of all tuple rankings. Second, in many settings of interest, the oracle answering such queries may be stochastic. For example, crowd oracles may aggregate responses from experts with differing similarity judgments [121], and individual oracles can be unreliable over time (especially for queries regarding similar objects).

These issues can be ameliorated in part by leveraging tools from active learning, the goal of which is to minimize the total labeling cost including the number of expert interactions (usually corresponding to monetary cost), aggregate response time, and computational cost needed to dynamically select queries. This is achieved through adaptive approaches that increase learning efficiency by using previous query responses to determine which information about a model is still "missing" as well as model the oracle's stochasticity. In this framework, unlabeled data points that optimize a measure of informativeness are selected for expert labeling. One such metric, mutual information, is a popular way to assess the reduction in uncertainty a query provides about unknown learning parameters [39, 41, 20]. In active similarity learning, the state-of-the-art is a strategy called "Crowd Kernel Learning" (CKL) that selects triplets that maximize the mutual information between a query response and the embedding coordinates of the head object [121]. However, CKL does not apply to ordinal queries of general tuples sizes ($k > 3$), and its formulation of mutual information only measures the information a query provides about the embedding coordinates of the head object, disregarding information about the locations of the other objects in the query.

In our second main contribution, we address these deficiencies and the lack of an active similarity learning strategy for our new query type by introducing a novel method for efficient and robust adaptive selection of tuplewise queries of arbitrary size. Our method, called *InfoTuple*, maximizes the mutual information a query response provides about the *entire* embedding, which is a direct measure of query informativeness that leverages the high degree of coupling between all of the objects in a query. InfoTuple relies on a novel

set of simplifying yet reasonable assumptions for tractable mutual information estimation from a single batch of Monte Carlo samples. Our approach accounts for all objects in a query, while avoiding the need to decompose mutual information into a prohibitive number of terms. We demonstrate the performance of this method across datasets, oracle models, and tuple sizes, using both synthetic tests and newly collected large-scale human response datasets. In particular, we empirically show that InfoTuple's performance exceeds that of CKL and random queries, and furthermore that it benefits significantly from using larger tuples even after normalizing for tuple size. We also demonstrate the utility of our novel query type by showing an increase in query consistency for larger tuples over triplets, and show that these advantages can be gained without excessive labeling-time increases.

5.2 Related Work

Similarity learning from triplets is increasingly commonplace in modern AI, and popular deep learning architectures have been developed to leverage triplet labels [123]. Frameworks such as that of [128] or t-STE [122] are relatively ubiquitous in the visualization community, and attempt to directly capture a notion of visual similarity close to that observed in psychometrics literature (e.g., [129]). However, for large datasets it is often punitively expensive to collect such exhaustive relationship data from labelers, so the development of approximate methods of learning such embeddings is a matter of interest to the AI community.

The bulk of the existing literature on active selection of ordinal queries for constructing these embeddings focuses on the case where distance relationships between objects can be determined with absolute certainty. This deterministic case is well studied, and lower bounds exist on the sample complexity needed to learn high-quality embeddings [127]. In reality, responses are often not deterministic for a number of practical reasons and probabilistic MDS methods have been proposed to model such cases [121]. Analytic results do exist characterizing bounds on prediction error in this setting [130], but determining optimal

strategies for query selection in the stochastic setting remains largely an open problem.

Specifically, to the best of our knowledge there have been no previous attempts to adaptively select relative comparisons with respect to a head object for general tuple sizes ($k \geq 3$) in the context of similarity learning. Prior work [125, 131] develops an active strategy for sampling tuples, but the query task is relative attribute ranking within the tuple according to some pre-specified attribute as opposed to comparison against a head object. Other work [132] actively samples the same query type as our study, but in the context of classification via label propagation. Research exists that is similar to our learning scenario since they actively sample tuples for relative similarity comparisons to a head for the sake of learning and searching an embedding of objects [133], but these comparisons are ternary 'similar', 'dissimilar', or 'neither' labels and their methodology differs from the mutual information approach presented here. Similarly, other work [134] actively samples tuplewise queries with binary 'similar' or 'dissimilar' label responses with respect to a head, but in the context of classification. Finally, the prior work [135] also employs such tuplewise binary queries for similarity learning, but with randomly selected queries. While no previous study addresses the similarity learning problem that we explore here, the existing literature demonstrates the effectiveness, efficiency, and feasibility of queries involving multiple objects and provides support for the practical use of our proposed query type.

5.3 Methods

The problem of adaptively selecting a tuplewise query can be formulated as follows: for a dataset $\mathcal{X}$ of N objects, assume that there exists a d-dimensional vector of embedding coordinates for each object which are concatenated as columns in matrix $M \in \mathbb{R}^{d \times N}$. The *similarity matrix* corresponding to M is given by $K = M^T M$, which implies an $N \times N$ matrix D of distances between the objects in $\mathcal{X}$. Specifically, the squared distance between the ith and jth objects in the dataset is given by $D^2_{i,j} = K_{i,i} - 2K_{i,j} + K_{j,j}$. These distances are assumed to be consistent in expectation with similarity comparisons from an *oracle*

(e.g., human expert or crowd) such that similar objects are closer and dissimilar objects are farther apart. Since relative similarity comparisons between tuples of objects inform their relative embedding distances rather than their absolute coordinates, our objective is to learn similarity matrix K rather than M, which can be recovered from K up to a change in basis [121].

A tuplewise oracle query at time step n is composed of a "body" of objects $B^n = \{b_1^n, b_2^n, \dots b_{k-1}^n\}$ which the oracle ranks by similarity with respect to some "head" object a_n. Letting $Q_n = \{a_n\} \cup B^n$ denote the nth posed tuple, we denote the oracle's ranking response as $R(Q_n) = \{R_1(Q_n), R_2(Q_n), \dots R_{k-1}(Q_n)\}$ which is a permutation of B^n such that $R_1(Q_n) \prec R_2(Q_n) \cdots \prec R_{k-1}(Q_n)$ where $b_i \prec b_j$ indicates that the oracle ranks object b_i as more similar to a_n than object b_j. Since the oracle is assumed to be stochastic, $R(Q_n)$ is a random permutation of B^n governed by a distribution that is assumed to depend on K. This assumed dependence is natural because oracle consistency is likely coupled with notions of object similarity, and therefore with distances between the objects in M. The actual recorded oracle ranking is a random variate of $R(Q_n)$ denoted as $r(Q_n)$. Letting $r^n = \{r(Q_1), r(Q_2), \dots r(Q_n)\}$, define $\widehat{K}^n$ as an estimate of K learned from previous rankings r^n, with corresponding distance matrix $\widehat{D}^n$.

Suppose that tuples $Q_1, Q_2, \dots Q_{n-1}$ have been posed as queries to the oracle with corresponding ranking responses r^{n-1}, and consider a Bayes optimal approach where after the nth query we estimate the similarity matrix as the maximum a posteriori (MAP) estimator over a similarity matrix posterior distribution given by $f(K|r^n)$, i.e. $\widehat{K}_n = \arg\max_K f(K|r^n)$. To choose the query Q_n, a reasonable objective is to select a query that maximizes the achieved posterior value of the resulting MAP estimator (or equivalently one that maximizes the achieved logarithm of the posterior), corresponding to a higher level of confidence in the estimate. However, because the oracle response $r(Q_n)$ is unknown before a query is issued, the resulting maximized posterior value is unknown. Instead, a more reasonable objective is to select a query that maximizes the *expected* value over the

posterior of $R(Q_n)$. This can be stated as

$$\arg\max_{Q_n} \ \mathbb{E}_{R(Q_n)} \left[\max_K \log f(K|R(Q_n), r^{n-1}) \mid r^{n-1} \right].$$

In practice, this optimization is infeasible since each expectation involves the calculation of several MAP estimates. Noting that maximization is lower bounded by expectation, this optimization can be relaxed by replacing the maximization over K with an expectation over its posterior distribution given $R(Q_n)$ and r^{n-1}, resulting in a feasible maximization of a lower bound given by

$$\arg\max_{Q_n} \ -h(K \mid R(Q_n), r^{n-1}), \tag{5.1}$$

where $h(K \mid R(Q_n), r^{n-1})$ denotes conditional differential entropy. Let the mutual information between K and $R(Q_n)$ given r^{n-1} be defined by

$$I(K; R(Q_n) \mid r^{n-1}) = h(K \mid r^{n-1}) - h(K \mid R(Q_n), r^{n-1}),$$

and note that the second term is equal to eq. (5.1) while the first term does not depend on the choice of Q_n. Thus, maximizing eq. (5.1) over Q_n is equivalent to maximizing $I(K, R(Q_n) \mid r^{n-1})$. Hence, we can adaptively select tuples that maximize mutual information as a means of greedily maximizing a lower bound on the log-posterior achieved by a MAP estimator, corresponding to a high estimator confidence.

However, calculating eq. (5.1) for a candidate tuple is an expensive procedure that involves estimating the differential entropy of a combinatorially large number of posterior distributions, since the expectation with respect to $R(Q_n)$ is taken over $(k-1)!$ possible rankings. Instead, in the spirit of [136] we leverage the symmetry of mutual information to write the equivalent objective

$$\arg\max_{Q_n} \ H(R(Q_n) \mid r^{n-1}) - H(R(Q_n) \mid K, r^{n-1}), \tag{5.2}$$

where $H(\cdot \mid \cdot)$ denotes conditional entropy of a discrete random variable. Estimating eq. (5.2) for a candidate tuple only involves averaging ranking entropy over a *single* posterior $f(K \mid r^{n-1})$, regardless of the value of k. This insight, along with suitable probability models discussed in the next sections, allows us to efficiently estimate mutual information for a candidate tuple over a single batch of Monte Carlo samples, rather than having to sample from $(k-1)!$ posteriors.

Furthermore, by interpreting entropy of discrete random variables as a measure of uncertainty, this form of mutual information maximization has a satisfying qualitative interpretation. The first entropy term in eq. (5.2) prefers tuples whose rankings are uncertain, preventing queries from being wasted on predictable or redundant responses. Meanwhile, the second term discourages tuples that have high expected uncertainty when conditioned on K; this prevents the selection of tuples that, even if K were somehow revealed, would *still* have uncertain rankings. Such queries are inherently ambiguous, and therefore uninformative to the embedding. Thus, maximizing mutual information optimizes the balance between these two measures of uncertainty and therefore prefers queries that are unknown to the learner but that can still be answered consistently by the oracle.

5.3.1 Estimating Mutual Information

To tractably estimate the entropy terms in eq. (5.2) for a candidate tuple, we employ several simplifying assumptions concerning the joint statistics of the query sequence and the embedding that allow for efficient Monte Carlo sampling:

(A1) As is common in active learning settings, we assume that each query response $R(Q_n)$ is statistically independent of previous responses r^{n-1}, when conditioned on K.

(A2) The distribution of $R(Q_n)$ conditioned on K is only dependent on the distances between a_n and the objects in B^n, notated as set $D_{Q_n} \coloneqq \{D_{a_n,b} : b \in B\}$. This direct dependence of tuple ranking probabilities on inter-object distances is rooted in the fact that the distance relationships in the embedding are assumed to capture oracle

response behavior, and is a common assumption in ordinal embedding literature [122, 121]. Furthermore, this conditional independence of $R(Q_n)$ from objects $x \notin Q_n$ is prevalent in probabilistic ranking literature [137]. In the next section, we describe a reasonable ranking probability model that satisfies this assumption.

(A3) D is conditionally independent of r^{n-1}, given $\widehat{D}^{n-1}$. This assumption is reasonable because embedding methods used to estimate $\widehat{K}^{n-1}$ (and subsequently $\widehat{D}^{n-1}$) are designed such that distances in the estimated embedding preserve the response history contained in r^{n-1}. In practice, it is more convenient to model an embedding posterior distribution by conditioning on $\widehat{D}^{n-1}$, learned from the previous responses r^{n-1}, rather than by conditioning on r^{n-1} itself. This is in the same spirit of CKL, where the current embedding estimate is used to approximate a posterior distribution over points.

(A4) Conditioned on $\widehat{D}^{n-1}$, the posterior distribution of D_{Q_n} is normally distributed about the corresponding values in $\widehat{D}^{n-1}_{Q_n}$, i.e. $D^{n-1}_{a_n,b} \sim \mathcal{N}(\widehat{D}^{n-1}_{a_n,b}, \sigma^2_{n-1})\ \forall b \in B$, where σ^2_{n-1} is a variance parameter. Imposing Gaussian distributions on inter-object distances is a recent approach to modeling uncertainty in ordinal embeddings [138] that allows us to approximate the distance posterior with a fixed batch of samples from a simple distribution. Furthermore, the combination of this model with item (A2) means that we only need to sample from the normal distributions corresponding to the objects in Q_n. We choose σ^2_{n-1} to be the sample variance of all entries in $\widehat{D}^{n-1}$, which is a heuristic that introduces a source of variation that preserves the scale of the embedding.

Combining these assumptions, with a slight abuse of notation by writing $H(X) = H(p(X))$ for a random variable X with probability mass function $p(X)$, and $\mathcal{N}^{n-1}_{Q_n}$ to represent normal

distribution $\mathcal{N}(\widehat{D}_{Q_n}^{n-1}, \sigma_{n-1}^2)$, we have

$$
\begin{aligned}
H(R(Q_n) \mid r^{n-1}) &= H\left(\mathbb{E}_K\left[p(R(Q_n) \mid K, r^{n-1}) \mid r^{n-1}\right]\right) \\
&= H\left(\mathbb{E}_K\left[p(R(Q_n) \mid K) \mid r^{n-1}\right]\right) && \text{(A1)} \\
&= H\left(\mathbb{E}_{D_{Q_n}}\left[p(R(Q_n) \mid D_{Q_n}) \mid r^{n-1}\right]\right) && \text{(A2)} \\
&= H\left(\mathbb{E}_{D_{Q_n}}\left[p(R(Q_n) \mid D_{Q_n}) \mid \widehat{D}^{n-1}\right]\right) && \text{(A3)} \\
&= H\left(\mathbb{E}_{D_{Q_n} \sim \mathcal{N}_{Q_n}^{n-1}}\left[p(R(Q_n) \mid D_{Q_n})\right]\right). && \text{(A4)}
\end{aligned}
$$

Similarly, we have

$$
H(R(Q_n) \mid K, r^{n-1}) = \mathbb{E}_{D_{Q_n} \sim \mathcal{N}_{Q_n}^{n-1}}\left[H\left(p(R(Q_n) \mid D_{Q_n})\right)\right].
$$

This formulation allows a fixed-sized batch of samples to be drawn and evaluated over, the size of which can be tuned based on real-time performance specifications. This enables us to separate our computational budget and mutual information estimation accuracy from the size of the tuple query.

5.3.2 Embedding Technique

In order to maximize the flexibility of our approach and draw a closer one-to-one comparison to existing methods for similarity learning, we train our embedding on our actively selected tuples by first decomposing a tuple ranking into $k-2$ *constituent triplets* defined by the set $\{R_i(Q_m) \prec R_{i+1}(Q_m) : 1 \leq i \leq k-2,\ m \leq n\}$, and then learning an embedding from these triplets with any triplet ordinal embedding algorithm of choice. Since we compare performance against CKL in our experiments, our proposed embedding technique follows directly from the probabilistic MDS formulation in [121] so as to evaluate the effectiveness of our novel query selection strategy in a controlled setting. We wish to constrain our learned similarity matrix to the set of symmetric unit-length PSD matrices, so we consider the set

S of such matrices: $S = \{K \succeq 0 | K_{11} = K_{22} = \cdots = K_{NN} = 1\}$. We denote the closest matrix in S to K as $P_S(K) = \arg\min_{A \in S} \sum_{ij} (K_{ij} - A_{ij})^2$. Projecting to the element in S closest to K is a quadratic program, which we solve by gradient projection descent on K. We do this by selecting an initial K^0 arbitrarily, and for each iteration computing $K^{t+1} = P_S(K^t - \eta \nabla l_t(K^t))$ with l_t being the empirical log-loss at iteration t i.e. $l_t = \log \frac{1}{p}$, and p being the probability that the oracle correctly ordered the constituent triplets of the selected tuples. For the response probability of an individual triplet, we adopt the model in [121] that is reminiscent of Bradley-Terry pairwise score models [139]: for parameter $\mu > 0$, $p(b_1 \prec b_2) = (D^2_{a,b_2} + \mu)/(D^2_{a,b_1} + D^2_{a,b_2} + 2\mu)$.

5.3.3 Tuple Response Model

Our proposed technique is compatible with any tuple ranking model that satisfies (A2). However, since we use the triplet response model listed above in the probabilistic MDS formulation, combined with the need for a controlled test against CKL, we extend their model to the tuplewise case as follows: we first decompose an oracle's ranking into its constituent triplets, and then apply

$$p(R(Q_n) \mid D_{Q_n}) \propto \prod_{i=1}^{k-2} \frac{D^2_{a,R_{i+1}(Q_n)} + \mu}{D^2_{a,R_i(Q_n)} + D^2_{a,R_{i+1}(Q_n)} + 2\mu},$$

for parameter $\mu > 0$. This model corresponds to oracle behavior that ranks objects proportionally to the ratio of their distances with respect to a, such that closer (resp. farther) objects are more (resp. less) likely to be deemed similar. Models of this type are generally held to be similar to the scale-invariant models present in some human perceptual systems [129].

5.3.4 Adaptive Algorithm

Combining these concepts, we have the following algorithm titled InfoTuple, summarized in Algorithm 2: the algorithm requires that some initial set of randomly selected tuples be

labeled to provide a reasonable initialization of the learned similarity matrix. Since the focus of this work is on the effectiveness of various adaptive selection methods, this initialization is standardized across methods considered in our results. Specifically, following established practice [121], a "burn-in" period is used where T_0 random triplets are posed for each object a in object set $\mathcal{X}$, with a being the head of each query. Then, for each time step n we learn a similarity matrix $\widehat{K}^{n-1}$ on the set of previous responses r^{n-1} by using probabilistic MDS. To make a comparison to CKL, we follow their procedure and subsequently pose a single tuple for each head $a \in \mathcal{X}$. However, it is possible to adaptively choose a with our method by searching over both head and body objects for a maximally informative tuple. The body of each tuple, given some head a, is chosen by uniformly downsampling the set of possible bodies and selecting the one that maximizes the mutual information, calculated using the aforementioned probability model in our estimation procedure. This highlights the importance of computational tractability in estimating mutual information, since for a fixed computing budget per selected query, less expensive mutual information estimation allows for more candidate bodies to be considered. For a tuple size of k we denote the run of an algorithm using that tuple size as InfoTuple-k.

5.4 Experiments

Our results on synthetic and human response datasets show that InfoTuple's adaptive selection outperforms both random query selection and that of CKL. This is true even when normalizing for changes in tuple size and when normalizing for labeling effort, showing that the incurred benefit is not only due to the increased information inherently present in larger tuples but also due to our improved adaptive selection. We also show that there are inherent consistency benefits to the use of larger queries, and that human labelers can respond to these query types in practice without undue cost.

Algorithm 2: InfoTuple-k

Input: object set $\mathcal{X}$, rate ω, sample size N_f, horizon T

$r^0 \leftarrow \emptyset$ initialize set of oracle responses

$\widehat{K}^0 \leftarrow$ initialize embedding

for $n = 1$ **to** T **do**

$\widehat{D}^{n-1} \leftarrow$ calculate pairwise distances from $\widehat{K}^{n-1}$

$\sigma^2_{n-1} \leftarrow \frac{1}{N^2} \sum_{d \in \widehat{D}^{n-1}} \left(d - \frac{1}{N^2} \sum_{d \in \widehat{D}^{n-1}} d \right)^2$

for all $a \in \mathcal{X}$ **do**

$\beta \leftarrow$ downsampled $k-1$ sized bodies at rate ω

for all $B \in \beta$ **do**

$Q \leftarrow \{a\} \cup B$

$D_s \sim \mathcal{N}(\widehat{D}^{n-1}_Q, \sigma^2_{n-1})$, drawn N_f times

$I_B \leftarrow H\left(\sum_{D \in D_s} \frac{p(R(Q)|D)}{N_f} \right) - \sum_{D \in D_s} \frac{H(p(R(Q)|D))}{N_f}$

end for

$B \leftarrow \arg\max_{B \in \beta} I_B$

$r \leftarrow$ oracle ranks objects in B relative to a

$r^n \leftarrow r^{n-1} \cup r$

end for

$\widehat{K}^n \leftarrow \mathrm{probabilisticMDS}(r^n)$

end for

Output: $\widehat{K}^T$

5.4.1 Datasets

To evaluate algorithm performance in a controlled setting, we constructed a synthetic evaluation dataset by generating a point cloud drawn from a d-dimensional multivariate normal distribution. To simulate oracle responses for this dataset, we use the popular Plackett-Luce permutation model to sample a ranking for a given head and body [126, 140]. In this response model, each object in a tuple body is assigned a score according to a scoring function, which in our case is based on the distance in the underlying space between each object and the head. For a given subset of body objects, the probability of an object being ranked as most similar to the head is its score divided by the scores of all objects in that subset, and we generate each simulated oracle response by sequentially sampling objects without replacement from a tuple according to this model. We chose this tested response model to differ from the one we use to estimate mutual information in order

to demonstrate the robustness of our method to mismatched noise models, and evaluate an additional Gaussian noise model in the appendix. This dataset was used to compare InfoTuple-3, InfoTuple-4, InfoTuple-5, CKL, Random-3, and Random-5 across noiseless, Gaussian, and Plackett-Luce oracles.

To demonstrate the broader applicability of our work in real-world settings and evaluate our proposed technique on perceptual similarity data, we also collected a large dataset of human responses to tuplewise queries through Amazon Mechanical Turk. Drawing 3000 food images from the Food-10k dataset [141], we presented over 7000 users with a total of 192,000 varying-size tuplewise queries chosen using Infotuple-3, InfoTuple-5, Random-3, and Random-5 as selection strategies across three repeated runs of each algorithm. Users were evaluated with one repeat query out of 25, and users who responded inconsistently to the repeat query were discarded. Query bodies were always shuffled when presented to minimize the impact of any possible order effect, and it was not found to be the case that there was any significant order effect in the human responses. Initial embeddings for each of these methods were trained on 5,000 triplet queries drawn from [141]. Although experimental costs prevented us from extending the experiments in Figure 5.3c to larger tuple sizes, in order to verify the feasibility of having humans respond to larger tuples in practice we performed a separate data collection in which we asked users to rank randomly selected tuples up to a size of $k = 10$ and recorded the labeling time for each response.

5.4.2 Evaluation Metrics

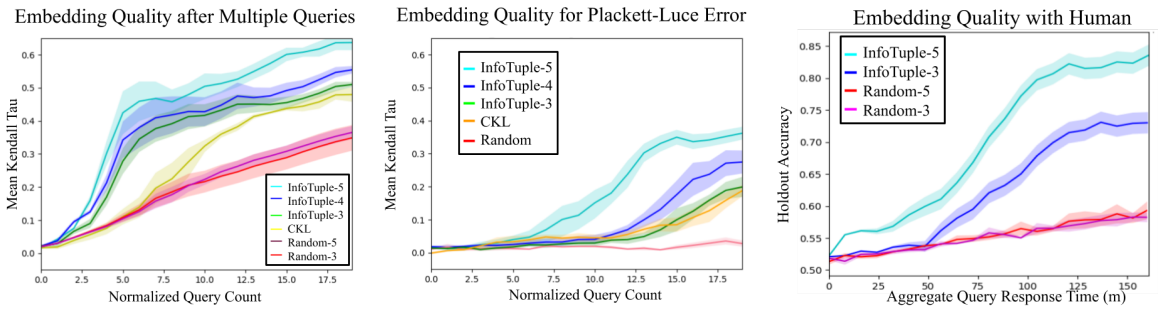
In order to directly measure the preservation of object rankings between the ground truth object coordinates and the embedding learned from oracle responses, we use Kendall's Tau rank correlation coefficient [142]. To get an aggregate measure of quality when comparing an estimated embedding to a ground truth embedding, we take the mean of Kendall's Tau across the total rankings obtained by setting each object as the head and sorting all objects by embedding distance to the head. In our experiments with human respondents it is not

possible to use this measure, as the "ground truth" embedding that corresponds to human preferences is not known. In these cases we instead measure the accuracy with respect to a held-out set of queries drawn from the Food-10k dataset [141], which is a common embedding quality metric [122, 141]. The holdout accuracy is the fraction of a held out set of triplet comparisons that agrees with distances in the final learned embedding. To capture a notion of the internal coherence between a set of oracle responses and an embedding that is learned from them, we measure the mean rank correlation between each response in this set and the ranking over the same objects imputed from the learned embedding–we refer to this as the *coherence* of a set of tuples.

One issue that naturally arises when comparing results from strategies that select tuples of different size is normalization, as larger tuples will naturally be more informative. In human-response studies normalization is relatively straightforward, as we can simply normalize with respect to the total time spent labeling queries in order to reflect the total labeling cost. While other more comprehensive measures of labeler effort exist, labeling time is a first-order approximation for the cognitive load of a labeling task and is the most salient metric for determining the cost of a large-scale data collection. In the case of synthetic data, we instead compute a *normalized query count* corresponding to the number of constituent triplet comparisons defining the relation of each body point to the head in the tuple. This is justified since in practice we decompose tuples in this way when feeding them into the embedding algorithm, and corresponds to the size of a tuple's transitive reduction (a common representation in learning-to-rank literature [143]). Additional experimental details such as hyperparameter selection are available in Appendix B.

5.4.3 Experimental Results

Using simulated data, we show a direct comparison of embedding quality from using InfoTuple, CKL, and Random queries under a simulated deterministic oracle (Figure 5.3a) and two simulated stochastic oracles (Figure 5.3b), and note that InfoTuple consistently



(a) (b) (c)

Figure 5.3: (a) and (b) show a comparison of the fidelity of the learned embedding to the ground truth embedding with a simulated deterministic (left) and a stochastic (right) oracle, plotted with ± 1 standard error. Results shown are for a synthetic dataset of $N = 500$ points from a two-dimensional dataset. (c) shows holdout accuracy on human-subject tests with $N = 5000$.

outperformed the other methods. We note two important observations from these results: first, regardless of the oracle used, larger tuple sizes for InfoTuple tended to perform better and converge faster than did smaller tuple sizes even after normalizing for the tuple size, showing the benefit of larger tuples beyond just providing more constituent triplets. Recalling that the Plackett-Luce oracle was not directly modeled in our estimate of mutual information, this lends support to the robustness of our technique to various oracle distributions. Second, results on Random-3, Random-4 and Random-5 are comparable, implying both that the improvements seen in InfoTuple are not solely due to the difference in tuple sizes and that our choice of normalization is appropriate. Note that since random query performance did not change with tuple size, Figure 5.3b only shows Random-3 for the sake of visual clarity.

Using the Mechanical Turk dataset described previously, we also show that these basic results extend to real data situations when the stochastic response model is not exactly known, and allows us to examine the complexity of acquiring data with increasing tuple sizes. While larger tuples sizes produce more informative queries, it is possible that the information gained incurs a hidden cost in the complexity or labeler effort involved in acquiring the larger query. Specifically, it can be the case that maximizing query informativeness can produce queries that are more difficult to answer [144]. Fortunately, the results on tuplewise

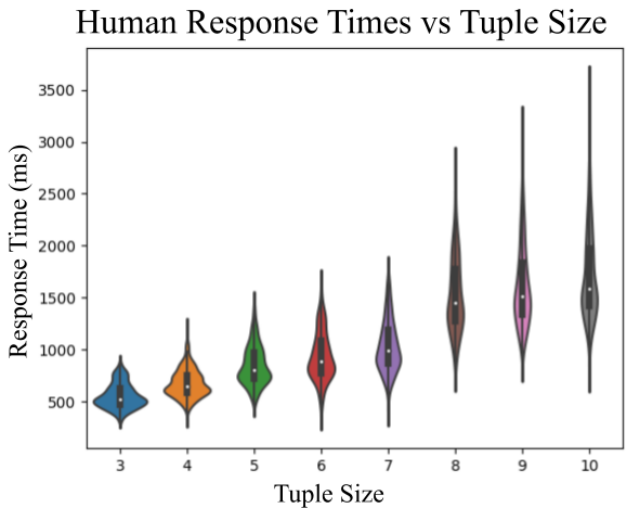


Figure 5.4: This violin plot shows the distribution of timing responses for random queries from size $k = 3$ to $k = 10$, for the purpose of measuring labeler effort. The response time for $k \leq 7$ shows only modest increases in cost, although responses above these sizes require significantly more effort.

comparisons collected for our Mechanical Turk dataset indicate that this is not an issue for our proposed use case. In particular, Figure 5.3c shows the accuracy results when predicting the labels from a held out set of 1200 triplet queries. These results show an increase in the effectiveness of InfoTuple adaptive selection as well as increasing tuples sizes when plotted against the aggregate query response time. In other words, any increase in query complexity (measured by response time) is more than compensated for by the increased information acquired by the query and the increase in the resulting quality of the learned embedding.

Figure 5.4 explores this issue further by examining the response times for our additional timing dataset as a function of query size. There are only modest increases in the ranking time cost with increasing tuple size, leading to the significant gains observed in normalized information efficiency in this range of tuple sizes. While it is true that complexity cost will continue to increase for larger tuple sizes and the gains in information efficiency are not guaranteed to increase indefinitely and there may also be additional factors in the choice of optimal tuple size for a given problem, we show that up to a modest tuple size it is strictly more useful to ask tuplewise queries than triplet queries.

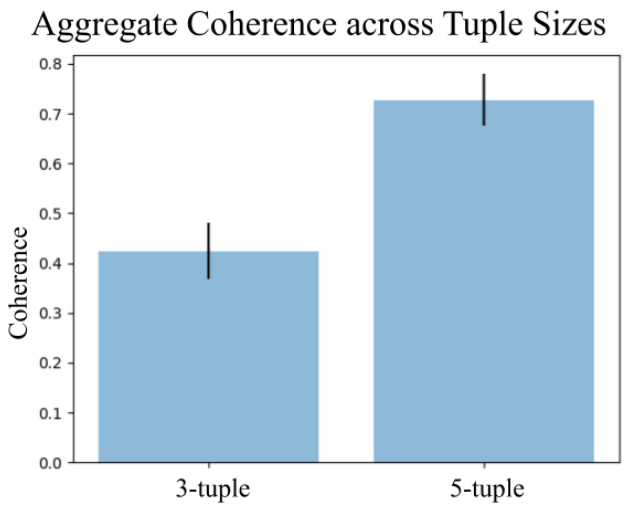


Figure 5.5: Measuring the aggregate coherence for all tuples of size 3 and size 5 (i.e. over 80,000 tuples at each size) with respect to an aggregate embedding learned for each tuple size, we find that there is a significant difference in their internal coherence as measured by a t-test (p=0.007181). We hypothesize that the difference is due to an increase in context available to the oracle. Error bars depict ± 1 standard error.

One possible reason for why tuples outperform triplets is that asking a query that contains more objects provides additional context for the oracle about the contents of the dataset, allowing it to more reliably respond to ambiguous comparisons than if these were asked as triplet queries. As a result of this increase in context, oracles tend to respond to larger queries significantly more coherently than they do to smaller ones, as shown in Figure 5.5. We note that this is not guaranteed to increase indefinitely as larger tuples are considered, but the effect is noticeable for modest increases in tuple sizes and is clear when comparing 5-tuples to triplets.

5.5 Discussion

In this chapter we proposed InfoTuple, an adaptive tuple selection strategy based on maximizing mutual information for relative tuple queries for similarity learning. We introduce the tuple query for similarity learning, present a novel set of assumptions for efficient estimation of mutual information, and through the collection of new user-response datasets, provide new insights into the gains acquired by using larger tuples in learning efficiency

and query consistency. After testing on synthetic and real datasets, InfoTuple was found to more effectively learn similarity-based object embeddings than random queries and state-of-the-art triplet queries for both synthetic data (with a typical oracle model) and in a real world experiment. The performance gains were especially evident for larger tuples and even after normalizing for tuple size, indicating that the proposed selection objective that maximizes the mutual information between the query response and the entire embedding yields information gains that are not simply due to an increase in tuple size. Taken together, these results suggest that large tuples selected with InfoTuple supply richer and more robust embedding information than their triplet and random counterparts.

In practice, larger tuple sizes can provide more context for the oracle, increasing the reliability of the responses without significant increases in labeling effort. In the pathological extreme, the level of effort almost certainly outweighs the benefits of larger tuples, as an oracle would have to provide a ranking over the entire dataset. Despite this downside in extreme tuple sizes, our human study results indicate that performance increases hold up in the real-world for moderate tuple sizes. This interesting tradeoff between informativeness per query and real-world oracle behavior merits a more comprehensive study on the psychometric aspects of the problem, in the spirit of [16].

CHAPTER 6

ACTIVE EMBEDDING SEARCH VIA NOISY PAIRED COMPARISONS

6.1 Preference Searching with Paired Comparisons

In this chapter,[1] we consider the task of user preference learning, where we have a set of *items* (e.g., movies, music, or food) embedded in a Euclidean space and aim to represent the preferences of a *user* as a continuous point in the same space (rather than simply a rank ordering over the items) so that their preference point is close to items the user likes and far from items the user dislikes. To estimate this point, we consider a system using the *method of paired comparisons*, where during a sequence of interactions a user chooses which of two given items they prefer [145]. For instance, to characterize a person's taste in food, we might ask them which one of two dishes they would rather eat for a number of different pairs of dishes. The recovered preference point can be used in various tasks, for instance in the recommendation of nearby items, personalized product creation, or clustering of users with similar preferences. We refer to the entire process of querying via paired comparisons and continuous preference point estimation as *pairwise search*, and note that this is distinct from the problem of searching for a single discrete item in the fixed dataset. A key goal of ours is to *actively* choose the items in each query and demonstrate the advantage over non-adaptive selection.

More specifically, given N items, there are $O(N^2)$ possible paired comparisons. Querying all such pairs is not only prohibitively expensive for large datasets, but also unnecessary since not all queries are informative; some queries are rendered obvious by the accumulation

[1] The material before Section 6.5 is in collaboration with Dr. Andrew Massimino, Dr. Mark Davenport, and Dr. Christopher Rozell. GC and AM contributed equally on most project components. GC was the lead author of the associated publication in [4]. CR and MD supervised the project. Section 6.5 is in collaboration with Matthew O'Shaughnessy, Dr. Mark Davenport, and Dr. Christopher Rozell. GC and MO contributed equally to the project (which was initially developed during an overnight layover in Charles de Gaulle Airport) and coauthored the resulting publication in [5]. CR and MD supervised the project.

of evidence about the user's preference point, while others are considered ambiguous due to noise in the comparison process. *Given these considerations, the main contribution of this chapter is the design and analysis of two new query selection algorithms for pairwise search that select the most informative pairs by directly modeling redundancy and noise in user responses*. While previous active algorithms have been designed for related paired comparison models, none directly account for probabilistic user behavior as we do here. To the best of our knowledge our work is the first attempt to search a low-dimensional embedding for a *continuous* point via paired comparisons while directly modeling *noisy* responses.

Our approach builds upon the popular technique in active learning and Bayesian experimental design of greedily maximizing information gain [39, 41, 20]. In our setting, this corresponds to selecting pairs that maximize the mutual information between the user's response and the unknown location of their preference point. We provide new theoretical and computational insights into relationships between information gain maximization and estimation error minimization in pairwise search, and present a lower bound on the estimation error achievable by any query strategy.

Due to the known difficulty of analyzing greedy information gain maximization [19] and the high computational cost of estimating mutual information for each pair in a pool, we propose two strategies that each maximize new lower bounds on information gain and are simpler to analyze and compute respectively. We present upper and lower bounds on the performance of our first strategy, which then motivates the use of our second, computationally cheaper strategy. We then demonstrate through simulations using a real-world dataset how both strategies perform comparably to information maximization while outperforming state-of-the-art techniques and randomly selected queries.

6.2 Background

6.2.1 Observation Model

Our goal in this work is to estimate a user's preference point (denoted as vector w) with respect to a given low-dimensional embedding of items constructed such that distances between items are consistent with item similarities, where similar items are close together and dissimilar items are far apart. While many items (e.g., images) exist in their raw form in a high-dimensional space (e.g., pixel space), this low-dimensional representation of items and user preferences offers the advantage of simple Euclidean relationships that directly capture notions of preference and similarity, as well as mitigating the effects of the curse of dimensionality in estimating preferences. Specifically, we suppose user preferences can be captured via an *ideal point model* in which each item and user is represented using a common set of parameters in $\mathbb{R}^d$, and that a user's overall preference for a particular item decreases with the distance between that item and the user's ideal point w [146]. This means that any item placed exactly at the user would be considered "ideal" and would be the most preferred over all other items. Although this model can be applied to the situation where a particular item is sought, in general we do *not* assume the user point w to be co-located with any item.

The embedding of the items can be constructed through a training set of triplet comparisons (paired comparisons regarding similarity of two items to a third reference item) using one of several standard non-metric embedding techniques such as the Crowd Kernel Learning [121] or Stochastic Triplet Embedding methods [122]. In this study, we assume that such an embedding is given, presumably acquired through a large set of crowdsourced training triplet comparisons. We do not consider this training set to be part of the learning cost in measuring a pairwise search algorithm's efficiency, since our focus here is on efficiently choosing paired comparisons to search an *existing* embedding.

In this work, we assume a noisy ideal point model where the probability of a user located

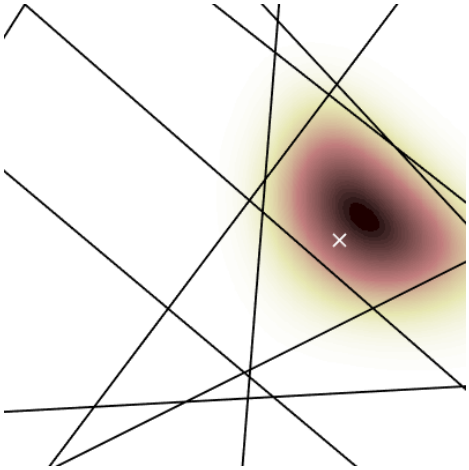


Figure 6.1: Paired comparisons between items can be thought of as a set of noisy hyperplane queries. In the high-fidelity case, this uniquely identifies a convex region of $\mathbb{R}^d$. In general, we have a posterior distribution which only approximates the shape of the ideal cell around the true user point, depicted with an x.

at w choosing item p over item q in a paired comparison is modeled using

$$P(p \prec q) = f(k_{pq}(\|w - q\|^2 - \|w - p\|^2)), \tag{6.1}$$

where $p \prec q$ denotes "item p is preferred to item q," $f(x) = 1/(1 + e^{-x})$ is the logistic function, and $k_{pq} \in [0, \infty)$ is the pair's *noise constant*, which represents roughly the signal-to-noise ratio of a particular query and may depend on the values of p and q. This type of logistic noise model is common in psychometrics literature and bears similarity to the Bradley–Terry model [139].

Note that eq. (6.1) can also be written as

$$P(p \prec q) = f(k_{pq}(a^T w - b)),$$

where $a = 2(p - q)$ and $b = \|p\|^2 - \|q\|^2$ encode the normal vector and threshold of a hyperplane bisecting items p and q. After a number of such queries, the response model in eq. (6.1) for each query can be multiplied to form a posterior belief about the location of w, as depicted in Figure 6.1.

Note that we allow the noise constant k_{pq} to differ for each item pair to allow for differing user behavior depending on the geometry of the items being compared. When $k_{pq} \to \infty$, this supposes a user's selection is made with complete certainty and cannot be erroneous. Conversely, $k_{pq} = 0$ corresponds to choosing items randomly with probability $1/2$. Varying k_{pq} allows for differing reliability when items are far apart versus when they are close together. Some concrete examples for setting this parameter are:

$$\text{constant} : k_{pq}^{(1)} = k_0, \tag{K1}$$

$$\text{normalized} : k_{pq}^{(2)} = k_0 \|a\|^{-1} = \frac{1}{2} k_0 \|(p-q)\|^{-1}, \tag{K2}$$

$$\begin{aligned} \text{decaying} : k_{pq}^{(3)} &= k_0 \exp(-\|a\|) \\ &= k_0 \exp(-2\|(p-q)\|). \end{aligned} \tag{K3}$$

6.2.2 Related Work

There is a rich literature investigating statistical inference from paired comparisons and related ordinal query types. However, many of these works target a different problem than considered here, such as constructing item embeddings [121], training classifiers [147], selecting arms of bandits [148], and learning rankings [149, 126, 150, 151, 152] or scores [153, 154] over items.

Paired comparisons have also been used for learning user preferences: [155] models user preferences as a vector, but preferences are modeled as linear weightings of item features rather than by relative distances between the user and items in an embedding, resulting in a significantly different model (e.g., monotonic) of preference. [156] considers the task of actively estimating the maximizer of an unknown preference function over items, while [136] and [157] actively approximate the preference function itself, the former study notably using information gain as a metric for selecting queries. Yet, these approaches are not directly comparable to our methods since they do not consider a setting where user points are assigned within an existing low-dimensional item embedding. [121] does consider the

same item embedding structure as our setting and actively chooses paired comparisons that maximize information gain for search, but only seeks *discrete items* within a fixed dataset rather than estimating a *continuous* preference vector as we do here. Furthermore we provide novel insights into selecting pairs via information gain maximization, and mainly treat information gain for pairwise search as a baseline in this work since our primary focus is instead on the development, analysis, and evaluation of alternative strategies inspired by this approach.

The most directly relevant prior work to our setting consists of the theory and algorithms developed in [158] and [116]. In [158], item pairs are selected in stages to approximate a Gaussian point cloud that surrounds the current user point estimate and dyadically shrinks in size with each new stage. In [116], previous query responses define a convex polytope in d dimensions (as in Figure 6.1), and their algorithm only selects queries whose bisecting hyperplanes intersect this feasible region. While this algorithm in its original form only produces a rank ordering over the embedding items, for the sake of a baseline comparison we extend it to produce a preference point estimate from the feasible region. Neither of these studies fundamentally models or handles noise in their active selection algorithms; slack variables are used in the user point estimation of [158] to allow for contradicting query responses, but the presence of noise is not considered when selecting queries. In an attempt to filter non-persistent noise (the type encountered in our work), [116] simply repeat each query multiple times and take a majority vote as the user response, but the items in the query pair are still selected using the same method as in the noiseless setting. Nevertheless, these methods provide an important baseline.

6.3 Query Selection

We now proceed to describe the pair selection problem in detail along with various theoretical and computational considerations. We show that the goal of selecting pairwise queries to minimize estimation error leads naturally to the strategy of information maximization and

subsequently to the development of our two novel selection strategies.

6.3.1 Minimizing Estimation Error

Let $W \in \mathbb{R}^d$ $(d \geq 2)$ denote a random vector encoding the user's preference point, assumed for the sake of analysis to be drawn from a uniform distribution over the hypercube $[-\frac{1}{2}, \frac{1}{2}]^d$ denoted by the prior density of $p_0(w)$. Let $Y_i \in \{0, 1\}$ denote the binary response to the i^{th} paired comparison involving items p_i and q_i, with $Y_i = 0$ indicating a preference for q_i and $Y_i = 1$ a preference for p_i. After i queries, we have the vector of responses $Y^i = \{Y_1, Y_2, \ldots Y_i\}$. We assume that each response Y_i is conditionally independent from previous responses Y^{i-1} when conditioned on preference W. Applying this assumption in conjunction with a recursive application of Bayes' rule, after i queries we have a posterior density of

$$p_i(w) \equiv p(w|Y^i) = \frac{p_0(w) \prod_{j=1}^{i} p(Y_j|w)}{p(Y^i)}, \tag{6.2}$$

where $p(Y_i|w)$ is given by the model in eq. (6.1). This logistic likelihood is log-concave, and since $p_0(w)$ is also log-concave we have from Section 2.1 that the posterior density given in eq. (6.2) is log-concave.

Suppose that after i queries, the posterior $p_i(w)$ is used to produce a Bayesian user point estimate $\widehat{W}_i$. We denote the mean squared error for this estimate by $\text{MSE}_i = \mathbb{E}_{W|Y^i}[\|W - \widehat{W}_i\|_2^2]$, which provides a direct measure of our estimation error and is a quantity we wish to minimize by adaptively selecting queries based on previous responses. One approach might be to greedily select an item pair such that MSE_{i+1} is minimized in expectation after the user responds. However, this would require both updating the posterior distribution and estimating MSE_{i+1} for each possible response over all item pairs. This would be very computationally expensive since under our model there is no closed-form solution for MSE_{i+1}, and so each such evaluation requires a "lookahead" batch of Monte Carlo samples from the posterior. Specifically, if S posterior samples are generated for each MSE_{i+1} evaluation over a candidate pool of M pairs at a computational cost of C per

sample generation, and MSE_{i+1} is estimated with $O(dS)$ operations per pair, this strategy requires $O((C+d)SM)$ computations to select each query. This is undesirable for adaptive querying settings where typically data sets are large (resulting in a large number of candidate pairwise queries) and queries need to be selected in or close to real-time.

Instead, consider the covariance matrix of the user point posterior after i queries, denoted as

$$\Sigma_{W|Y^i} = \mathbb{E}[(W - \mathbb{E}[W|Y^i])(W - \mathbb{E}[W|Y^i])^T|Y^i].$$

For the minimum mean squared error (MMSE) estimator, given by the posterior mean $\widehat{W}_i = \mathbb{E}[W|Y^i]$, we have

$$\text{MSE}_i = \text{Tr}(\Sigma_{W|Y^i}) \geq d|\Sigma_{W|Y^i}|^{\frac{1}{d}},$$

where the last inequality follows from the arithmetic-geometric mean inequality (AM–GM) [159]. This implies that a necessary condition for a low MSE is for the *posterior volume*, defined here as the determinant of the posterior covariance matrix, to also be low. Unfortunately, actively selecting queries that greedily minimize posterior volume is too computationally expensive to be useful in practice since this also requires a set of "lookahead" posterior samples for each candidate pair and possible response, resulting in a computational complexity of $O(((C+d^2)S+d^3)M)$ to select each query from the combined cost per pair of generating samples ($O(CS)$), estimating $\Sigma_{W|Y^i}$ ($O(d^2S)$), and calculating $|\Sigma_{W|Y^i}|$ ($O(d^3)$).

6.3.2 Information Theoretic Framework

By utilizing statistical tools from information theory, we can select queries that approximately minimize posterior volume (and hence tend to encourage low MSE) at a more reasonable computationally cost. Furthermore, an information theoretic approach provides convenient analytical tools which we use to provide performance guarantees for the query

selection methods we present.

Towards this end, we define the *posterior entropy* as the differential entropy of the posterior distribution after i queries:

$$h_i(W) \equiv h(W|y^i) = -\int_w p_i(w)\log_2(p_i(w))dw. \tag{6.3}$$

As we show in the following lemma, the posterior entropy of log-concave distributions is both upper and lower bounded by a monotonically increasing function of posterior volume, implying that low posterior entropy is both necessary and sufficient for low posterior volume, and hence a necessary condition for low MSE. The proofs of this lemma and subsequent results are provided in the supplementary material.

Lemma 6.3.1. *For a log-concave posterior distribution $p(w|Y^i)$ in $d \geq 2$ dimensions, where $c_d = e^2d^2/(4\sqrt{2}(d+2))$,*

$$\frac{d}{2}\log_2\frac{2|\Sigma_{W|Y^i}|^{\frac{1}{d}}}{e^2c_d} \leq h_i(W) \leq \frac{d}{2}\log_2(2\pi e|\Sigma_{W|Y^i}|^{\frac{1}{d}}).$$

This relationship between MSE, posterior volume, and posterior entropy suggests a strategy of selecting queries that minimize the posterior entropy after each query. Since the actual user response is unknown at the time of query selection, we seek to minimize the *expected* posterior entropy after a response is made, i.e., $\mathbb{E}_{Y_{i+1}}[h_{i+1}(W)|y^i]$. Using a standard result from information theory, we have $\mathbb{E}_{Y_i}[h_i(W)|y^{i-1}] = h_{i-1}(W) - I(W;Y_i|y^{i-1})$, where $I(W;Y_i|y^{i-1})$ is the mutual information between the location of the unknown user point and the user response, conditioned on previous responses. Examining this identity, we observe that selecting queries that minimize the expected posterior entropy is equivalent to selecting queries that maximize the mutual information between the user point and the user response, referred to here as the *information gain*.

In this setting, it is generally difficult to obtain sharp performance bounds for query selection via information gain maximization. Instead, we use information theoretic tools

along with Lemma 6.3.1 to provide a lower bound on MSE for *any* estimator and query selection scheme in a manner similar to [160] and [27]:

Theorem 6.3.2. *For any user point estimate given by $\widehat{W}_i$ after i queries, the MSE (averaged over user points and query responses) for any selection strategy is bounded by*

$$\mathbb{E}_{W,Y^i}\|W - \widehat{W}_i\|_2^2 \geq \frac{d2^{-2\frac{i}{d}}}{2\pi e}.$$

This result implies that the best rate of decrease in MSE one can hope for is exponential in the number of queries and slows down in a matter inversely proportional to the dimension, indicating quicker possible preference convergence in settings with lower dimensional embeddings. To estimate the information gain of a query, we can use the symmetry of mutual information to write

$$I(W; Y_i|y^{i-1}) = H(Y_i|y^{i-1}) - H(Y_i|W, y^{i-1}) \tag{6.4}$$

$$H(Y_i|y^{i-1}) = -\sum_{Y_i\in\{0,1\}} p(Y_i|y^{i-1}) \log_2 p(Y_i|y^{i-1}) \tag{6.5}$$

$$H(Y_i|w, y^{i-1}) = -\sum_{Y_i\in\{0,1\}} p(Y_i|w) \log_2 p(Y_i|w) \tag{6.6}$$

$$H(Y_i|W, y^{i-1}) = \mathbb{E}_{W|y^{i-1}}[H(Y_i|W, y^{i-1})]. \tag{6.7}$$

Unlike the greedy MSE and posterior volume minimization strategies, information gain estimation only requires a *single* batch of posterior samples at each round of query selection, which is used to estimate the discrete entropy quantities in eqs. (6.4) to (6.7). Eq. (6.4) can be estimated in $O(dS)$ operations per pair, resulting in a computational cost of $O(dSM)$ for selecting each query, which although more computationally feasible than the methods proposed so far is still likely prohibitive for highly accurate information gain estimates over a large pool of candidate pairs.

Because of these analytical and computational challenges, we develop two strategies

that mimic the action of maximizing information gain while being more analytically and computationally tractable, respectively. In the next section we present our first strategy, which we analyze for more refined upper and lower bounds on the number of queries needed to shrink the posterior to a desired volume. Then we introduce a second strategy which benefits from reduced computational complexity while still remaining theoretically coupled to maximizing information gain.

6.3.3 Strategy 1: Equiprobable, Max-variance

In developing an approximation for information gain maximization, consider the scenario where *arbitrary* pairs of items can be generated (unconstrained to a given dataset), resulting in a bisecting hyperplane parameterized by (a_i, b_i). In practice, such queries might correspond to the generation of synthetic items via tools such as generative adversarial networks [161]. With this freedom, we could consider an *equiprobable* query strategy where b_i is selected so that each item in the query will be chosen by the user with probability $\frac{1}{2}$. This strategy is motivated by the fact that the information gain of query i is upper bounded by $H(Y_i|y^{i-1})$, which is maximized if and only if the response probability is equiprobable [27].

To motivate the selection of query hyperplane directions, we define a query's *projected variance*, denoted as σ_i^2, as the variance of the posterior marginal in the direction of a query's hyperplane, i.e., $\sigma_i^2 = a_i^T \Sigma_{W|y^{i-1}} a_i$. This corresponds to a measure of how far away the user point is from the hyperplane query, in expectation over the posterior distribution. With this notation, we have the following lower bound on information gain for equiprobable queries.

Proposition 6.3.3. *For any "equiprobable" query scheme with noise constant k_i and projected variance σ_i^2, for any choice of constant $0 \leq c \leq 1$ we have*

$$I(W; Y_i|y^{i-1}) \geq \left(1 - h_b\Big(f\Big(\frac{ck_i\sigma_i}{2}\Big)\Big)\right)(1-c) =: L_{c,k_i}(\sigma_i),$$

where $h_b(p) = -p\log_2 p - (1-p)\log_2(1-p)$.

This lower bound is monotonically increasing with $k_i\sigma_i$ and achieves a maximum information gain of 1 bit at $k_i \to \infty$ and/or $\sigma_i \to \infty$ (with an appropriate choice of c). This suggests choosing a_i that *maximize projected variance* in addition to selecting b_i according to the equiprobable strategy. Together, we refer to the selection of equiprobable queries in the direction of largest projected variance as the equiprobable-max-variance scheme, or EPMV for short.

Our primary result concerns the expected number of comparisons (or query complexity) sufficient to reduce the posterior volume below a specified threshold set a priori, using EPMV.

Theorem 6.3.4. *For the EPMV query scheme with each selected query satisfying $k_i\|a_i\| \geq k_{min}$ for some constant $k_{min} > 0$, consider the stopping time $T_\varepsilon = \min\{i : |\Sigma_{W|y^i}|^{\frac{1}{d}} < \varepsilon\}$ for stopping threshold $\varepsilon > 0$. For $\tau_1 = \frac{d}{2}\log_2(\frac{1}{2\pi e\varepsilon})$ and $\tau_2 = \frac{d}{2}\log_2\frac{e^2 c_d}{2\varepsilon}$, we have*

$$\tau_1 \leq E[T_\varepsilon] \leq \tau_2 + \frac{\tau_2 + 1}{l(\tau_2)} - \frac{1}{l(\tau_2)}\int_0^{\tau_2} l(x)dx,$$

where $l(x) = L_{c,k_{min}}\left(\frac{2^{\frac{-x}{d}}}{\sqrt{2\pi e}}\right)$ for any constant $0 \leq c \leq 1$ as defined in Proposition 6.3.3. Furthermore, the lower bound is true for any query selection scheme.

This result follows from a martingale stopping-time analysis of the entropy at each query. Our next theorem presents a looser upper bound, but is more easily interpretable.

Theorem 6.3.5. *The EPMV scheme, under the same assumptions as in Theorem 6.3.4, satisfies*

$$\mathbb{E}[T_\varepsilon] = O\left(d\log\frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon k_{min}^2}\right)d^2\log\frac{1}{\varepsilon}\right).$$

Furthermore, for any query scheme, $\mathbb{E}[T_\varepsilon] = \Omega\left(d\log\frac{1}{\varepsilon}\right)$.

This result has a favorable dependence on the dimension d, and the upper bound can be interpreted as a blend between two rates, one of which matches that of the generic lower bound. The second term in the upper bound provides some evidence that our ability

to recover w worsens as $k_{\min}$ decreases. This is intuitively unsurprising since small $k_{\min}$ corresponds to the case where queries are very noisy. We hypothesize that the absence of such a penalty term in the lower bound is an artifact of our analysis, since increasing noise levels (i.e., decreasing $k_{\min}$) should limit achievable performance by any querying strategy. On the other hand, for asymptotically large k_i, we have the following corollary:

Corollary 6.3.1. *In the noiseless setting ($k_{min} \to \infty$), EPMV has optimal expected stopping time complexity for posterior volume stopping.*

Proof. When $k_{\min} \to \infty$, from Theorem 6.3.5 $\mathbb{E}[T_\varepsilon] = O\left(d \log \frac{1}{\varepsilon}\right)$; for any scheme, $\mathbb{E}[T_\varepsilon] = \Omega\left(d \log \frac{1}{\varepsilon}\right)$. □

Taken together, these results suggest that EPMV is optimal with respect to posterior volume minimization up to a penalty term which decreases to zero for large noise constants. While low posterior volume is only a necessary condition for low MSE, this result could be strengthened to an upper bound on MSE by bounding the condition number of the posterior covariance matrix, which is left to future work. Yet, as we empirically demonstrate in Section 6.4, in practice our methods are very successful in reducing MSE.

While EPMV was derived under the assumption of arbitrary hyperplane queries, depending on the application we may have to select a pair from a fixed pool of items in a given dataset. For this purpose we propose a metric for any candidate pair that, when maximized over all pairs in a pool, approximates the behavior of EPMV. For a pair with items p and q in item pool $\mathcal{P}$, let $a_{pq} = 2(p-q)$ and $b_{pq} = \|p\|^2 - \|q\|^2$ denote the weights and threshold parameterizing the bisecting hyperplane. We choose a pair that maximizes the utility function (for some $\lambda > 0$)

$$\eta_1(p, q; \lambda) = k_{pq}\sqrt{a_{pq}^T \Sigma_{W|Y^{i-1}} a_{pq}} - \lambda\left|\widehat{p}_1 - \frac{1}{2}\right| \tag{6.8}$$
$$\widehat{p}_1 = P(Y_i{=}1|Y^{i-1}) = \mathbb{E}_{W|Y^{i-1}}[f(k_{pq}(a_{pq}^T W - b_{pq}))].$$

This has the effect of selecting queries which are close to equiprobable and align with the

direction of largest variance, weighted by k_{pq} to prefer higher fidelity queries. While $\Sigma_{W|Y^{i-1}}$ can be estimated once from a batch of posterior samples, $\widehat{p}_1$ must be estimated for each candidate pair in $O(dS)$ operations, resulting in a computational cost of $O(dSM)$ which is on the same order as directly maximizing information gain. For this reason, we develop a second strategy that approximates EPMV while significantly reducing the computational cost.

6.3.4 Strategy 2: Mean-cut, Max-variance

Our second strategy is a *mean-cut* strategy where b_i is selected such that the query hyperplane passes through the posterior mean, i.e. $a_i^T \mathbb{E}[W|Y^{i-1}] - b_i = 0$. For such a strategy, we have the following proposition:

Proposition 6.3.6. *For mean-cut queries with noise constant k_i and projected variance σ_i^2 we have*

$$\left|p(Y_i|y^{i-1}) - \frac{1}{2}\right| \leq \frac{e-2}{2e} + \frac{\ln 2}{k_i\sigma_i}$$

$$\textit{and, } I(W;Y_i|y^{i-1}) \geq h_b\left(\frac{1}{e} - \frac{\ln 2}{k_i\sigma_i}\right) - \frac{\pi^2(\log_2 e)}{3k_i\sigma_i}.$$

For large projected variances, we observe that $|p(Y_i|y^{i-1}) - \frac{1}{2}| \lessapprox 0.14$, suggesting that mean-cut queries are somewhat of an approximation to equiprobable queries in this setting. Furthermore, notice that the lower bound to information gain in Proposition 6.3.6 is a monotonically increasing function of the projected variance. As $\sigma_i \to \infty$, this bound approaches $h_b(1/e) \approx 0.95$ which is nearly sharp since a query's information gain is upper bounded by 1 bit. This implies some correspondence between maximizing a query's information gain and maximizing the projected variance, as was the case in EPMV. Hence, our second strategy selects *mean-cut, maximum variance* queries (referred to as MCMV) and serves as an approximation to EPMV while still maximizing a lower bound on information gain.

Algorithm 3: Pairwise search with noisy comparisons

Input: item set $\mathcal{X}$, parameters S, β, λ
  $\mathcal{P} \leftarrow$ set of all pairwise queries from items in $\mathcal{X}$
  $\widetilde{W}_0, \mu_0, \Sigma_0 \leftarrow$ initialize from samples of prior
  for $i = 1$ **to** T **do**
    $\mathcal{P}_\beta \leftarrow$ uniformly downsample $\mathcal{P}$ at rate $0 < \beta \leq 1$
    InfoGain: $p_i, q_i \leftarrow \underset{p,q \in \mathcal{P}_\beta}{\arg\max}\ \eta_0(p, q; \widetilde{W}_{i-1})$
    EPMV: $p_i, q_i \leftarrow \underset{p,q \in \mathcal{P}_\beta}{\arg\max}\ \eta_1(p, q; \lambda, \widetilde{W}_{i-1})$
    MCMV: $p_i, q_i \leftarrow \underset{p,q \in \mathcal{P}_\beta}{\arg\max}\ \eta_2(p, q; \lambda, \mu_{i-1}, \Sigma_{i-1})$
    $y_i \leftarrow \text{PairedComparison}(p_i, q_i)$, $y^i \leftarrow y_i \cup y^{i-1}$.
    $\widetilde{W}_i \leftarrow$ batch of S samples from posterior $W|Y^i$
    $\mu_i, \Sigma_i \leftarrow \text{Mean}(\widetilde{W}_i), \text{Covariance}(\widetilde{W}_i)$
    $\widehat{W}_i \leftarrow \mu_i$
  end for
Output: user point estimate $\widehat{W}_T$

For implementing MCMV over a fixed pool of pairs (rather than arbitrary hyperplanes), we calculate the orthogonal distance of each pair's hyperplane to the posterior mean as $|a_{pq}^T \mathbb{E}[W|Y^{i-1}] - b_{pq}|/\|a_{pq}\|_2$ and the projected variance as $a_{pq}^T \Sigma_{W|Y^{i-1}} a_{pq}$. We choose a pair that maximizes the following function which is a tradeoff (tuned by $\lambda > 0$) between minimizing distance to the posterior mean, maximizing noise constant, and maximizing projected variance:

$$\eta_2(p, q; \lambda) = k_{pq} \sqrt{a_{pq}^T \Sigma_{W|Y^{i-1}} a_{pq}} - \lambda \frac{|a_{pq}^T \mathbb{E}[W|Y^{i-1}] - b_{pq}|}{\|a_{pq}\|_2}. \tag{6.9}$$

This strategy is attractive from a computational standpoint since the posterior mean $\mathbb{E}[W|Y^{i-1}]$ and covariance $\Sigma_{W|Y^{i-1}}$ can be estimated *once* in $O(d^2 S)$ computations, and subsequent calculation of the hyperplane distance from mean and projected variance requires only $O(d^2)$ computations per pair. Overall, this implementation of the MCMV strategy has a computational complexity of $O(d^2(S + M))$, which scales more favorably than both the information gain maximization and EPMV strategies.

We unify the information gain (referred to as InfoGain), EPMV, and MCMV query

selection methods under a single framework described in Algorithm 3. At each round of querying, a pair is selected that maximizes a utility function $\eta(p, q)$ over a randomly downsampled pool of candidates pairs, with $\eta_0(p, q) \equiv I(W; Y_i|y^{i-1})$ for InfoGain and η_1 from eq. (6.8) and η_2 from eq. (6.9) denoting the utility functions of EPMV and MCMV, respectively. We include a batch of posterior samples denoted by $\widetilde{W}$ as an input to η_0 and η_1 to emphasize their dependence on posterior sampling, and add mean and covariance inputs to η_2 since once these are estimated, MCMV requires no additional samples to select pairs. For all methods, we estimate the user point as the mean of the sample batch since this is the MMSE estimator.

6.4 Results

To evaluate our approach, we constructed a realistic embedding (from a set of training user-response triplets) consisting of multidimensional item points and simulated our pairwise search methods over randomly generated preference points and user responses. We constructed an item embedding of the Yummly `Food-10k` dataset of [141, 135], consisting of 958,479 publicly available triplet comparisons assessing relative similarity among 10,000 food items. The item coordinates are derived from the crowdsourced triplets using the popular probabilistic multidimensional scaling algorithm of [121] and the implementation obtained from the NEXT project[2].

6.4.1 Methods Comparison

We compare InfoGain, EPMV, and MCMV as described in Algorithm 3 against several baseline methods:

Random: pairs are selected uniformly at random and user preferences are estimated as the posterior mean.

GaussCloud-Q: pairs are chosen to approximate a Gaussian point cloud around the prefer-

[2] http://nextml.org

ence estimate that shrinks dyadically over Q stages, as detailed in [158].

ActRank-Q: pairs are selected that intersect a feasible region of preference points and queried Q times; a majority vote is then taken to determine a single response, which is used with the pair hyperplane to further constrain the feasible set [116]. Since the original goal of the algorithm was to rank embedding items rather than estimate a continuous preference point, it does not include a preference estimation procedure; in our implementation we estimate user preference as the Chebyshev center of the feasible region since it is the deepest point in the set and is simple to compute [159].

In each simulation trial, we generate a point W uniformly at random from the hypercube $[-1, 1]^d$ and collect paired comparisons using the item points in our embedding according to the methods described above. The response probability of each observation follows eq. (6.1) (referred to herein as the "logistic" model), using each of the three schemes for choosing k_{pq} described in (K1) through (K3). In each scheme we optimized the value of k_0 over the set of training triplets via maximum-likelihood estimation according to the logistic model. We use the Stan Modeling Language [162] to generate posterior samples when required, since our model is log-concave and therefore is particularly amenable to Markov chain Monte Carlo methods [163].

Note that unlike GaussCloud-Q and ActRank-Q, the Random, InfoGain, EPMV, and MCMV methods directly exploit a user response model in the selection of pairs and estimation of preference points, which can be advantageous when a good model of user responses is available. Below we empirically test each method in this *matched* scenario, where the noise type (logistic) and the model for k_{pq} (e.g., "constant", "normalized", or "decaying") are revealed to the algorithms. We also test a *mismatched* scenario by generating response noise according to a non-logistic response model while the methods above continue to calculate the posterior as if the responses were logistic. Specifically, we generate responses

according to a "Gaussian" model

$$y_i = \text{sign}(k_{pq}(a_i^T w - b_i) + Z) \qquad Z \sim \mathcal{N}(0, 1),$$

where k_0 *and* the model for k_{pq} are selected using maximum-likelihood estimation on the training triplets.

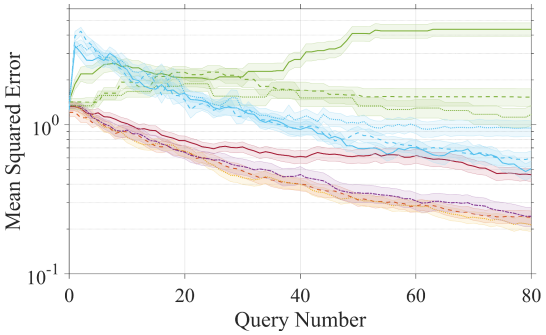
6.4.2 Mean Squared Error Evaluation

The left column of Figure 6.2 plots the MSE of each method's estimate with respect to the ground truth location over the course of a pairwise search run. In the matched model case of Figure 6.2a, our strategies outperform Random, ActRank-Q, and GaussCloud-Q for multiple values of Q by a substantial margin. Furthermore, both of our strategies performed similarity to InfoGain, corroborating their design as information maximization approximations. Note that Random outperforms the other baseline methods, supporting the use of Bayesian estimation in this setting (separately from the task of active query selection). Although mismatched noise results in decreased performance overall in Figure 6.2c, the same relative trends between the methods as in Figure 6.2a are evident.

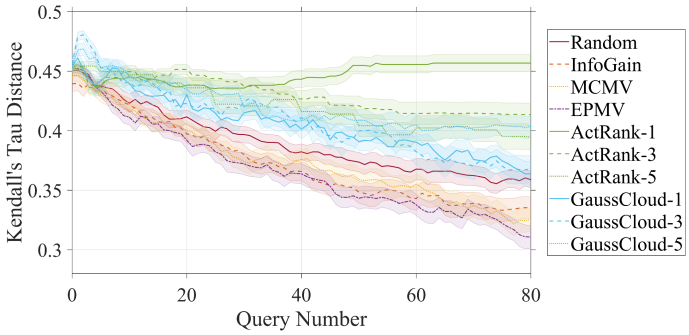
6.4.3 Item Ranking Evaluation

We also consider each method's performance with respect to ranking embedding items in relation to a preference point. For each trial, a random set of 15 items is sampled from the embedding without replacement and ranked according to their distance to a user point estimate. This ranking is compared to the ground truth ranking produced by the true user point by calculating a normalized Kendall's Tau distance, which is 0 for identical rankings and 1 for completely discordant rankings [116]. This metric measures performance in the context of a recommender system type task (a common application of preference learning) rather than solely measuring preference estimation error. This metric is depicted in the right

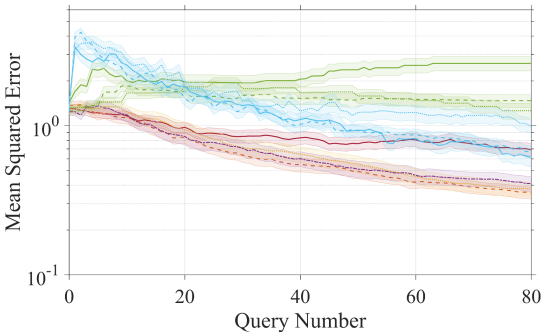
column of Figure 6.2, for the matched model case in Figure 6.2b and mismatched case in Figure 6.2d. The same trends as observed in MSE analysis occur, with our strategies performing similarly to InfoGain and outperforming all other methods. This is a particularly noteworthy result in that our method produces more accurate rankings than ActRank-Q, which to our knowledge is the state-of-the-art method in active embedding ranking.



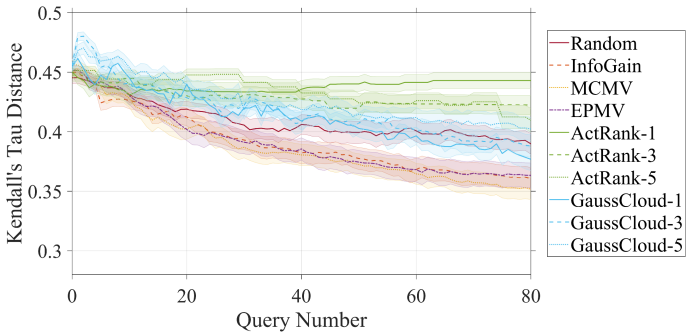
(a) Estimation error: matched logistic noise, $d = 4$



(b) Ranking performance: matched logistic noise, $d = 4$



(c) Estimation error: mismatched Gaussian noise, $d = 4$



(d) Ranking performance: mismatched Gaussian noise, $d = 4$

Figure 6.2: Performance evaluation over 80 simulated search queries, averaged over 50 trials per method and plotted with $\pm$ one standard error. (Left Column) MSE. (Right Column) for each trial, a batch of 15 items was uniformly sampled without replacement from the dataset, and the normalized Kendall's Tau distance (lower distance is better) was calculated between a ranking of these items by distance to the ground truth preference point and a ranking by distance to the estimated point. To get an unbiased estimate, this metric was averaged over 1000 batches per trial, and error bars calculated with respect to the number of trials. (Top Row) "normalized" logistic model with matching noise in $d = 4$. (Bottom Row) "decaying" logistic model with mismatched Gaussian "normalized" noise in $d = 4$. Additional plots testing a wider selection of parameters are available in the supplement. Overall, our new strategies (EPMV, MCMV) outperform existing methods and also perform comparably to information gain maximization (InfoGain), which they were designed to approximate.

6.5 Extension to Ideal Point Estimation with Dynamics

In this section, we discuss an extension of pairwise search to the case where the ideal point evolves over time according to unknown dynamics. Many applications for pairwise search can be naturally extended to include time-varying dynamics and system identification. For example, in a simple two- or three-dimensional setting one might wish to triangulate an object's location using an array of sensors, where the only available information is which of any two given sensors the object is closer to (since exact sensor range measurements might be unavailable or too noisy to be utilized directly). Rather than being static and a known object type, the localized object may be one of several vehicle types navigating along a path. In this case, one may wish to jointly estimate the vehicle's position, velocity, and acceleration (state estimation), as well as identify the vehicle type from its dynamics (system identification). In higher dimensional settings such as recommender systems [155, 156, 164, 149, 126, 150, 151, 152], a user's preferences between pairs of items may change with time, and these changes may be characteristic of one of several user phenotypes. While the task of selecting pairs and estimating the state vector has been studied in the static case [4, 165, 166, 158], the time-varying setting has not been addressed.

Mathematically, we consider the problem of tracking the evolution of a vector $w \in \mathbb{R}^d$ which varies over time according to an unknown dynamics model f as

$$x_t = \begin{bmatrix} w_t \\ v_t \end{bmatrix}, \quad x_{t+1} = f(x_t) + \nu_{t+1}, \quad x_0 \sim P_0, \tag{6.10}$$

where $v_t \in \mathbb{R}^l$ is a vector of latent state variables (e.g., velocity and acceleration) which together with w_t comprise the state vector $x_t \in \mathbb{R}^{d+l}$. The dynamics are perturbed by *innovation noise* $\nu_{t+1} \sim \mathcal{N}(0, R)$ with known covariance R. We assume that f is drawn from a finite set of candidate dynamics models $\mathcal{F} = \{f_1, f_2, \dots f_K\}$ with prior distribution $p_f(f_i)$, and that the initial state is drawn from a known prior distribution P_0.

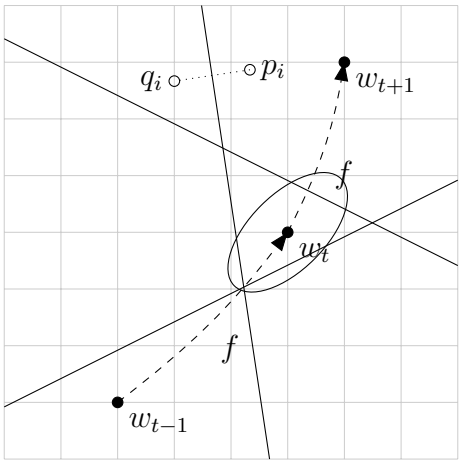


Figure 6.3: We consider the problem of using paired comparisons to jointly infer the trajectory of the state w_0^M and the dynamics model $f \in \mathcal{F}$ that describes its evolution.

At each time step $t = 0, \ldots, M$, we can access the state only through binary measurements consisting of paired comparisons that indicate which of two landmark points w_t is closer to. Our task is to actively select a sequence of paired comparisons between landmark pairs to jointly estimate the state trajectory x_0^M and identify the true dynamics model f.

In Section 6.5.1, we extend ideas for active selection of paired comparisons from the static setting to the time-varying setting, and use these insights in Section 6.5.2 to describe our Bayesian particle filter-based method for measurement selection, state tracking, and system identification. We illustrate the operation of our method with an example in Section 6.5.3, and evaluate its performance in Section 6.5.4.

6.5.1 Measurement Selection

At each time step t we take a measurement of the form $\|p_t - w_t\| \gtrless \|q_t - w_t\|$, where p_t, q_t are selected from a known set of landmark points $\mathcal{X} \subset \mathbb{R}^d$. Geometrically, this paired comparison indicates that w_t lies on one side of the hyperplane bisecting points p_t and q_t, as illustrated in Figure 6.3. This hyperplane is defined by normal vector $a_t = \frac{p_t - q_t}{\|p_t - q_t\|}$ and intercept $b_t = \frac{\|p_t\|^2 - \|q_t\|^2}{2\|p_t - q_t\|}$. However, in many practical applications we can access only *noisy* measurements, where comparisons are more likely to be erroneous when w_t is equidistant from the landmark points (i.e., close to the bisecting hyperplane). To represent this type of

observation noise, we denote the t^{th} measurement by $Y_t \in \{0, 1\}$, where $Y_t = 1$ indicates that w_t is closer to p_t and $Y_t = 0$ indicates that w_t is closer to q_t. We then model Y_t with the logistic likelihood

$$p(Y_t = 1 \,|\, w_t) = \frac{1}{1 + e^{-k(a_t^T w_t - b_t)}}, \tag{6.11}$$

where k represents the signal-to-noise ratio of the measurements. We assume that Y_t depends only on w_t; that is, letting $A \perp\!\!\!\perp B \mid C$ denote that A is conditionally independent of B given C, we have $Y_t \perp\!\!\!\perp w_u \mid w_t$ for $u \neq t$, $Y_t \perp\!\!\!\perp v_0^M \mid w_t$, and $Y_t \perp\!\!\!\perp f \mid w_t$. We adopt a Bayesian framework, representing our knowledge of the state x_t and dynamics model f by the posterior densities $p\left(x_t \,|\, y_0^{t-1}\right)$ and $p\left(f_i \,|\, y_0^{t-1}\right)$, where y_t denotes an observed instantiation of Y_t.

A natural question arising in this stochastic measurement model is how to select the measured paired comparison at each time step. In the static setting, as we described earlier in this chapter it has been shown that some measurements become more informative than others as w is localized, and dramatic improvements in inference are possible by *adaptively* selecting landmark points [158, 4]. In the time-varying setting considered here, at each time step we wish to select the landmark points (p_t, q_t) defining measurement Y_t that provide the most information about the trajectory x_0^M and the dynamics model f. We propose a similar approach as earlier in the chapter by selecting paired comparisons that maximize the information gain [20] each measurement provides about both the state trajectory and dynamics model, defined as the mutual information between the measurement Y_t and unknown trajectory x_0^M and dynamics f, conditioned on the previous measurements y_0^{t-1}:

$$(p_t, q_t) = \underset{p,q \in \mathcal{X}}{\arg\max}\ I\left(Y_t\,;\, x_0^M, f \,|\, y_0^{t-1}\right). \tag{6.12}$$

Intuitively, this quantity represents the amount a paired comparison decreases our uncertainty about the trajectory and dynamics model.

Because we seek to jointly infer the trajectory and dynamics model, it is at first unclear

whether one should select measurements that are more informative about the trajectory or about the model. However, the conditional independence of the measurement model in eq. (6.11) greatly simplifies this design choice: applying the chain rule of mutual information [27] to eq. (6.12) and simplifying using the conditional independences admitted by our model yields

$$\begin{aligned} I(Y_t; x_0^M, f \mid y_0^{t-1}) &= I(Y_t; w_0^M, v_0^M, f \mid y_0^{t-1}) \\ &= I(Y_t; w_0^M \mid y_0^{t-1}) + I(Y_t; v_0^M | w_0^M, y_0^{t-1}) + I(Y_t; f \mid w_0^M, v_0^M, y_0^{t-1}) \\ &= I(Y_t; w_0^M \mid y_0^{t-1}) \\ &= I(Y_t; w_t \mid y_0^{t-1}) + I(Y_t; w_0^{t-1}, w_{t+1}^M | w_t, y_0^{t-1}) \\ &= I(Y_t; w_t \mid y_0^{t-1}). \end{aligned}$$

Therefore, jointly maximizing the information gain with respect to the entire state trajectory and underlying dynamics model is equivalent to simply selecting paired comparisons that maximize the information gain about w_t.

6.5.2 Methods

Unfortunately, the measurement likelihood in eq. (6.11) does not admit a closed-form expression for the information gain $I(Y_t; w_t \mid y_0^{t-1})$ of a candidate pair, and approximating it with samples from the posterior $p(w_t|y_0^{t-1})$ is computationally prohibitive when evaluating a large pool of pairs. Instead, as in the static case we approximate the action of maximizing information gain by using the MCMV selection strategy, selecting the pair whose bisecting hyperplane cuts through the posterior mean in the direction of maximum variance. Specifically, we select measurements by evaluating an acquisition function for each candidate pair in a downsampled pair pool and selecting the maximizing pair, as described in Algorithm 4. As described previously, this procedure has a computational complexity that scales favorably with the number of candidate pairs, and is a provable approximation to information gain

maximization.

To select measurement pairs using MCMV, we need to evaluate the posterior mean $\mu_t \coloneqq \mathbb{E}_{w_t}\left[w_t|y_0^{t-1}\right]$ and covariance $\Sigma_t \coloneqq \mathbb{E}_{w_t}\left[(w_t-\mu_t)(w_t-\mu_t)^T|y_0^{t-1}\right]$, which can be computed as

$$\mu_t = \sum_i \mathbb{E}_{w_t}\left[w_t|f_i, y_0^{t-1}\right] p(f_i|y_0^{t-1}) \tag{6.13}$$

$$\Sigma_t = \sum_i \mathbb{E}_{w_t}\left[w_t w_t^T|f_i, y_0^{t-1}\right] p(f_i|y_0^{t-1}) - \mu_t\mu_t^T. \tag{6.14}$$

After taking a measurement, we update the posteriors over x_t and f, estimate the state as the posterior mean

$$\widehat{x}_t \coloneqq \mathbb{E}[x_t|y_0^{t-1}] = \sum_i \mathbb{E}_{x_t}\left[x_t|f_i, y_0^{t-1}\right] p(f_i|y_0^{t-1}), \tag{6.15}$$

and update the dynamics model posterior as

$$\begin{aligned} p(f_i|y_0^t) &= \frac{p(y_t|f_i, y_0^{t-1})p(f_i|y_0^{t-1})}{p(y_t|y_0^{t-1})} \\ &= \frac{\mathbb{E}_{w_t}\left[p(y_t|w_t)|f_i, y_0^{t-1}\right] p(f_i|y_0^{t-1})}{\sum_j \mathbb{E}_{w_t}\left[p(y_t|w_t)|f_j, y_0^{t-1}\right] p(f_j|y_0^{t-1})}. \end{aligned} \tag{6.16}$$

We observe that to calculate each of these quantities we can simply track a separate state posterior $p(x_t|y_0^{t-1}, f_i)$ for each candidate dynamical system $i = 1, \ldots, K$, from which we can compute the necessary expected values.

Because our pairwise measurement process eq. (6.11) is nonlinear, we cannot use the closed form updates of the Kalman filter to track each posterior $p(x_t|y_0^{t-1}, f_i)$. Instead, we use the *particle filter*, which allows us to incorporate both the nonlinear likelihood and an arbitrary (potentially nonlinear) candidate dynamics models [167]. In the particle filter framework, the required probability distributions are represented by Monte Carlo particles which can be propagated through the candidate dynamics models f_i. We use N particles to

Algorithm 4: MCMV-DF using particle filter

Draw N particles from $p_0(x)$ for each candidate dynamics model
for $t = 1, \ldots, M$ **do**
  Estimate state $\widehat{x}_t$ using eq. (6.15)
  Estimate μ_t and Σ_t from all particles using eq. (6.13) through eq. (6.14)
  $\mathcal{P}_\beta \leftarrow$ downsample set of candidate pairs at rate β
  $(p_t, q_t) \leftarrow \arg\max_{\mathcal{P}_\beta} \sqrt{a_{pq}^T \Sigma_t a_{pq}} - |a_{pq}^T \mu_t - b_{pq}|$
  $y_t \leftarrow \text{PairedComparison}(p_t, q_t)$, $y_0^t \leftarrow y_t \cup y_0^{t-1}$
  for $i = 1, \ldots, K$ **do**
    Resample particles using likelihood eq. (6.11)
    Propagate particles through dynamics eq. (6.10)
  end for
  Update $p(f_i|y_0^t)$ using eq. (6.16)
end for
$f \leftarrow \arg\max_{f_i} p(f_i)$

represent the state posterior associated with each candidate dynamics model, resulting in a total of NK tracked particles. We present our algorithm in its entirety, called *mean-cut max-variance dynamic filtering* (MCMV-DF), in Algorithm 4.

6.5.3 Explanatory Example

Figure 6.4 illustrates our approach with a stylized numerical example. We track a point $w \in \mathbb{R}^2$ evolving purely along the *horizontal* axis according to a spring-like system, with latent state $v \in \mathbb{R}^4$ representing velocity and acceleration in each dimension. We consider $K = 2$ candidate dynamics models: the true dynamical system f_h, and a similar system f_v that evolves purely along the *vertical* axis. We run MCMV-DF for $M = 100$ time steps and observe the inferred state and dynamics model posteriors. As MCMV-DF converges to correctly identify the true (horizontal) dynamical system f_h at approximately $t = 40$ (as shown in (d) by the posterior probability $p(f_h|y_0^{t-1})$ approaching unity), the vertical position estimate becomes more accurate. This is reflected in both the tight distribution of probability mass around $\widehat{w}_v(t)$ in (a) and (c) and the closer to vertical orientation of the hyperplanes corresponding to measurements y_{25} and y_{75} in (a). This vertical orientation

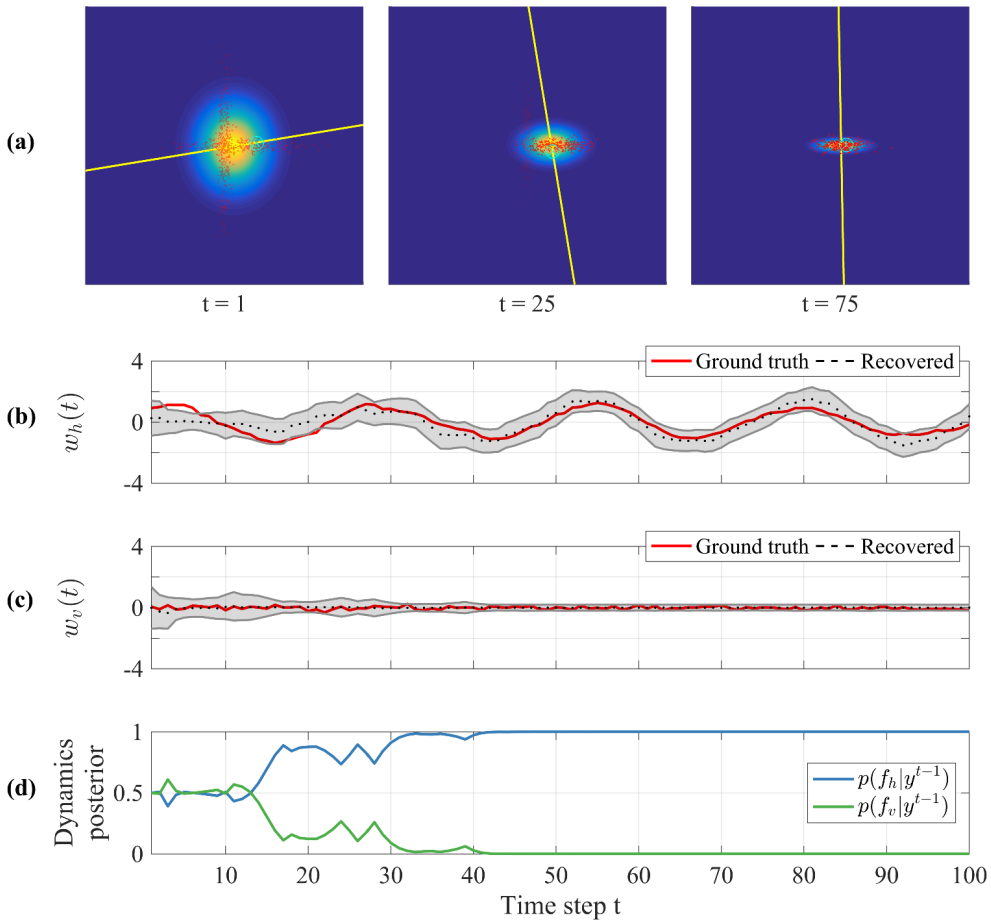


Figure 6.4: Stylized demonstration of MCMV-DF. (a) Posterior position distributions at three time instants. Surface plots: state posterior $p(w_t|y_0^{t-1})$; red circles: particles representing $p(w_t|y_0^{t-1}, f_i)\ \forall\ f_i$ with opaqueness representing $p(f_i|y_0^{t-1})$; cyan target: true position w_t; yellow line: hyperplane corresponding to selected measurement (p_t, q_t). (b-c) True trajectory and recovered marginal posterior $p(w_t|y_0^{t-1})$ for horizontal and vertical components of position; shaded region corresponds to 95% confidence interval on posterior. (d) Posterior over dynamics models $p(f_h|y_0^{t-1})$ and $p(f_v|y_0^{t-1})$.

maximizes variance after the trajectory has been identified as purely along the horizontal axis. As the measurements begin to focus on accurately estimating the horizontal position, the horizontal position estimates also become more accurate, as displayed in (b).

6.5.4 Numerical Experiments

In this section, we demonstrate the performance of MCMV-DF with simulations on synthetic data, evaluating the effects of observation and innovation noise as well as the number

of candidate dynamical systems K on the accuracy of trajectory estimation and system identification. In both experiments, we randomly generate an initial state $x_0 \sim \mathcal{N}(\mathbf{0}, I)$ with dimensionality $d = l = 4$ and compute its trajectory using eq. (6.10) and f with $R = \sigma^2 I$ for various settings of σ^2. We generate 1500 landmark points, distributed in $\mathbb{R}^4$ as $\mathcal{N}(0, \sigma_p^2 I)$ with $\sigma_p^2 = 9$, and use the downsampling rate $\beta = 0.01$ when selecting landmark points for measurements.

In each trial, we generate K random linear dynamics models $f_1, \ldots, f_K$ by placing $d + l$ eigenvalues in complex conjugate pairs on the unit circle (making the resulting systems marginally stable) at angles distributed as $\theta \sim U\left[\frac{\pi}{6}, \frac{\pi}{3}\right]$ (controlling the velocity of the resulting trajectories), with random orthogonal eigenvectors. We arbitrarily select one of the K candidate dynamics models as the true system.

In Figure 6.5, we evaluate the accuracy of tracking and identification with four different noise levels. To set the observation noise level, we vary the signal-to-noise constant k in eq. (6.11): "low" observation noise corresponds to $k = 10$ (resulting in approximately 5% incorrect comparisons), and "high" observation noise corresponds to $k = 1$ (resulting in approximately 25% incorrect comparisons). The "low" and "high" values of innovation noise are $\sigma^2 = 10^{-3}$ and $\sigma^2 = 10^{-2}$, respectively. Figure 6.5 shows the tracking error of the *entire* state x, $\|x_t - \widehat{x}_t\|_2$, and posterior probability of the true dynamics model $p(f|y_0^{t-1})$ over a horizon of $M = 40$ time steps. We observe that MCMV-DF successfully identifies the true dynamics system in a modest number of measurements and quickly achieves low state estimation error. Increasing the observation and innovation noise reduces estimation accuracy and increases the number of time steps required to identify the true dynamical system, but our method still tracks the state and eventually recovers the correct dynamics model.

In Figure 6.6, we evaluate the effect of the number of candidate dynamics models K on MCMV-DF's performance with fixed noise levels $k = 3$ (resulting in approximately 15% incorrect comparisons) and $\sigma^2 = 5 \times 10^{-3}$. We observe that the higher complexity

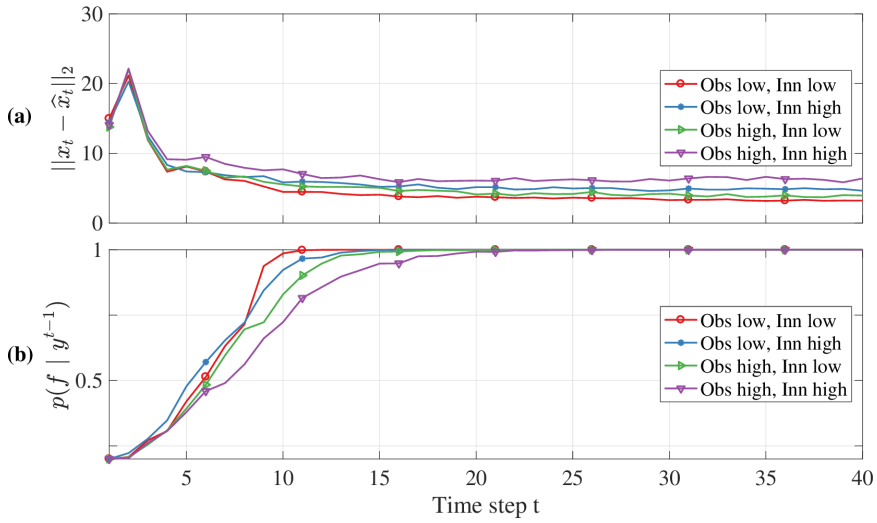


Figure 6.5: Tracking performance as observation noise ("obs") and innovation noise ("inn") levels change; each point shows the median over 200 trials. (a) Trajectory reconstruction accuracy, shown as error $\|x_t - \widehat{x}_t\|_2$ at each time step. (b) Dynamical system identification, shown as posterior probability of true system $p(f|y_0^{t-1})$.

of the set of candidate systems $\mathcal{F}$ resulting from increasing K makes the problem harder, reducing tracking accuracy and increasing the number of time steps required to identify the true dynamical system; however, system recovery is still possible.

6.6 Discussion

Our simulations in Section 6.4 demonstrate that both InfoGain approximation methods, EPMV and MCMV, significantly outperform the state-of-the-art techniques in active preference estimation in the context of low-dimensional item embeddings with noisy user responses, and perform similarity to InfoGain, the method they were designed to approximate. This is true even when generating noise according to a different model than the one used for Bayesian estimation. These empirical results support the theoretical connections between EPMV, MCMV, and InfoGain presented in this study, and suggest that the posterior volume reduction properties of EPMV may in fact allow for MSE reduction guarantees.

These results also highlight the attractiveness of MCMV, which proved to be a top performer in embedding preference learning yet is computationally efficient and simple to

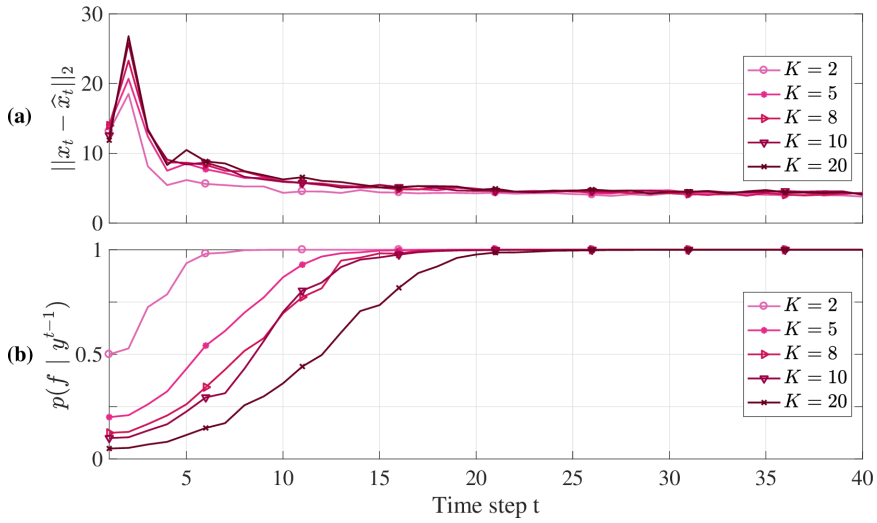


Figure 6.6: Tracking performance as the number of candidate dynamics models K increases; each point shows the median over 200 trials. (a) Trajectory reconstruction accuracy, shown as error $\|x_t - \widehat{x}_t\|_2$ at each time step. (b) Dynamical system identification, shown as posterior probability of true system $p(f|y_0^{t-1})$.

implement. This technique may also find utility as a subsampling strategy in supervised learning settings with implicit pairwise feedback, such as in [164]. Furthermore, although in this work pairs were drawn from a fixed embedding, MCMV is easily adaptable to continuous item spaces that allow for generative construction of new items to compare. This is possible in some applications, such as facial composite generation for criminal cases [168] or in evaluating foods and beverages, where we might be able to generate nearly arbitrary stimuli based on the ratios of ingredients [169].

Additionally, the results in Section 6.5.4 evaluating a dynamical systems problem extension demonstrate MCMV-DF's ability to successfully estimate the state trajectory from intelligently selected paired comparisons and discern between multiple candidate dynamics models. Further study is warranted to evaluate MCMV-DF's performance in real-world systems, including exploring the possibility of an *uncountable* set of candidate dynamical systems $\mathcal{F}$ consisting of continuously parameterized dynamics models, which would enable general system identification.

CHAPTER 7

FEEDBACK CODING FOR ACTIVE LEARNING

So far this in thesis, we have discussed several areas of overlap between IML and coding theory, directly applied PM to a class of query selection problems in HCI, and broadly applied tools in information theory to leverage IML query structures for the design of computationally efficient approximations to information gain maximization. In this chapter,[1] we synthesize these efforts by directly applying principles from PM for informative example selection in general active learning problems, while utilizing specific query structures for computationally efficient approximations. Specifically, the main contributions of this chapter are a formulation of general active learning problems in terms of a feedback communications system, a new PM-style coding scheme designed specifically for this framework, and a demonstration of this approach through its application to active logistic regression. Although there exists a large literature studying the intersection of information theory with machine learning [17] and specifically active learning [170], there remain open questions about the best ways to directly leverage techniques in channel coding for active example selection, as we explore here.

To motivate this approach, we first examine active learning through the lens of feedback channel coding by identifying communications system components, including a deterministic encoder, noisy channel, channel input constraints, and capacity-achieving distribution. With these components identified, we show how typical structural constraints in active learning problems prevent the direct application of existing feedback coding approaches such as posterior matching [31]. We address this challenge by proposing *Approximate Posterior Matching* (APM), an optimal transport-based active learning scheme that extends posterior

[1] This chapter is in collaboration with Dr. Matthieu Bloch and Dr. Christopher Rozell. CR and MB provided project guidance. GC was the lead author of the associated publication in [6]. CR supervised the project.

matching to account for the type of encoder constraints found in active learning problems. To demonstrate the power of this approach, we apply APM to Bayesian logistic regression, a popular model in active learning. We identify the communication system components in logistic regression, derive a corresponding APM selection scheme (APM-LR), provide analytical results concerning each selected example's information content, and empirically demonstrate on several datasets how APM-LR attains a sample complexity comparable to other active logistic regression methods at a reduced computational cost. While this example scenario highlights the capabilities of APM as a specific data selection method, the feedback communications framework we develop provides a unified approach for designing and analyzing active learning systems in general.

7.1 Related Work

Bayesian active learning methods are intimately related to concepts in information and coding theory, and the intersection between these topics has a long history rooted in the study of sequential design of experiments [41, 42] and active hypothesis testing [38]. Since this early work, direct estimation and maximization of information gain has emerged as a popular active learning method [20], and has been approximated for computational tractability [171]. More recently, [170] have studied the direct application of an information-theoretic active hypothesis testing method to active learning problems. This method is limited to discriminating between a finite number of hypotheses (as opposed to estimating arbitrary model parameters) and to our knowledge has not been applied to popular machine learning models such as logistic regression. Other works have described at a high-level the similarities between active learning and coding with feedback over a noisy channel but do not exploit this observation to leverage existing coding schemes for example selection [19, 172].

Posterior matching [22, 31] has been applied to tasks beyond telecommunications such as brain-computer interfacing [23, 50] and aircraft path planning [52], but has limited

application to example selection in active learning. [173] study an active learning algorithm related to posterior matching that learns decision boundaries in discretized spaces, but does not directly maximize information about hyperplane parameters in a continuous space as we do here. More generally, to our knowledge existing work has not framed the task of active learning as a feedback communications system for the purpose of identifying an equivalent capacity-achieving distribution and selecting examples whose channel input distribution most closely approximates it, as we do here.

Logistic regression is a popular setting for the study of active learning, and has served as a testbed for the evaluation of competing example selection techniques. [174] surveyed modern active learning methods for logistic regression and evaluated them on many datasets. They generally found that uncertainty sampling and random sampling match or exceed the performance of more sophisticated (and computationally intensive) example selection methods. Uncertainty sampling, where examples closest to the estimated decision boundary are selected for labeling, is arguably the most popular active learning method for linear classification [175]. Other active learning methods for linear classifiers are discussed in the literature related to learning halfspaces under bounded noise [176].

7.2 Active Learning as a Communications Model

Let $\mathcal{U} \subseteq \mathbb{R}^d$ denote a pool of unlabeled examples from which at each training iteration $n \in \mathbb{N}$ an example $x_n \in \mathcal{U}$ is selected for labeling by an expert, who assigns label $Y_n \in \{1, 2, \dots K\}$ according to a probabilistic model. We consider a Bayesian framework in which we assume the existence of ground truth model parameters $\theta \in \Theta$ distributed according to a prior p_θ that parameterizes a distribution $p(Y \mid x, \theta)$ governing the expert's labeling behavior. As is common in active learning, we assume that the labels $\{Y_n\}$ are independent when conditioned on θ. At each iteration n, a learning algorithm A is trained on a labeled dataset $\mathcal{L}_n = \{(x_i, y_i)\}_{i=1}^n$ (using lowercase to denote previously observed labels), resulting in a trained model with parameters $\widehat{\theta}_n \in \Theta$. The task of active learning is

to design a policy π_n that, at each iteration, uses the label history $\mathcal{L}_{n-1}$ to select example x_n from the remaining unlabeled examples $\mathcal{U}_n \coloneqq \mathcal{U} \setminus \{x_i\}_{i=1}^{n-1}$, such that the classifier trains a generalizable model with as few labeled examples as possible.

In active logistic regression, θ encodes the weights of a linear separator, with $\Theta = \mathbb{R}^d$ (we consider only homogeneous logistic regression in this work). We assume a Gaussian prior $p_\theta \sim \mathcal{N}(0, \frac{1}{\lambda} I)$ with hyperparameter $\lambda > 0$. The label $Y \in \{-1, 1\}$ for data example x is assumed to be distributed according to

$$p(Y = 1 \mid x, \theta) = \frac{1}{1 + e^{-x^T \theta}}. \tag{7.1}$$

Given a labeled dataset $\mathcal{L}$, we consider a maximum a posteriori (MAP) learning algorithm given by the convex program

$$\begin{aligned} \mathrm{A}(\mathcal{L}) &= \operatorname*{arg\,max}_{\theta \in \mathbb{R}^d} \ \ln p_\theta \prod_{(x,y) \in \mathcal{L}} p(y \mid x, \theta) \\ &= \operatorname*{arg\,min}_{\theta \in \mathbb{R}^d} \ \frac{\lambda}{2} \|\theta\|_2^2 + \sum_{(x,y) \in \mathcal{L}} \ln(1 + e^{-y x^T \theta}). \end{aligned} \tag{7.2}$$

Our key insight in this work is to define an intermediate variable $L = h_\theta(x)$, where $h_\theta(x) \coloneqq x^T \theta$, and decompose the labeling distribution in eq. (7.1) into $p(Y = 1 \mid x, \theta) = p(Y = 1 \mid L) = \frac{1}{1+e^{-L}}$. This decomposition of the labeling distribution into a deterministic function $h_\theta(x)$ and conditional distribution $p(Y \mid L)$ can be found in many machine learning models. For instance, in Bayesian neural networks [48], $h_\theta(x)$ is typically given by the composition of several nonlinear layers with $L = h_\theta(x)$ encoding the final layer feature vector, and $p(Y \mid L)$ is given by the softmax function. Figure 7.1a depicts this decomposition for logistic regression, and Figure 7.1b illustrates the full active learning decomposition in the general case.

By decomposing active learning in this manner, we are able to draw direct connections to feedback channel coding, in which a message θ is encoded into a sequence of symbols $\{L_n\}$,

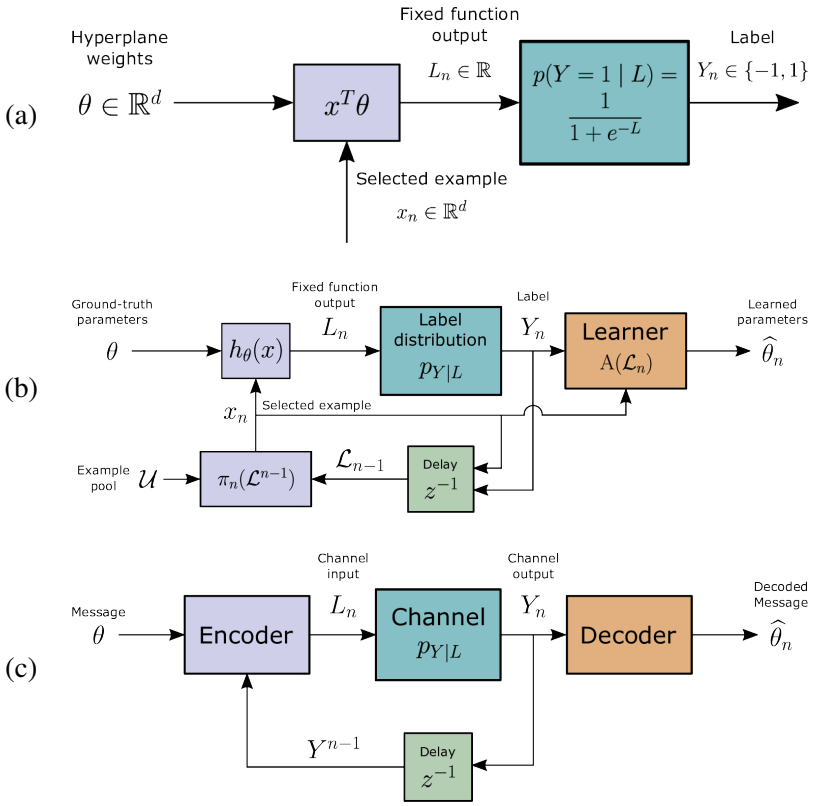


Figure 7.1: (a) Decomposition of logistic regression into an inner product between hyperplane θ and example x_n, and a logistic label distribution that depends only on this product. (b) Active learning decomposed into a deterministic function h, label distribution $p(Y \mid L)$, and feedback of the labeling history $\mathcal{L}_{n-1}$ to example selection policy π_n. (c) Coding with feedback, where a message is transmitted across a noisy channel as a sequence of symbols and subsequently decoded (copied from Figure 2.2). By comparing (b) and (c), one can draw direct connections between active learning and coding with feedback.

transmitted across a channel with transition probability $p(Y \mid L)$ yielding noisy output symbols $\{Y_n\}$, and subsequently decoded into an estimated message $\widehat{\theta}_n$. The availability of noiseless feedback from the channel output to the encoder provides the encoder with the history of received symbols, and allows it to adaptively select an informative channel input (Figure 7.1c). By comparing Figures 7.1b and 7.1c, we can see the direct correspondence between active learning and channel coding with feedback: model parameters θ serve as the message, which is encoded by function h (parameterized by x_n) into channel input $L_n = h_\theta(x_n)$. Label distribution $p(Y \mid L)$ can be interpreted as a noisy channel, with label Y_n as the channel output. Algorithm A decodes labeled data $\mathcal{L}_n$ into a decoded message $\widehat{\theta}_n$,

and $\mathcal{L}_n$ is passed as noiseless feedback to the encoder. This formulation of active learning as a feedback communications system allows one to leverage existing tools in channel coding for the design of an example selection scheme π_n. While similar decompositions have been observed in prior work [170, 19], we believe our work is the first to use this approach to analyze active learning in a real-world setting such as logistic regression.

7.2.1 Optimal Feedback Coding

In devising a feedback coding scheme for selecting a sequence of channel inputs $\{L_n\}$, there are several quantities that characterize optimal performance. As a reminder, we denote the mutual information $I(L;Y)$ between random variables L and Y as a function of marginal distribution p_L and conditional distribution $p_{Y|L}$ (using the notation p_L and $p_{Y|L}$ interchangeably with $p(L)$ and $p(Y \mid L)$) given by $I(p_L, p_{Y|L})$:

$$I(p_L, p_{Y|L}) \coloneqq \int_{L,Y} p_L \, p_{Y|L} \log_2 \frac{p_{Y|L}}{p_Y},$$

where p_Y denotes the output distribution of channel $p_{Y|L}$ with input distribution p_L. Letting $y^i \coloneqq \{y_1, \dots y_i\}$ denote the history of observed channel outputs, at iteration n we seek to maximize the information gain $I(\theta; Y_n \mid y^{n-1})$, which measures the one-step decrease in uncertainty about the message upon receiving each channel output. For deterministic encoders, information gain is equal to $I(L_n; Y_n \mid y^{n-1}) = I(p_{L_n|y^{n-1}}, p_{Y|L})$ [27]. Note that for a fixed channel $p_{Y|L}$, information gain is only a function of the channel input distribution $p_{L_n|y^{n-1}}$, conditioned on the history of channel outputs.

As discussed in Chapter 2, a key quantity in channel coding is the channel capacity C, defined as the maximum mutual information across the channel for any channel input distribution p_L within some class $\mathcal{C}$:

$$p_L^*(\mathcal{C}) \coloneqq \arg\max_{p_L \in \mathcal{C}} I(p_L, p_{Y|L}) \quad C \coloneqq I(p_L^*, p_{Y|L}).$$

The capacity-achieving distribution $p_L^*(\mathcal{C})$ is the input distribution in $\mathcal{C}$ that maximizes information across the channel. Through achievability and converse arguments, a central result in information theory is that optimal coding schemes, when marginalized over the message set, should induce the capacity-achieving distribution on the channel input [15]. In working towards applying existing feedback coding schemes to active example selection, we first characterize the capacity-achieving distribution for logistic regression, which is a core contribution of our work and forms the basis of our novel active logistic regression scheme in Section 7.3.

Channel Capacity in Logistic Regression. Letting $f(\ell) \coloneqq \frac{1}{1+e^{-\ell}}$, we observe from Figure 7.1a that logistic regression has a binary output channel with transition probability $p(Y = 1 \mid L) = f(L)$. Without constraints on the channel input, the information gain can be maximized by placing masses of equal weight at $\pm\infty$. However, logistic regression imposes the structural constraint $L = x^T\theta$, so that such a distribution would require data points of infinite energy for finite model weights. Therefore, to characterize logistic regression capacity in practice, we consider the capacity-achieving distribution within the class of power-constrained distributions given by $\mathcal{C}_P \coloneqq \{p_L : \mathbb{E}[L^2] \leq P\}$; we discuss the selection of P in Section 7.3. With this class defined, we have our first result.

Proposition 7.2.1 (Capacity of Logistic Regression)**.** *For $p(Y = 1 \mid L) = f(L)$, we have $p_L^*(\mathcal{C}_P) = B_{\sqrt{P}}$, where B_t is defined as $B_t(\ell) \coloneqq \frac{1}{2}\delta(\ell - t) + \frac{1}{2}\delta(\ell + t)$ and δ denotes the Dirac delta function. Furthermore, we have $C = I(B_{\sqrt{P}}, f) = 1 - h_b(f(\sqrt{P}))$, where h_b denotes the binary entropy function.*

The proof follows closely to that of [177] for the one-bit quantized Gaussian channel; the proofs of Proposition 7.2.1 and all subsequent results are presented in Section D.1.

7.2.2 Posterior Matching

By characterizing the channel capacity and capacity-achieving distribution of active learning models, we enable the use of existing feedback coding schemes that achieve capacity. Recently, a multidimensional extension of posterior matching has been developed to select a sequence of channel inputs $\{L_n\}$ to maximize the information gain across a given channel $p_{Y|L}$. The central concept is to construct an encoder that by definition induces $p_{L_n|y^{n-1}} = p_L^*$ for every n, which in essence hands the decoder the information that it is still "missing" [31]. This involves the construction of an encoder mapping $S_{y^{n-1}} : \theta \to L$ parameterized by y^{n-1} such that $S_{y^{n-1}}(\theta) \sim p_L^*$ for every n.

While posterior matching is an attractive feedback coding scheme, there are challenges in applying it to active learning: given the structural constraints of any particular active learning problem as depicted in Figure 7.1b, it may not always be the case that a mapping from $p_{\theta|\mathcal{L}_{n-1}}$ to p_L^* exists, since the encoder is constrained to the set of mappings given by $\{h_\theta(x) : x \in \mathcal{U}_n\}$.[2] For example, in active logistic regression under mild assumptions, there exists no x such that $h_\theta(x) \sim p_L^*$, as shown in the following proposition.

Proposition 7.2.2. *Under a log-concave prior distribution p_θ, in Bayesian logistic regression for any n there exists no x_n that induces $p_{L_n|\mathcal{L}_{n-1}} \sim p_L^*$.*

Since we assume a Gaussian prior p_θ (which is log-concave), Proposition 7.2.2 applies and therefore there exists no active logistic regression scheme π_n corresponding to a posterior matching mapping from θ to p_L^*. We suspect that the infeasibility of p_L^* holds generally in other real-world machine learning models (e.g., Bayesian neural networks) due to similar structural constraints imposed by $h_\theta(x)$, preventing the direct application of posterior matching for example selection. In the next section, we extend concepts from posterior matching to a novel active learning scheme compatible with this constrained encoder structure.

[2] The analogous distribution to $p_{L_n|y^{n-1}}$ in active learning is $p_{L_n|\mathcal{L}_{n-1}}$. When considering only deterministic example selection schemes, $p_{L_n|\mathcal{L}_{n-1}}$ is induced directly from $p_{\theta|\mathcal{L}_{n-1}}$, through $h_\theta(x)$.

7.2.3 Approximate Posterior Matching

To address the impossibility of finding $x \in \mathcal{U}$ that induces p_L^* on L, we introduce a scheme that instead selects an example x_n such that $p_{L|\mathcal{L}_{n-1}}$ is distributed "as close as possible" to p_L^*, as measured by a distance between distributions. Specifically, we use the 2-Wasserstein distance because of its convenient geometric properties and compatibility with non-overlapping distribution supports [178]. The p-Wasserstein distance between distributions μ and ν is given by

$$W_p(\mu, \nu) = \left(\inf_{\gamma \in \Pi(\mu,\nu)} \int_u \int_v |u - v|^p \gamma(u, v) \right)^{\frac{1}{p}},$$

where $\Pi(\mu, \nu)$ is the set of couplings with marginal distributions μ and ν [179]. Our selection scheme, called *Approximate Posterior Matching* (APM), is then given by

$$x_n = \pi_n(\mathcal{L}_{n-1}) \coloneqq \operatorname*{arg\,min}_{x \in \mathcal{U}_n} \; W_2(p_{L_n|\mathcal{L}_{n-1}}, p_L^*). \tag{7.3}$$

While APM is intuitively appealing because it steers the induced channel distribution as close as possible to p_L^*, we justify this strategy in the next section for the case of logistic regression by showing that information gain does in fact approach its maximum possible value as $W_2(p_{L|\mathcal{L}_{n-1}}, p_L^*)$ is minimized.

7.3 APM in Logistic Regression

Under the power constraint $\mathbb{E}[L^2] \leq P$, Proposition 7.2.1 establishes that the capacity-achieving distribution in the logistic regression system is given by $B_{\sqrt{P}}$. We now show an information continuity result for this capacity-achieving distribution, which provides a mathematical justification for the APM Wasserstein distance minimization in eq. (7.3).

Theorem 7.3.1. *Let $\widetilde{C}_n = \max_{x \in \mathcal{U}_n} I(p_{L_n|\mathcal{L}_{n-1}}, f)$ denote the maximum information gain from* any *example selected at iteration n, and suppose $P > 0$ is selected such that $p_{L_n|\mathcal{L}_{n-1}} \in$*

$\mathcal{C}_P$ *for any* $x \in \mathcal{U}_n$*. Then for any* $x \in \mathcal{U}_n$*,*

$$\widetilde{C}_n - I(p_{L_n|\mathcal{L}_{n-1}}, f) \leq K_P W_2(p_{L_n|\mathcal{L}_{n-1}}, B_{\sqrt{P}}),$$

where $K_P > 0$ *is a constant that only depends on* P*.*

For decreasing $W_2(p_L, B_{\sqrt{P}})$, this result bounds $I(p_{L_n|\mathcal{L}_{n-1}}, f)$ towards its maximum possible information gain $\widetilde{C}_n$. In other words, minimizing the distance to the *known* capacity-achieving distribution (even if not achievable in practice) ensures that the information gain approaches its maximum value within the set of possible input distributions — a value which is *unknown* a priori. As we shall see in the results and experiments that follow, targeting the known capacity-achieving distribution affords geometric simplifications and computational benefits over the strategy of directly selecting the example that achieves $\widetilde{C}_n$. Unlike APM, the latter method does not benefit from analytical knowledge of the information structure of the channel and constraint set, and so it must instead conduct an expensive brute-force maximization of information gain.

7.3.1 Closed-form Results

For logistic regression, the calculation of $W_2(p_L, B_t)$ takes a convenient closed-form expression, which simplifies the example selection in eq. (7.3):

Proposition 7.3.2. *For* $t > 0$*, with* $\mathrm{med}_{p_L}(L)$ *denoting the median of* L *according to distribution* p_L*,*

$$W_2^2(p_L, B_t) = \mathbb{E}_{p_L}[L^2] - 2t\,\mathbb{E}_{p_L}[|L - \mathrm{med}_{p_L}(L)|] + t^2.$$

We can simplify this expression even further when p_L is normally distributed:

Corollary 7.3.1. *For $L \sim \mathcal{N}(\mu, \sigma^2)$,*

$$W_2^2(p_L, B_t) = \mu^2 + \left(\sigma - \sqrt{\frac{2}{\pi}}t\right)^2 + \left(1 - \frac{2}{\pi}\right)t^2.$$

At iteration n, suppose that $p_{\theta|\mathcal{L}_{n-1}}$ is approximated by $\mathcal{N}(\mu_n, \Sigma_n)$, resulting in channel input $L_n = \theta^T x_n$ being distributed as $\mathcal{N}(\mu_n^T x_n, x_n^T \Sigma_n x_n)$. Although $p_{\theta|\mathcal{L}_{n-1}}$ is not normally distributed in logistic regression, it is common to make this approximation in practice [180]. By applying Corollary 7.3.1 and omitting constant terms, we derive our APM selection policy for logistic regression with power constraint P.

Definition 7.3.1. Approximate Posterior Matching for Logistic Regression (APM-LR):

$$\pi_n(\mathcal{L}_{n-1}) = \underset{x \in \mathcal{U}_n}{\arg\min} \ (\mu_n^T x)^2 + \left(\sqrt{x^T \Sigma_n x} - \sqrt{\frac{2}{\pi}P}\right)^2. \tag{7.4}$$

This objective is a combination of two terms: the first term corresponds to minimizing the distance between example x and the posterior mean hyperplane. If μ_n is taken as an estimate of θ, this term corresponds to the well-known *uncertainty sampling* active learning method, which samples points close to the current hyperplane estimate [175]. The second term prefers examples that align with the direction of maximum posterior covariance. Specifically, for $x^T \Sigma_n x < \frac{2}{\pi}P$, the second term is a decreasing function of $x^T \Sigma_n x$, encouraging x to align with posterior covariance eigenvectors with large eigenvalues.

These two terms together can be interpreted as encouraging "exploitation" and "exploration," respectively: the first term encourages the selection of examples that are close to the current estimate of θ, exploiting this estimate to only query examples whose labels are ambiguous. The second term balances this exploitation by probing in directions of the hyperplane posterior that have not yet been sufficiently explored, reducing uncertainty about the hyperplane itself. Figure 7.2 visualizes this tradeoff in comparison to uncertainty sampling, which only queries examples close to the current hyperplane estimate and does not account for the fact that there may be directions of the hyperplane posterior that have

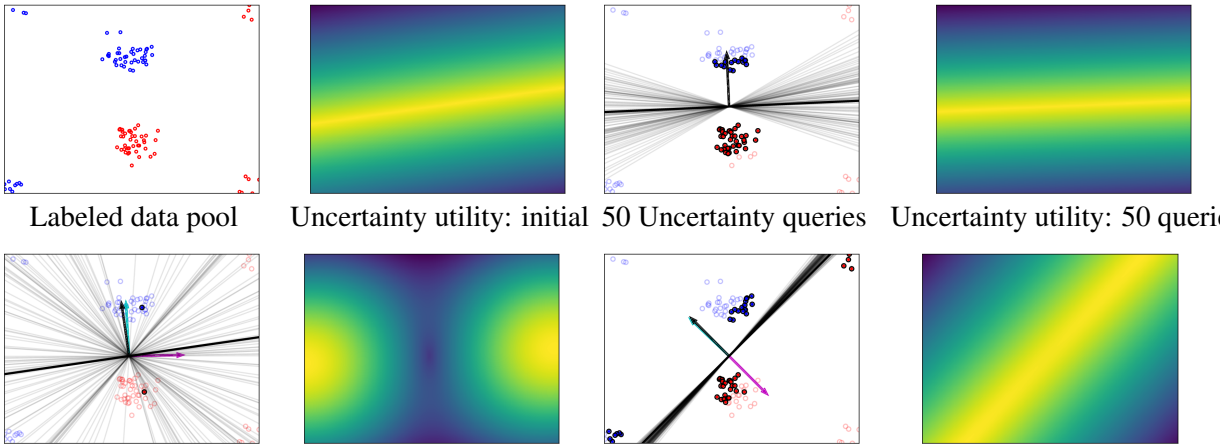


Figure 7.2: (a) Top: linearly separable dataset (optimal hyperplane is diagonal) demonstrating the failure of uncertainty sampling (dataset adapted from [185]). Bottom: samples from hyperplane posterior, given two seed labels. The black hyperplane and corresponding normal vector depict the initial logistic regression solution, the cyan arrow indicates the normal vector to the posterior mean hyperplane, and the purple arrow indicates the maximal eigenvector of the posterior. (b) Utility function heatmap for uncertainty sampling (top) and APM-LR (bottom) — the unlabeled example with the highest utility is selected for labeling. Uncertainty sampling selects examples close to the current hyperplane, while APM-LR selects examples that are both close to the posterior mean hyperplane and align with the direction of largest posterior variance. (c-d) After 50 queries, uncertainty sampling (c-top) has not selected samples in the dataset corners, leading to sampling bias and continued sampling of the center clusters (d-top). Meanwhile, APM-LR (c-bottom) has sufficiently explored the dataset, while continuing to sample examples in only the most ambiguous regions (d-bottom).

not been sufficiently explored. This myopic behavior is an instance of *sampling bias*, a well-known phenomenon in active learning where a policy continually selects examples that reinforce the learner's belief in an incorrect hypothesis [181, 182, 183]. The balance of exploitation and exploration terms in APM-LR helps prevent this type of sampling bias, in a spirit similar to other active learning methods that balance uncertainty reduction with diverse example selection [184, 185].

An attractive computational feature of eq. (7.4) is that the posterior mean and covariance can be estimated *once* at each selection iteration and then simply projected onto each candidate example, resulting in a computational cost of only $O(d^2)$ per example evaluation.

Note that these computational advantages along with the natural balance between exploration and exploitation in APM-LR emerged naturally from first-principles of feedback coding, demonstrating the potential of identifying the capacity-achieving distribution and applying APM as a universal means of designing geometrically intuitive, computationally efficient active selection schemes.

7.4 Experimental Results

We evaluate the performance of APM-LR against baseline example selection methods for logistic regression on a variety of datasets from different tasks, as measured by holdout test accuracy and selection compute time. For each method, we follow [174] and set the regularization parameter in eq. (7.2) to $\lambda = 0.01$, which we solve with the LIBLINEAR solver [186]. After each example is labeled, we approximate $p_{\theta|\mathcal{L}_{n-1}}$ with a normal distribution by applying the variational approximation described in [187], which is solved in only a few iterations of an expectation-maximization procedure (referred to here as "VariationalEM"). The final component needed to apply APM-LR is the selection of power constraint P in eq. (7.4).

Selecting Power Constraint Although our approach is rooted in feedback coding theory, regarding the power constraint there are two key differences between our model and traditional communications systems. First, unlike telecommunications systems that have physical restrictions such as limited battery levels, in our framework there is no external prescription of the power budget P and therefore we can select any valid upper bound on the channel input power induced by the unlabeled examples. Secondly, unlike coding schemes which globally maximize information gain over the entire trajectory of channel inputs, we seek to myopically maximize the one-step information gain at every channel input. Since the goal at each iteration is to separately solve a local information maximization problem, there is no need for the power constraint P to be constant across iterations, and therefore we

Algorithm 5: Approximate Posterior Matching for Logistic Regression (APM-LR)

Input: data pool $\mathcal{X}$, hyperparameter $\lambda > 0$, horizon N, initial training set $\mathcal{L}$

$\mu \leftarrow 0, \Sigma \leftarrow \frac{1}{\lambda} I$
$B \leftarrow \max_{x \in \mathcal{U}} \|x\|_2$
$\mathcal{U} \leftarrow \mathcal{X}$
for $n = 1$ **to** N **do**
  $P \leftarrow B^2 \lambda_1(\Sigma)$
  $x^* \leftarrow \arg\min_{x \in \mathcal{U}} (\mu^T x)^2 + \left(\sqrt{x^T \Sigma x} - \sqrt{\frac{2}{\pi} P} \right)^2$
  $y^* \leftarrow \text{ExpertLabel}(x^*)$
  $\mathcal{U} \leftarrow \mathcal{U} \setminus \{x^*\}, \mathcal{L} \leftarrow \mathcal{L} \cup (x^*, y^*)$
  $\mu, \Sigma \leftarrow \text{VariationalEM}(\mathcal{L})$
  $\theta^* \leftarrow \text{A}(\mathcal{L})$ i.e., eq. (7.2)
end for

Output: hyperplane θ^*

set a separate power constraint P_n for each iteration.

Since the selection of P_n parameterizes the target distribution in APM-LR, it is important for P_n to be set as tight as possible so that the target capacity-achieving distribution is well-matched to the set of feasible channel input distributions. This is because at each iteration the capacity-achieving distribution serves as a proxy for the optimal input distribution induced by a real example, and a setting of P_n that is too loose will result in APM targeting a proxy that is not well-matched to the feasible input distributions. To select a satisfactory setting of P_n, we derive an upper bound on the channel input power to use as an implicit constraint.

Suppose for a given dataset that there exists a known $B > 0$ such that $\|x\|_2 < B$ (this is a reasonable assumption in many real-world settings). Let $\lambda_1(M)$ denote the largest magnitude eigenvalue of matrix M. We then have (with expectations taken with respect to $p_{L_n|\mathcal{L}_{n-1}}$)

$$\mathbb{E}[L_n^2] = x^T(\mu_n \mu_n^T + \Sigma_n)x \leq B^2 \lambda_1(\mu_n \mu_n^T + \Sigma_n).$$

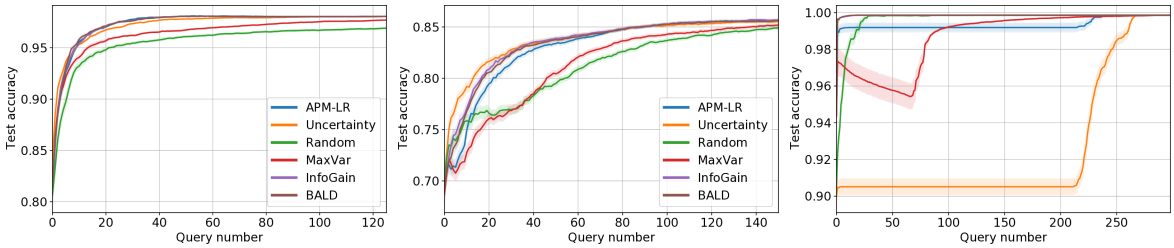
For each n we can therefore set $P_n = B^2 \lambda_1(\mu_n \mu_n^T + \Sigma_n)$. In our experiments we select a slightly modified parameter $P_n = B^2 \lambda_1(\Sigma_n)$, which we justify as a more practical heuristic in Section D.2.1. We summarize APM-LR in full in Algorithm 5, including power constraint calculation and variational posterior updating.

Datasets We follow previous work in active learning for logistic regression [185, 174] and test each method on several UCI datasets [188] including *vehicle*, *letter*, *austra*, and *wdbc*. We also evaluate performance on several synthetic datasets including the dataset depicted in Figure 7.2 (adapted from [185]), which we refer to as *cross* (see Section D.2.2 for details on all datasets). For each simulation trial, we first randomly divide the dataset into an equally-sized data pool ($\mathcal{U}$) and held-out test set. We normalize $\mathcal{U}$ to zero-mean and coordinate-wise unit-variance, and apply the same transformation to the test set. Before evaluating each example selection method, the training dataset ($\mathcal{L}$) is seeded to consist of one randomly selected labeled example from each class.[3]

Baseline Methods We evaluate the following baseline methods, each described with their computational cost per candidate example evaluation (see Section D.2.3 for details):

- *Uncertainty*: select closest example to current hyperplane estimate (i.e. $\arg\min_{x \in \mathcal{U}_n} x^T \widehat{\theta}_{n-1}$) at cost $O(d)$. The action of Uncertainty sampling is comparable to that of the first term in eq. (7.4).
- *Random*: each example is selected uniformly at random from $\mathcal{U}_n$, at $O(1)$ cost.
- *MaxVar*: to isolate the effect of the second term in eq. (7.4), we evaluate a control strategy that selects the example that induces the largest channel input variance (i.e. $\arg\max_{x \in \mathcal{U}_n} x^T \Sigma_n x$), at cost $O(d^2)$.
- *InfoGain*: selects the example with the largest information gain $I(\theta; Y_n \mid \mathcal{L}_{n-1})$, estimated by sampling s times from the normally approximated hyperplane posterior (here we set $s = 100$) and for each candidate example evaluating a Monte Carlo approximation of information gain, at $O(ds)$ cost.
- *BALD*: we approximate the logistic function $f(\ell)$ with a probit function and apply the probit regression active learning method of [171], at cost $O(d^2)$. Like APM-LR, BALD

[3] Our experiments are synchronized across data selection methods: each trial uses the same training/test split and seed examples for each tested method.



(a) *letterDP* (b) *austra* (c) *cross*

Figure 7.3: Average test classification accuracy plotted against number of labeled examples (error bars show ± 1 standard error) across select UCI datasets (a-b) and the synthetic *cross* dataset (c, with legend shared with a-b and omitted for visual clarity). Overall, APM-LR performs comparably to other methods seeking to approximately maximize information gain. While uncertainty sampling performs well on some datasets (a-b), it can fail in cases where it suffers from sampling bias (c). Most of the tested active learning methods (except the control, MaxVar) outperform random sampling. For visual clarity we show different numbers of queried examples for each dataset.

approximates the action of InfoGain and only requires the mean and covariance of the normally approximated hyperplane posterior.

InfoGain is the most computationally intensive selection method, since it requires a brute-force Monte Carlo approximation of information gain for each candidate example. BALD and APM-LR have the next least expensive cost per example at $O(d^2)$, followed by Uncertainty and Random sampling.

Performance Comparison In Figure 7.3, we compare the learning performance of each data selection method by plotting holdout test accuracy against number of queried examples (excluding the seed set) across select datasets (see Section D.2.4 for full results). We generally find that the tested active data selection methods outperform random sampling. The exception is MaxVar, which performs comparably to random selection and worse than APM-LR. Although simple Uncertainty sampling matches the performance of other active methods on several datasets (Figure 7.3a-b) as previously observed by [174], in additional tests on synthetic datasets we find that APM-LR outperforms uncertainty sampling. This is the case for the *cross* dataset (Figure 7.3c), demonstrating how Uncertainty sampling can be susceptible to sampling bias that leads to insufficient exploration (see Section D.2.6

Table 7.1: Comparison of median cumulative time (s) for each method to select the first 40 examples (excluding seed points and time for model retraining). Generally, APM-LR has a cost an order of magnitude lower than InfoGain and BALD (which directly approximate the action of information maximization), while Uncertainty, MaxVar, and Random sampling have the cheapest cost.

	letterDP	*austra*	*cross*
APM-LR	0.336	0.150	0.125
Uncertainty	0.149	0.063	0.053
BALD	4.230	1.770	1.521
InfoGain	12.755	5.089	2.722
Random	0.005	0.003	0.002
MaxVar	0.118	0.050	0.040

for additional failure mode analysis). These tests together lend evidence to the mixture of terms in eq. (7.4) having combined benefits over pure exploration of directions with large posterior variance or pure exploitation of ambiguous examples with respect to the current hyperplane estimate. Finally, APM-LR generally performs similarly to InfoGain and BALD, both of which directly approximate the action of information gain maximization, in contrast to APM's geometric, indirect approach.

Table 7.1 depicts the computational cost for each method across select datasets (see Section D.2.5 for full results and expanded timing evaluations). Similar to the analysis in [174], for each method we evaluate the cumulative compute time to select the first 40 examples (excluding seed examples and time for model retraining), and compute the median time over all trials. We see that InfoGain is the most expensive of all methods, since it directly approximates information gain with Monte Carlo sampling. BALD has the next highest cost, followed by APM-LR — the two latter methods only require a single computation of posterior mean and variance, which can be projected onto each candidate example. Uncertainty sampling and random sampling have the lowest computational cost.

Although BALD can also be computed using only the posterior mean and covariance, it is unclear how the approximation in BALD can be applied beyond probit regression. In contrast, the APM formulation in eq. (7.3) can be applied generally to *any* active learning problem that can be decomposed into a deterministic encoder and noisy channel, along with

a known capacity-achieving distribution. The combined results of Figure 7.3 and Table 7.1 suggest that the universal APM approach of leveraging this analytical knowledge of the capacity-achieving distribution affords a geometric active selection approach that performs well in terms of both sample and computational complexity.

7.5 Discussion

To our knowledge, our work is the first effort to both reframe active learning as a feedback communications system and utilize analytical knowledge of the corresponding capacity-achieving distribution to derive an active learning scheme. The analytical and empirical results in this work for the special case of logistic regression demonstrate the potential of this coding-based active learning approach: information continuity results show how examples selected with APM-LR have information gain approaching their maximum possible value, APM-LR has a convenient geometrical formulation resulting from analytical knowledge of the capacity-achieving distribution for logistic regression (characterized here for the first time) that can lead to computationally efficient example selection, and when tested on multiple datasets APM-LR performs comparably to baseline active learning methods including brute-force information maximization. APM-LR's attractive balance between exploration and exploitation emerged naturally from first-principles of channel coding, extending beyond the common approach of uncertainty sampling.

More generally, a fundamental feature of Approximate Posterior Matching is that analytical knowledge of the capacity-achieving distribution converts the usually unwieldy *information maximization* problem in active learning to a *geometric* problem. In logistic regression, this geometry led to computational advantages over direct information maximization, and we conjecture that similar benefits may emerge in more complex settings. Additionally, as we discuss further in Chapter 8 the general formulation of APM in eq. (7.3) presents several opportunities to leverage existing computational algorithms to aid example selection, including estimating p_L^* when it is analytically unknown [189, 190] and optimiz-

ing Wasserstein distances with state-of-the-art methods [191]. Overall, we believe that our coding-theoretic approach opens several new directions for future work in active learning.

CHAPTER 8
CONCLUSIONS AND FUTURE WORK

As we demonstrate in this thesis, interactive machine learning as a field of study shares many similarities and areas of overlap with feedback communications: the expert's knowledge can be abstracted as a message to be communicated iteratively to a learner through a sequence of intelligently selected interactions that encode this knowledge. When IML is directly framed as a feedback coding problem using the language and tools of information theory, it becomes clear which system structures and mathematical quantities are important for designing and measuring the performance of efficient (both in terms of computational and sample complexity) interaction selection strategies. In this thesis, we make strides in this direction by utilizing tools and concepts from feedback coding theory to identify and model query structures in IML and develop direct information maximization solutions as well as approximations that allow for computationally efficient query selection. The contributions of this thesis progress in three parts from directly deploying a feedback coding scheme for interaction design in HCI systems to formulating active learning in a feedback coding paradigm for the design of a general example selection strategy through first principles of coding theory.

- Chapters 3 and 4 explore the problem of information maximization in human-computer interaction as a step towards studying information maximization in general IML problems. We consider a broad class of HCI systems where the expert interacts with an effector through noisy one-bit interactions. We adopt one-dimensional posterior matching as an interaction algorithm by mathematically modeling the human expert as a comparator whose output is passed through a binary symmetric channel. In practice, posterior matching in this context distills to an efficient, robust, and easy-to-use interaction policy where at each iteration the human issues refinements to

the effector by comparing their desired effector behavior to a guess that bisects the set of remaining candidate behaviors. To accomplish this, in each setting we create a dictionary of effector behaviors with a human-interpretable ordering rule for sorting any two behaviors in the dictionary. We demonstrate the success of this approach on both robot swarm control using a brain-computer interface and segmenting objects in images. These two chapters demonstrate the potential for *direct* solutions to information maximization by explicitly deploying an existing feedback coding scheme.

- Chapters 5 and 6 explore how query structure can be leveraged for computationally efficient approximations to information maximization in interactive similarity and preference learning. Unlike the comparator model in Chapters 3 and 4 and its natural connection to posterior matching, the query structures in these tasks do not lend themselves to clear application of an existing feedback coding scheme for example selection. However, the query structures in these problems along with tools in information theory can still be utilized to devise computationally efficient information maximization strategies. In similarity learning, a set of basic assumptions on the query model allows us to devise a Monte Carlo sampling scheme to approximate the information gain of any candidate ranking query. In preference learning, we leverage the query geometry of paired comparisons with a logistic noise model to devise two simple, geometrically appealing strategies that approximate the action of information gain maximization while having a computationally cheaper acquisition function. We show in both similarity and preference learning how taking advantage of query structure can lead to simplifying approximations that outperform (both in terms of computational and sample complexity) baseline methods.

- Chapter 7 addresses the challenge encountered in Chapters 5 and 6 of being able to identify useful query structures to design interaction policies, but lacking a solu-

tion that directly leverages existing algorithms in feedback coding as in Chapters 3 and 4. In this chapter, we introduce a general coding-theoretic solution to example selection by framing active learning as a communications system and identifying the corresponding encoder, channel, decoder, and feedback components. Specifically, we identify a fixed, deterministic function and conditional label distribution as the critical components involved in translating active learning problems to coding problems. By identifying these components, we can then characterize the equivalent capacity-achieving distribution in active learning, which we present for the first time for logistic regression. With capacity characterized, we discuss the challenges in applying multidimensional posterior matching for example selection, and instead propose the Approximate Posterior Matching selection algorithm which mimics the action of posterior matching for use with the type of encoder constraints typically encountered in machine learning. We then explore APM in the case of logistic regression, and show how the resulting acquisition function elegantly balances between data space exploration and uncertainty exploitation, while taking a convenient computational form that is cheaper to evaluate than existing information maximization methods while performing comparably on various datasets. This chapter serves as a culmination of the work in this thesis by formulating general active learning as a coding problem and proposing a universal coding-theoretic approach to the design of active learning policies, even in the face of structural encoder constraints.

While information gain maximization in interactive learning has a long history and is a popular approach used by many methods, the work in this thesis shows how existing models and tools in information theory can be applied to derive general interaction policies from first principles of coding theory by identifying and capitalizing on query structures found in each problem. Identifying such structures and applying existing or modifications of existing feedback coding schemes is an exciting and fruitful endeavor, since coding and information theory are mature fields that offer many untapped avenues for future work at the intersection

with interactive machine learning. As we demonstrate in this thesis, directly exploiting this intersection for the design of IML query selection strategies is a powerful and rewarding approach that offers new perspectives on IML theory and algorithm development.

Using the coding-theoretic frameworks presented in this thesis as a starting point, there are several immediate areas of future work that could potentially offer new insights into IML. Although the work in this thesis is focused on channel coding, it would be interesting to apply tools from joint source-channel coding [192], which is a more general and possibly more appropriate lens through which to study IML. Since the seminal work of posterior matching in [22], there have been advancements in the study of feedback coding in additional settings such as coding with noisy feedback [193] and coding over channels with memory [194]; such extensions may inspire query selection strategies in IML problems involving similar structures. In IML problems with one-bit query structures such as pairwise search and logistic regression, there are connections to be explored in the information theory literature on one-bit quantization with variable thresholds, where one-bit information maximization has been studied in depth [177, 195, 196]. As an outlook on possible extensions to the coding-theoretic concepts and algorithms for IML presented in this thesis, we conclude with a brief discussion of specific ongoing and future work.

Information-theoretic Sequential Machine Teaching Although the problems explored in this thesis can all be translated to an active learning formulation — where the learner has full agency in selecting queries — it would be interesting to explore the application of feedback information and coding-theoretic tools to sequential machine teaching, where the expert (or *teacher*) has agency in selecting examples for labeling. Observing Figure 7.1b, it is straightforward to modify this active learning coding framework to accommodate the machine teaching problem by making the ground truth parameters available to the example selection policy π_n, as diagrammed in Figure 8.1. This formulation is in fact a generalization of the framework in Figure 7.1b, since any machine teaching selection policy that does not

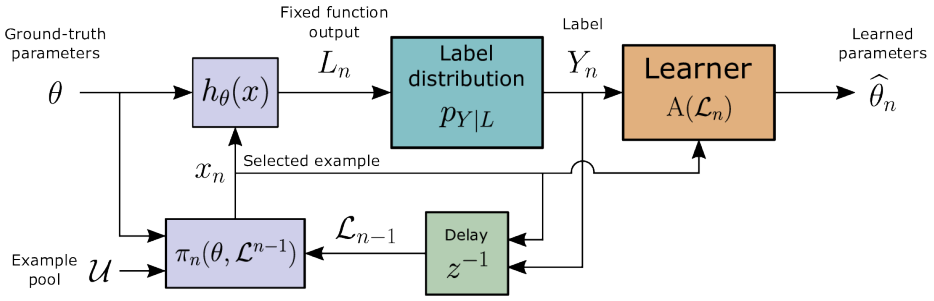


Figure 8.1: Sequential machine teaching as a communications system with feedback

utilize θ simply specializes to the active learning framework.

While the differences between Figure 7.1b and Figure 8.1 are subtle, in the case of machine teaching there are two information pathways between the ground truth parameters and the learner: information is encoded through not only the labels of examples, but also through the example features themselves. For instance, in the one-dimensional machine teaching example from Figure 1.1c, the locations of two oppositely labeled data points directly encode the value of the ground truth threshold to the learner, since these points were selected by the expert to be bisected by the threshold. This intuition of collective information encoding between labels and features can be formalized by considering the full information gain (conditioned on the labeled history) that the labeled example at iteration n provides to the learner about the ground truth parameters, and by applying the chain rule of mutual information:

$$I(X_n, Y_n; \theta \mid \mathcal{L}_{n-1}) = I(X_n; \theta \mid \mathcal{L}_{n-1}) + I(Y_n; \theta \mid X_n, \mathcal{L}_{n-1}). \tag{8.1}$$

The first term on the right-hand side of eq. (8.1) is the information gain that the selected example itself provides about the ground truth parameters, and the second term is the information gain provided by that example's label. In active learning, the first term is null since knowledge of an unlabeled example cannot provide information to the learner about the ground truth parameters, since it was the learner who selected this example initially.

In essence, the two terms in eq. (8.1) correspond to *two* channels between the ground

truth parameters and the learner. This formulation presents several exciting avenues for both practical and theoretical future work at the intersection of coding theory and machine teaching. In the case of human teachers, in the spirit of Chapters 3 and 4 there is an open question of how to translate mathematical concepts in feedback coding to human decision-making in this setting. One approach is to consider human-interpretable query types where the human teacher at iteration n selects an example to label from some fixed set S_n of N_x examples, along with a teaching instruction such as *"Label the example from set S_n whose label you feel most confident about."* We can model the human's response to such a query based on their knowledge of parameters θ.

For instance, in the case of a human's classification behavior being adequately modeled by a linear classifier, notions of confidence can arguably be translated to distance relationships between examples and the underlying hyperplane parameterized by θ, since by assumption the human provides ambiguous labels near the hyperplane boundary, and more consistent labels for examples further away. Since the human's response may have an element of randomness (due to inconsistent answers or inadequate feature or model complexity), we can devise a *discrete choice model* [197] to capture the teacher's behavior, which we model as being governed by a probability distribution $p(x \mid S_n, \theta)$. Such a probability model for the teacher's behavior can be interpreted as a transition probability for a discrete memoryless example selection channel with input and output alphabets of size N_x.

Therefore, an informative machine teaching policy should populate the set S_n with examples that simultaneously maximize the information gained through the selection of the example itself (i.e., $I(X_n; \theta \mid \mathcal{L}_{n-1})$) and the information gained from the label of the example that is actually selected by the teacher (i.e., $I(Y_n; \theta \mid X_n, \mathcal{L}_{n-1})$). While the human teacher is ultimately the one who selects the example X_n for labeling, computational and theoretical tools from information and coding theory could potentially be used to choose the example set S_n by solving a simultaneous information maximization problem over two channels given by $p(x \mid S_n, \theta)$ and $p(Y \mid L)$.

In such an investigation, it would also be important to theoretically characterize the excess information gain provided by machine teaching over the most informative active learning policy in the same problem setting. Formally, we are interested in the sign of the difference

$$\sup_{\pi_n} I(X_n, Y_n; \theta \mid \mathcal{L}_{n-1}) - \sup_{\pi_n} I(Y_n; \theta \mid X_n, \mathcal{L}_{n-1}), \tag{8.2}$$

where the first term represents the maximum one-step information gain achievable under any machine teaching policy, and the second term is the maximum information gain possible under any active learning policy (which only maximizes information with respect to the label channel). Since any machine teaching policy can choose to ignore knowledge of θ during example selection (setting $I(X_n; \theta \mid \mathcal{L}_{n-1}) = 0$), the difference in eq. (8.2) is trivially greater than or equal to zero. The interesting question then is if there exist problem settings where the difference in eq. (8.2) can be shown to be strictly greater than zero, which would theoretically highlight the benefits of machine teaching over active learning.

Approximate Posterior Matching with Unknown Capacity In Chapter 7, we showed how analytical knowledge of the equivalent capacity-achieving distribution of an active learning problem could be used in APM to transform the information maximization problem of example selection to a geometric problem, which potentially offers algorithmic and computational advantages over brute-force information maximization. A critical requirement of this process is that the target capacity-achieving distribution is well-matched to the set of feasible input distributions induced by real examples. In the case of logistic regression, we ensured a close match between the target distribution and the feasible set by deriving a power constraint on the channel input based on the problem geometry, and using this constraint to derive a corresponding capacity-achieving distribution. However, for general IML problems it is unclear if there exist similar geometric constraints that could be applied to make the problem of determining an approximately feasible capacity-achieving distribution well-posed. Even if channel input constraints such as a power constraint could be derived

and imposed from the problem geometry, it is difficult in the general case to arrive at an analytical expression for a capacity-achieving distribution.

One interesting and exciting avenue to apply APM in general cases where an analytical characterization of an approximately feasible capacity-achieving distribution is unavailable is to *solve* for an appropriate target distribution using computational methods. Specifically, our approach is to solve for a capacity-achieving distribution *iteratively* in the spirit of Blahut-Arimoto algorithms [189, 190], while ensuring that the resulting target distribution remains within the realm of feasible distributions by applying an optimal transport regularizer. To briefly introduce these ideas, we restate the information maximization problem introduced in Chapter 7. For simplicity, we omit the label history $\mathcal{L}_{n-1}$ from our notation, as well as the index n of the selected example in the current iteration. We let $p_{h_\theta(x)}$ denote the distribution of random variable θ transformed through mapping $h_\theta(x)$ parameterized by a fixed example x. The information maximization problem for label distribution $p_{Y|L}$ selects the next example as

$$x^* = \arg\max_{x \in \mathcal{U}} \; I(p_{h_\theta(x)}, p_{Y|L}). \tag{8.3}$$

Letting $H(\mathcal{U}) \coloneqq \{p_{h_\theta(x)} : x \in \mathcal{U}\}$ denote the set of feasible channel input distributions, we can rewrite eq. (8.3) as an equivalent problem in terms of the most informative distribution within $H(\mathcal{U})$:

$$p_L^* = \arg\max_{p_L \in H(\mathcal{U})} \; I(p_L, p_{Y|L}). \tag{8.4}$$

This information maximizing distribution within feasible set $H(\mathcal{U})$ is in a sense the "ideal" target distribution for APM, since it maximizes information across the channel (i.e., is capacity-achieving) while remaining feasible since it is induced by an actual example.

To apply computational methods to solve for an approximation to p_L^*, we can relax eq. (8.4) using optimal transport. Noting that the 2-Wasserstein distance $W_2(p_L, q_L)$ between distributions p_L and q_L is zero if and only if $p_L = q_L$, we can rewrite eq. (8.4) as an equivalent

program by introducing an intermediate distribution q_L:

$$p_L^* = \arg\max_{p_L} \quad I(p_L, p_{Y|L})$$
$$\text{s.t.} \quad \min_{q_L \in H(\mathcal{U})} W_2^2(p_L, q_L) = 0.$$

The constraint $\min_{q_L \in H(\mathcal{U})} W_2^2(p_L, q_L) = 0$ implicitly ensures that $p_L \in H(\mathcal{U})$. By rewriting this constraint in a Lagrangian with hyperparameter $\lambda > 0$ and negating to a minimization, we can solve the unconstrained problem

$$p_L^* = \arg\min_{p_L} \left[-I(p_L, p_{Y|L}) + \lambda \min_{q_L \in H(\mathcal{U})} W_2^2(p_L, q_L) \right]. \tag{8.5}$$

At first glance, it might be unclear why the search for an APM target distribution in eq. (8.5) is more computationally attractive than brute-force information maximization over the pool of examples $\mathcal{U}$. The key insight is that in the context of eq. (8.5), optimizing over the examples in $\mathcal{U}$ (through the constraint $q_L \in H(\mathcal{U})$) serves as a optimal transport regularizer to keep p_L — which is being optimized to maximize the information gain term $I(p_L, p_{Y|L})$ — "close" to the set of feasible distributions $H(\mathcal{U})$ induced by real examples. With this interpretation in mind, it may be possible that only a few landmark examples $\widehat{\mathcal{U}} \subset \mathcal{U}$ are needed for this regularization, and q_L can be optimized within a smaller set of distributions $H(\widehat{\mathcal{U}})$:

$$p_L^* = \arg\min_{p_L} \left[-I(p_L, p_{Y|L}) + \lambda \min_{q_L \in H(\widehat{\mathcal{U}})} W_2^2(p_L, q_L) \right]. \tag{8.6}$$

Once eq. (8.6) is solved numerically with a combination of Blahut-Arimoto algorithms and modern methods in computational optimal transport [198, 199], p_L^* can then be used as a target distribution in APM against which the entire example pool is evaluated. This process invests initial computation (before evaluating specific examples in $\mathcal{U}$) in estimating an informative, approximately feasible target distribution p_L^*, a process which may only

require a small number of landmark data points $\widehat{\mathcal{U}}$ for regularization. Then, once this target distribution p_L^* is solved for, it can be used as a "template" against which individual examples $x \in \mathcal{U}$ are evaluated with an optimal transport cost according to APM. Specifically, APM searches the entire pool $\mathcal{U}$ for the example x that minimizes $W_2(p_{h(x)}, p_L^*)$, which as in APM-LR may be more computationally tractable than brute-force maximization of information gain over $\mathcal{U}$.

More generally, the decoupling of information maximization in eq. (8.6) from the task of evaluating all candidate examples in a pool makes strides in addressing a critical computational challenge in pool-based active learning of evaluating an expensive acquisition function over many candidate examples. This decoupling of information maximization from a pool-based search and conversion of the pool-based search component to a geometric optimal transport problem can potentially allow for more computationally efficient pool-based active learning. These insights and algorithmic ideas are fundamentally motivated and enabled by the coding-theoretic concepts presented in this thesis, further illustrating the benefits of directly exploring the intersections of coding theory with interactive machine learning.

Appendices

APPENDIX A
METHODS AND SUPPLEMENTARY DETAILS FOR ONE-BIT HUMAN-COMPUTER INTERACTION

A.1 Modified Burnashev-Zigangirov Algorithm

This appendix chapter is mostly dedicated to methods and supplemental material for Chapter 3. However, we first begin with a description of the Burnashev-Zigangirov (BZ) algorithm, which is the core backend algorithm used in both Chapters 3 and 4. To maintain context, we describe the algorithm in its application to Chapter 3, and conclude its presentation with a discussion of its application to Chapter 4.

In this section, our notation and conventions for dictionary construction are inspired from Omar et al. [23], and the mathematical algorithm is drawn from Castro and Nowak [35]. Let $\boldsymbol{z}_j = \{\sigma_j^h, \sigma_j^v, \sigma_j^n, \sigma_j^s\}$ denote the jth swarm configuration in the dictionary, where $\sigma_j^h, \sigma_j^v, \sigma_j^n$, and σ_j^s respectively denote character indices in the horizontal position (h), vertical position (v), number of sides (n), and size (s) alphabets. Letting N_a denote the number of characters in alphabet $a \in \{h, v, n, s\}$, we have $\sigma_j^a \in 1, 2, \ldots N_a$ with alphabet precedence corresponding to character index ordering, i.e., character σ_j^a precedes character σ_k^a in alphabet a if and only if $\sigma_j^a < \sigma_k^a$. Letting $N_d = N_h N_v N_s N_s$ denote the total number of strings in the dictionary, string $\boldsymbol{z}_j$ precedes string $\boldsymbol{z}_k$ in the total dictionary ordering if and only if $j < k$, where $j, k \in 1, 2, \ldots N_d$. Equivalently, $\boldsymbol{z}_j$ precedes $\boldsymbol{z}_k$ if and only if $\sigma_j^{a^*} < \sigma_k^{a^*}$, where a^* is the first character position where $\boldsymbol{z}_j$ and $\boldsymbol{z}_k$ differ.

With this dictionary notation established, let $Z_j = \frac{\sigma_j^h - 1}{N_h} + \frac{\sigma_j^v - 1}{N_h N_v} + \frac{\sigma_j^n - 1}{N_h N_v N_s} + \frac{\sigma_j^s - 1}{N_h N_v N_s N_s} = (j-1)/N_d$ denote a real number representation of the jth dictionary string; it is straightforward to show that $Z_j \in [0, 1)$ and that string $\boldsymbol{z}_j$ precedes $\boldsymbol{z}_k$ if and only if $Z_j < Z_k$. Each Z_j corresponds to the start of a $1/N_d$ length interval, creating a mapping between

the configuration dictionary and equally sized intervals that uniformly partition $[0, 1)$. In particular, $Z_j \in \{0, 1/N_d, 2/N_d, \ldots 1 - 1/N_d\}$, each corresponding to the start of interval $I_j = [(j-1)/N_d, j/N_d)$ with length $1/N_d$.

With this notation and real number mapping defined, posterior matching can be used as an interaction algorithm to convey the user's desired configuration. While we use the term "posterior matching" here to remain consistent with previous feedback information-theoretic BCI literature [23], the mathematical algorithm we use in this work is a discrete variation of posterior matching known as the Burnashev-Zigangirov (BZ) algorithm [35, 38]. We use this variation since our swarm dictionary is discrete and finite, and therefore our message set corresponds to a finite partition of the unit interval rather than spanning the entire interval. Still, in this work we refer to the BZ algorithm interchangeably with "posterior matching" since the differences between the two algorithms are minor.

The BZ algorithm searches for one of N_d, length $1/N_d$ intervals on the unit interval by taking adaptive one-bit measurements of points on the set $\{0, 1/N_d, \ldots 1 - 1/N_d, 1\}$ and updating a probability distribution over interval set $\{I_j\}$. We adapt this algorithm to searching over a finite, discrete dictionary by utilizing the mapping described above between each string and a subinterval I_j, and "measuring" configurations $Z_j \in \{0, 1/N_d, 2/N_d, \ldots 1 - 1/N_d\}$ by presenting the corresponding configuration as a swarm behavior to the user. Mathematically speaking, the user indicates if their configuration's real number representation is less or greater than the guess's real number value. Unlike the BZ algorithm formulation, in our setting there exists no measurement $Z_j = 1$; we address this point below. While the original BZ algorithm tracks a probability distribution over intervals, for clarity in our adaption below we describe the posterior distribution over configurations directly, rather than the intervals they correspond to.

Initialization Let the jth swarm string in the dictionary for $1 \leq j \leq N_d$ have a posterior probability after k user inputs given by $\alpha_j(k) \in [0, 1]$, corresponding to a probability

distribution over the set of intervals $\{I_j\}$. We initialize this probability distribution with a uniform prior, with $\alpha_j(0) = \frac{1}{N_d}$.

Guess selection After k user inputs, define the posterior median $N(k) \in \{1 \dots N_d\}$ as the dictionary index such that

$$\sum_{j=1}^{N(k)-1} \alpha_j(k) < 1/2, \quad \sum_{j=1}^{N(k)} \alpha_j(k) \geq 1/2. \tag{A.1}$$

Denoting the swarm guess after k inputs as $\hat{\boldsymbol{z}}(k)$, we set $\hat{\boldsymbol{z}}(k)$ to be an adjusted version of this median string. Specifically, let

$$\nu_1(k) = \sum_{j=N(k)}^{N_d} \alpha_j(k) - \sum_{j=1}^{N(k)-1} \alpha_j(k)$$

and

$$\nu_2(k) = \sum_{j=1}^{N(k)} \alpha_j(k) - \sum_{j=N(k)+1}^{N_d} \alpha_j(k).$$

The adjusted median configuration index, denoted $n(k)$, is set to $N(k)$ with probability $\pi_1(k) = \nu_2(k)/(\nu_1(k) + \nu_2(k))$, or $N(k) + 1$ with probability $\pi_2(k) = 1 - \pi_1(k)$. The swarm configuration is then updated as $\hat{\boldsymbol{z}}(k) = \boldsymbol{z}_{n(k)}$. For the edge case of $N(k) = N_d$, we set $n(k) = N_d$ with probability 1. This adjustment is due to the fact that, unlike the original BZ algorithm, there is no measurement at $Z_{N_d+1} = 1$ available.

Noisy user input The algorithm receives $Y_{k+1} = \mathrm{BSC}(X_{k+1}, p)$ from the user, i.e., the output of a binary symmetric channel ($Y_{k+1} \in \{0, 1\}$) with crossover probability $0 \leq p < 1/2$, where X_{k+1} is issued as a left-hand motor imagery input ($X_{k+1} = 0$) if the target precedes the guess $\boldsymbol{z}_{n(k)}$, or a right-hand motor imagery input ($X_{k+1} = 1$) if the target succeeds *or equals* the guess. Mathematically, if $\boldsymbol{z}_t$ is the target configuration with real number representation Z_t, then $X_{k+1} = 0$ if $Z_t < Z_{n(k)}$, and $X_{k=1} = 1$ if $Z_t \geq Z_{n(k)}$.

Update posterior Let $q = 1 - p$, and define

$$\nu = \sum_{j=1}^{n(k)-1} \alpha_j(k) - \sum_{j=n(k)}^{N_d} \alpha_j(k). \tag{A.2}$$

If $j < n(k)$, then

$$\alpha_j(k+1) = \begin{cases} \frac{2q}{1+\nu(q-p)}\alpha_j(k) & Y_{k+1} = 0 \\ \frac{2p}{1-\nu(q-p)}\alpha_j(k) & Y_{k+1} = 1 \end{cases}. \tag{A.3}$$

Otherwise, if $j \geq n(k)$ then

$$\alpha_j(k+1) = \begin{cases} \frac{2p}{1+\nu(q-p)}\alpha_j(k) & Y_{k+1} = 0 \\ \frac{2q}{1-\nu(q-p)}\alpha_j(k) & Y_{k+1} = 1 \end{cases}. \tag{A.4}$$

Algorithm notes As discussed in Section A.2, we can opt to run the BZ algorithm with a stopping criterion. To do so, a priori the BCI user must choose a threshold parameter $\tau \in [0, 1]$ which corresponds to a convergence confidence threshold. Noting that $\alpha_j(k)$ is the posterior probability after k inputs of swarm configuration j being the user's desired configuration, the algorithm halts at the first instance of $\alpha_j(k) \geq \tau$ for any j, k. Let j^* denote the configuration whose posterior probability crosses the threshold, and k^* be the number of user inputs received at this point. We say that the algorithm "converges" to swarm j^* after k^* user inputs, and j^* is selected as the final estimate of the user's configuration. Note that $1 - a_{j^*}(k^*)$ is the posterior probability of the ground truth target being a configuration other than j^* — in other words, the probability of a configuration convergence error. Therefore, $1 - \tau$ can be interpreted as a maximum error tolerance for convergence, or conversely τ is a threshold for minimum convergence accuracy.

When simulating posterior matching, for simplicity the simulation can operate directly on the unit interval, rather than needing to maintain and operate on the full set of characters $\{\sigma^h, \sigma^v, \sigma^n, \sigma^s\}$ for each dictionary string. Instead, we can simply track the posterior

distribution $\alpha_j(k)$ over the dictionary strings directly, and simulate user responses by comparing the real number representations of the current guess $Z_{n(k)}$ and the target Z_t. Note that once the dictionary size N_d is specified, such a simulation can be run without explicit knowledge of the characters that each string corresponds to, alphabets, alphabet sizes, or number of degrees of freedom. This is because the total order between configurations is fully captured by their representations Z_j on the unit interval, obviating the need to make comparisons between specific characters. Such comparisons are only relevant when a human user issues commands, since the dictionary representation is crucial for a human to be able to sort according to the total ordering. However, when running simulations on a computer we can forgo this step and operate directly on the unit interval.

A.1.1 Application to Interactive Object Segmentation

We can apply the BZ algorithm to interactive object segmentation by applying the algorithm in Section A.1 to the ellipse dictionary in EllipseLex. Listed in order of precedence, the EllipseLex alphabets correspond to each ellipse's vertical position (denoted y, with m_y alphabet characters), horizontal position (denoted x, with m_x alphabet characters), angle from the horizon (denoted θ, with m_θ alphabet characters), half-length of the major axis (denoted a, with m_a alphabet characters), and aspect ratio of the minor axis to major axis (denoted r, with m_r alphabet characters). In the notation of Section A.1, an ellipse string is then denoted as $\boldsymbol{z}_j = \{\sigma_j^y, \sigma_j^x, \sigma_j^\theta, \sigma_j^a, \sigma_j^r\}$, where $j \in 1, 2, \ldots N_d$ and $N_d = m_y m_x m_\theta m_a m_r$.

At iteration k, ellipse $\widehat{\boldsymbol{z}}(k)$ is presented to the user as feedback. We empirically observed that as k increases and $\widehat{\boldsymbol{z}}(k)$ converges to the target $\boldsymbol{z}_{j^*}$, where j^* indexes the index of the target ellipse, $\widehat{\boldsymbol{z}}(k)$ may oscillate between $\boldsymbol{z}_{j^*}$ and ellipse $\boldsymbol{z}_{j^*+1}$ due to the random selection of $n(k)$ as $N(k)$ or $N(k)+1$. For an ellipse dictionary with a fine enough resolution, this oscillatory behavior does not result in dramatic changes in F1 score. However, for cases where $j^* \bmod m_r = 0$, this oscillation results in an overflow of the minor-to-major axis ratio r such that the F1 score may oscillate dramatically, even with increasing numbers of

inputs.

To resolve this oscillatory issue, note that it only occurs when $N(k) = j^*$; in this case, the algorithm should generate $\boldsymbol{z}_{N(k)}$ as a segmentation output, even if $n(k)$ is selected as $N(k) + 1$. The user continues to observe $\boldsymbol{z}_{n(k)}$ as feedback, even though the algorithm outputs $\boldsymbol{z}_{N(k)}$ as a generated segment. This discrepancy between the ellipse delivered as feedback and the ellipse used to generate a segment for F1 score calculation is no cause for concern, since such a mismatch only occurs when the user's target ellipse is, in fact, the latter. To make this adjustment precise, continue to let $\hat{\boldsymbol{z}}(k)$ denote the ellipse presented as feedback to the user at time step k, and define $\boldsymbol{z}_o(k)$ as the ellipse selected as the segmentation output and used for F1 score calculation at step k.

If $N(k) \bmod m_r = 0$, then

$$\widehat{\boldsymbol{z}}(k) = \boldsymbol{z}_{n(k)}$$
$$\boldsymbol{z}_o(k) = \boldsymbol{z}_{N(k)}.$$

Otherwise, if $N(k) \bmod m_r \neq 0$, then

$$\widehat{\boldsymbol{z}}(k) = \boldsymbol{z}_o(k) = \boldsymbol{z}_{n(k)}.$$

A.2 Brain-computer Interfacing Methods

Protocols for both the online user study (Protocol H16266) and robot control portions (Protocol H10263) were approved by the Georgia Tech Institutional Review Board. Both studies complied with ethical regulations set by the Review Board, including online user study participants providing informed consent. GC consents to his image being published. Unless otherwise noted, all software is written and executed in MATLAB.

Dictionary construction We constructed the swarm dictionary with the following characters in each configuration string, in order of character precedence: horizontal position, vertical position, number of sides, and size. Horizontal position and vertical position refer to the coordinates of the center of each polygon, respectively (see Supplementary Figure A.1). Size refers to the distance between the polygon center and each vertex (this value is the same for each vertex since the polygons are regular). The number of characters in each alphabet is as follows: 5 horizontal positions; 2 vertical positions; 3 numbers of sides; and 2 polygon sizes. Characters in the horizontal position alphabet were chosen to uniformly span the robot arena (virtual or physical), as were the characters in the vertical position alphabet. The "number of sides" alphabet has characters given by 3, 4, or 5 sides, with the polygon rotation set by fixing a vertex at the "12 o'clock" position of each shape. The two size distances were tuned such that size differences were visually discernible, while not causing robots to overflow outside the span of the arena. With an arena of width 1.5 and height 1 (specified in units relative to the arena height), these specifications translate to the following alphabet characters: horizontal position $\{0.4, 0.575, 0.75, 0.925, 1.10\}$; vertical position $\{0.4, 0.6\}$; number of sides $\{3, 4, 5\}$; size $\{0.3, 0.4\}$. These values (except for number of sides) are specified in abstract units relative to the arena height, and are scaled at runtime to the physical dimensions of the actual swarm arena; for instance, if the physical swarm arena is 2.5 feet in height, then the first horizontal position character is $0.4 \times 2.5 = 1$ foot from the left arena edge. In total, this combination of alphabets produces a dictionary with $5 \times 2 \times 3 \times 2 = 60$ total possible polygons, and hence 60 possible swarm behaviors.

Online user study We conducted the online user study via Amazon Mechanical Turk[1] by creating a Human Intelligence Task (HIT) for participant submission. The HIT contained both a set of graphical and text instructions teaching the swarm dictionary to the participant, followed by a set of 150 shape pair queries. Once a participant accepted a HIT task, they proceeded to read the instructions, answer all queries, and submit their responses. In total,

[1] https://www.mturk.com/

150 participants were recruited in the study, corresponding to 150 submitted and accepted HITs. Each shape pair query presented a blue, solid shape and a red, dashed shape as in Figure 3.3b (polygon outlines were presented rather than actual swarm configurations), and asked "For the image below, select whether the test shape (red dashed edges) comes after or before the reference shape (blue solid edges), as defined in the instructions above.", which the participant responded with "Before" or "After." During the study, each participant had access to an informational graphic presented in Figure 3.3a as a visual aid in recalling the dictionary ordering.

The 150 shape pairs were randomly generated in such a way that the critical character determining the order of each pair was evenly distributed across all four letters. Within this query set, 6 "cheat detection" pairs were presented each consisting of two identical triangles with all the same parameters except horizontal position, which is an "easy" question and is unlikely to be answered incorrectly unless a participant is randomly selecting answers to finish the study as quickly as possible (Supplementary Figure A.2); the participants were not told that these pairs were used for cheating detection. Before approving a participant's HIT submission, we evaluated their responses on these cheat detection queries to assess if they were simply selecting answers at random. The remaining 144 queries were evenly distributed between shape pairs where the horizontal position, vertical position, number of sides, or size was the first character to differ between the two configurations in question (36 shape queries per critical character, resulting in $36 \times 4 + 6 = 150$ total pairs).

To generate a shape pair with the desired critical character (36 pairs for each critical character), a character was randomly generated for each alphabet that precedes the critical character, and set for *both* shapes in the pair. This way, the critical character would in fact be the first character that differs between the shapes in question. Next, two *distinct* characters were randomly selected from the critical character alphabet, one for each shape in the pair. Finally, the remaining characters succeeding the critical character were randomly populated separately for each shape in the pair. All generated shape pair queries (including pairs for

cheating detection) were then shuffled into a random order. Although the query order was randomly shuffled, each HIT (and therefore each participant) responded to the same fixed order of queries; in other words, query order was not randomized *between* participants.

To qualify for participation in the study, participants must have had a record of at least 1,000 approved HITs from previous tasks on Mechanical Turk, and must have had an overall HIT approval rate of 95% or greater at the time of submission. After qualifying participants accepted and completed our HIT, they were automatically approved unless flagged as being suspect of randomly selecting answers, in which case they were manually reviewed. Participants were automatically rejected if they did not answer every query, or if they had already completed the HIT previously. The details of this process are presented in Section A.3.1. Each approved participant was paid $8 for completing all pair orderings, and was awarded a $4 bonus if they achieved an accuracy of 95% or higher of correct pair orderings. Overall, 150 participants were recruited, of which all 150 were approved. Of these, 125 achieved an overall response accuracy of over 95% and so were awarded a $4 bonus.

The 6 cheat detection queries were omitted during data analysis, resulting in 144 shape pairs analyzed per participant. Overall sorting accuracy was calculated per subject as the fraction of correct responses to these 144 regular queries. Sorting accuracy was calculated per critical character as the fraction of correctly answered queries among the 36 shape pair queries with the respective critical character. Distributions are plotted in Figures 3.3c and 3.3d as kernel density estimates.

Robot swarm setup The Robotarium arena and its virtual counterpart, both provided by the Georgia Robotics and InTelligent Systems Laboratory (GRITS), were used as swarm operating spaces. The Robotarium [200] is a remotely accessible, multi-robot research facility that provides global position and orientation tracking of fiducial markers placed on each robot, a WiFi communication infrastructure to broadcast information to the robots,

and an automatic recharging mechanism. The robot swarm consists of GRITSBots [201], which are differential-drive wheeled mobile robots with WiFi communication and infrared range-sensing capabilities. These robots may be modeled as unicycles, i.e., for the i^{th} robot in the swarm, the planar position $p_i = (x_i, y_i)$ and orientation θ_i follow the dynamics given by

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) & 0 \\ \sin(\theta_i) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_i \\ \omega_i \end{bmatrix},$$

where v_i, ω_i are its linear and angular velocities, respectively. The Robotarium API [202] provides a simulator that enables the testing of algorithms in a virtual setting prior to deployment in the real robots.

Each swarm configuration (physical or virtual) consists of ten robots ($n = 10$), which collectively conform to a specified coverage density $\phi(q, t) \in (0, \infty)$ which describes the desired distribution for all points q in the space $D \subset \mathbb{R}^2$ at time t [90]. The robots achieve this distribution by finding an optimal configuration with respect to the the locational cost [203] as weighted by the reference density ϕ, defined as

$$\mathcal{H}(p_i, t) = \sum_{i=1}^{n} \int_{V_i} \|q - p_i\|^2 \phi(q, t)\, dq,$$

where the $V_i \subset D$ form a Voronoi tessellation of the space using the position of the robots as generators, and properly partition D. The optimal configuration is achieved through a distributed control law [90] which relies only on nearby neighbor information, given by

$$\dot{p_i} = \kappa(c_i(p_i, t) - p_i) + \frac{\partial c_i}{\partial t} + \sum_{j \in \mathcal{N}_i(t)} \frac{\partial c_i}{\partial p_j} \left(\kappa(c_j(p_j, t) - p_j) + \frac{\partial c_j}{\partial t} \right),$$

where $\kappa > 0$ is a tuning parameter, $c_i(p_i, t)$ is the center of mass for V_i, and $\mathcal{N}_i(t)$ is the

set of robots near robot i at time t. This control law is mapped into the unicycle dynamics through a near-identity diffeormorphism [204]. Specifically, for $\lambda > 0$ the linear and angular velocities are obtained as

$$\begin{bmatrix} v_i \\ \omega_i \end{bmatrix} = \begin{bmatrix} \cos(\theta_i) & \sin(\theta_i) \\ -\frac{1}{\lambda}\sin(\theta_i) & \frac{1}{\lambda}\cos(\theta_i) \end{bmatrix} \dot{p}_i.$$

To use this interface, the abstract polygons in our dictionary need to be translated to a continuous density function describing swarm coverage. This was achieved by constructing a Gaussian mixture model (GMM) from the vertices and edges of a given polygon. Specifically, we placed an isotropic Gaussian distribution at each polygon vertex, and on each edge we placed two Gaussian distributions with means evenly spaced between the edge's vertices, and with a 10/1 ratio of variance parallel to the edge to variance perpendicular to the edge (see Supplementary Figure A.3). To define this GMM more formally, let $v_1 = [x_1, y_1]^T$ and $v_2 = [x_2, y_2]^T$ denote two vertex coordinate pairs connected by an edge, and let $w = 2(v_2 - v_1)/3$. An isotropic Gaussian distribution with coordinate-wise variance of $0.007\|w\|_2$ was placed at each vertex, in units relative to the arena height. Two additional Gaussian distributions with means at $v_1 + w/2$ and $v_1 + w$ were placed on the edge between v_1 and v_2, each with a covariance matrix of

$$\Sigma = \begin{bmatrix} \frac{w}{\|w\|_2} & \frac{w^\perp}{\|w\|_2} \end{bmatrix} \begin{bmatrix} 0.07\|w\|_2 & 0 \\ 0 & 0.007\|w\|_2 \end{bmatrix} \begin{bmatrix} \frac{w}{\|w\|_2} & \frac{w^\perp}{\|w\|_2} \end{bmatrix}^T \quad \text{where } w^\perp = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} w.$$

This GMM was then transmitted to the Robotarium using User Datagram Protocol (UDP) packets via WiFi.

Motor imagery input classification In order for the user to provide a binary input through the use of a mental command detected by EEG, raw signals from scalp electrodes must be processed and subsequently classified into one of two commands. Although EEG

is associated with low spatial resolution and high sensitivity to noise, its high temporal resolution can be leveraged to extract simple mental commands from electrical activity. In the case of motor imagery, it has been shown that mental imagery of left or right hand dorsiflexions produces discernible EEG features over different spatial regions on the scalp [92]. Specifically, left and right hand motor imagery produces a decrease in the power of the mu (8-12 Hz) and beta (18-26 Hz) bands over the contralateral side of the scalp (a phenomenon called *event-related desynchronization*, or ERD), and sometimes produces an increase of power in these bands over the ipsilateral side (called *event-related synchronization*, or ERS) [53, 92]. If these signature changes in power spectra can be recognized, then binary classification can be performed to detect left or right hand motor imagery.

To built such a motor imagery classifier with acceptable accuracy, we adopt a procedure that combines protocols from a series of studies related to optimal spatial filtering of EEG signals for motor imagery classification [93, 205, 99, 206, 207]. At a high level, the method first temporally filters EEG measurements in an ERD/ERS frequency range of interest, then trains spatial filter coefficients that maximize the signal power in one motor imagery class and minimize it in the other. This spatial filtering process, known as Common Spatial Patterns (CSP) filtering, yields features that discriminate between power spectrum changes due to different motor imagery classes. Finally, these filtered and processed features are classified with a linear discriminant analysis (LDA) classifier.

EEG measurements are sampled at 2 kHz from a 32-electrode BioSemi ActiveTwo system. The use of CSP filtering requires the use of at least 18 electrodes over the motor cortex [205]; here, we record electrodes F3, Fz, F4, FC5, FC1, FC2, FC6, T7, C3, Cz, C4, T8, CP5, CP1, CP2, CP6, P3, Pz, and P4 based on the International 10/20 system. BioSemi ActiView[2] is used to monitor EEG signal quality during scalp recording setup. Signals are downsampled to 128 Hz and referenced using the Common Average Reference

[2]https://www.biosemi.com/download.htm

(CAR), which subtracts the mean of all electrodes from each individual signal [208]. Then, signals are temporally filtered with a 3rd order Butterworth notch filter centered at 60 Hz with a band of 57-63 Hz and a pass band ripple of 0.5 dB, a 6th order Butterworth band pass filter with a band of 0.5-50 Hz and a pass band ripple of 0.5 dB, and a 6th order Butterworth bandpass filter with a band of 8-30 Hz and a pass band ripple of 0.5 dB to limit the considered frequencies to the mu and beta ranges [206].

In order to detect the power spectrum changes due to ERD/ERS during motor imagery, the choice of spatial filter coefficients among electrodes must be optimized to maximally discriminate between left and right hand motor imagery. The CSP method is ideal for this type of discrimination since it maximally distinguishes between intraclass signal power, which directly translates to the discrimination of ERD/ERS activity and therefore to the detection of binary motor imagery commands (see Section A.3.1 for details of CSP training). The two most discriminative CSP filters per class (four filters total) are applied to spatially filter the temporally filtered signals, yielding a signal with four channels. A temporal average of the square of each channel is taken over a window of length T with an offset of t_0 seconds (see below for details of parameter selection), resulting in an average power p_i for channel $i \in \{1, 2, 3, 4\}$. The final feature vector f of length four is then constructed by taking the natural log of each channel power p_i, normalized by the total power across all channels, i.e. $v_i = \ln\left(\frac{p_i}{\sum_{i=1}^{4} p_i}\right)$. Finally, this feature vector is passed through a binary linear discriminant analysis (LDA) classifier [209] to extract the issued left or right hand motor imagery command. We summarize this process in the feature extraction portion of Supplementary Figure A.6.

CSP filters and LDA classifiers are trained with a procedure adapted from Guger et al. [207]. The BCI user sits in front of a monitor and imagines left or right hand dorsiflexions according to a corresponding left or right arrow cue which appears on screen (Supplementary Figure A.5). During a training session, each motor imagery class (left or right hand) is presented for 30 synchronized recorded training points, with all 60 inputs presented

in a randomized order. During each synchronized training point recording, a fixation cross appears for 2 seconds, at which point a left or right arrow cue is displayed for 1.25 seconds, prompting the subject to imagine the corresponding movement. The fixation cross remains for 3.75 seconds after, during which the subject continues to imagine the instructed movement. This results in a total training interval of 5 seconds. The cross is then cleared, followed by an inter-stimulus-interval of uniformly randomly length between 1 and 2.5 seconds. Windows at a length of $T = 4$ s offset by 0.5 s are extracted from the 5 second training interval (e.g., windows with $t_0 = 0$ s, $t_0 = 0.5$ s, or $t_0 = 1$ s) and used to train CSP filters and LDA classifiers based on the signal processing procedure described previously. 10x10 cross-validation is used to evaluate the accuracy of each 4 second window over all training data, and the best 4 second window is selected to use for synchronous user inputs using during testing.

If a cross-validation accuracy of 0.7 is exceeded for the best 4 second window, the feature extraction and classifier pipeline is considered trained. Otherwise, additional sessions of 15 training points for each class are collected until some subset of training sessions results in filters and a classifier with a cross-validated accuracy of at least 0.7. For instance, suppose a first training set of 30 data points per class, labeled as dataset S_1, does not result in a sufficient cross-validation accuracy. Then, a second training session of 15 data points per class is run, resulting in an additional training dataset labeled S_2. The same filter, classifier, and window extraction procedure described above is performed individually on S_2 and $S_1 \cup S_2$, and the best model saved. If this model's cross-validated accuracy does not exceed 0.7, another training set of 15 data points per class is collected, resulting in training dataset S_3. The best model from $S_3, S_1 \cup S_3, S_2 \cup S_3$, and $S_1 \cup S_2 \cup S_3$ is saved; this procedure continues until a trained model exceeds the 0.7 threshold. The final cross-validated error from the saved model is used to estimate the crossover probability parameter in the posterior matching procedure during testing.

During testing, the distance-to-hyperplane output of the LDA classifier is used to create

a feedback bar updated in real-time to aid the user in tuning their motor imagery features [210]. The feedback bar points in the direction of the classifier's detected input (left or right) and has a length proportional to the distance from each instantaneous feature vector f to the classifier hyperplane. As we describe in Section A.3.2, this distance is a direct measure of classification confidence. Feedback is generated over $T = 1$ second windows overlapped by 0.0625 seconds, and is displayed continuously during the entire testing phase.

OpenVibe[3] is used for the real-time collection and processing of EEG signals, with CSP filters and LDA classifier training performed offline in MATLAB. During testing, the lab streaming layer[4] communication protocol is used to interface in real-time between signal acquisition, feature extraction, and feedback presentation in OpenVibe, and feature classification and posterior matching operation in MATLAB.

Swarm control trials In order to demonstrate SCINET's performance, GC (henceforth referred to as "the subject") learned the dictionary ordering for swarm configurations, trained CSP filters and LDA classifiers with left/right imagined dorsiflexions, and evaluated his swarm control ability using a virtual Robotarium arena over 70 trials. On each day (with one session of trials per day), the subject sat in front of two monitors, one of which presented the visualizations required for training and feedback for testing (run on a PC laptop) and the other ran the Robotarium simulation in MATLAB (run on a MacBook laptop). At the start of each session, the subject trained spatial filters and classifiers using the aforementioned procedure until the specified training threshold was met. Then, the subject performed 10 test trials per day on a Robotarium simulation. For each test trial, a target swarm configuration was selected randomly without replacement from a set of possible targets and displayed in the simulator as a blue outline (as in Figure 3.4b). The target set was constructed as a single copy of each string in the dictionary (60 total), plus 40 additional copies that are evenly spaced throughout the dictionary (for a total of 100 configurations).

[3] http://openvibe.inria.fr/

[4] https://github.com/sccn/labstreaminglayer

The subject then issued the appropriate motor imagery commands to steer the swarm to each specified target configuration according to the posterior matching algorithm (see section A.1 for a mathematical algorithm description). For the special case where a target and guess configuration were equal, a "right" command was issued. The subject issued each command in a synchronized input window of 5 seconds in length. After each command was issued, a new configuration was broadcast to the robot swarm, and the robots readjusted their positions while the subject waited and observed their movement. After each robot's velocity fell below a prespecified threshold, the swarm controller detected that the swarm had settled on a single configuration and another input was requested from the subject. At this point, the subject heard a single audible beep, which indicated that the swarm had made its guess, and that they should decide on their next input. After a two-second pause, the subject heard three more beeps, each separated by a single second, to count down to the start of the synchronized input window. A final beep signaled the start of a 5 second input window, during which the subject visually fixated on the real-time feedback bar. A single beep signaled the end of the synchronization window, at which point the subject could stop their command. Feature extraction and classification was performed using the same CSP filters, LDA weights, and timing parameter t_0 for extraction of a $T = 4$ second window as during training. After each input was issued the system indicated the classification result on-screen with a left or right arrow and the swarm rearranged to its updated configuration, after which a new input window began and the subject observed the swarm as feedback for their next command (Supplementary Figure A.4).

This process iterated until the posterior matching algorithm converged to a final estimate of the subject's configuration, at which point three short, audible beeps were played. Convergence was defined by the algorithm maintaining a posterior distribution for the subject's target configuration, and stopping when any configuration met or exceeded a prespecified posterior threshold. A single trial ended at the sooner of posterior matching converging or the number of issued inputs reaching a maximum of 50 inputs. When the trial ended by either

means, the maximum posterior probability configuration was selected as the algorithm's final estimate.

The threshold for the convergence stopping criterion was selected from a lookup table of convergence thresholds specified for various BSC crossover probabilities and desired trial lengths. For a given crossover probability, 500 posterior matching simulated trials (described below) were performed offline for each of several candidate thresholds, and the corresponding table entry was set as the threshold that achieved the highest convergence accuracy while not having an average number of inputs greater than the specified trial length. Our specified average trial length for threshold lookup was set to 25 inputs, which is an estimated number of synchronous inputs an EEG user can issue before becoming fatigued. The lookup table was constructed by evaluating crossover probabilities from 0 to 50% at increments of 5%, and posterior stopping thresholds of 0% to 100% at increments of 5%. If the model's crossover probability (i.e., the trained classifier's cross-validation error) did not appear in the lookup table, the next highest crossover probability in the table was used for lookup.

To compute the configuration accuracy and expected number of input values in the lookup table, each posterior matching simulation trial used the specified crossover probability and candidate stopping threshold. Unlike the simulations described for modeling a non-stationary input error profile, here each crossover probability used to generate input errors was fixed throughout the entire simulation trial, and this generated error crossover was equal to the crossover assumed by posterior matching in its posterior distribution updates. In each simulation trial, a target string was selected at random from the configuration dictionary, and the rules of posterior matching followed to simulate the role of a user. Each simulated user input was passed through a simulated BSC with a fixed crossover probability at the specified value. Each simulated trial was run for a maximum of 50 inputs, as was the case for the virtual swarm control experiments.

The subject engaged in 7 total days of completed trials spread over the course of 3 weeks,

with 10 trials performed per day. On each day, the subject trained the EEG classifier using the aforementioned procedure, completed 5 virtual swarm control trials, took a rest period, and then completed 5 more trials. During one particular session, the subject perceived that the EEG classifier feedback bar was qualitatively deteriorating after the first 5 trials, and added 2 additional training sessions of 15 data points per class to the training set for the second half of the session. On the other 6 days, both sets of 5 trials used the same initially trained classifier. There was an 8th day of trials omitted from this study. On this day, the subject trained the classifier as above and completed 3 trials on this day, but aborted the session due to a feeling of complete loss of ability to issue motor imagery inputs. Upon further investigation, it was found that these 3 sessions had a net EEG input error of 53%, explaining the lack of control. These two ad hoc adjustments (additional training, aborted session) are justifiable since the purpose of this experiment is to evaluate SCINET's overall performance under the assumption of a reasonably trained and sustainable EEG classifier.

Realistic simulation baseline To fit a non-stationary crossover probability model to the empirical data in Figure 3.4c for use in a realistic SCINET simulation, an input error profile was modeled by first fitting a least-squares cubic regression to the empirical crossover probability curve (Supplementary Figure A.7a). The data points corresponding to one issued input, the minimum point, and maximum point of this cubic function were then used to fit a piecewise cubic Hermite interpolating polynomial (PCHIP), where the maximum point was held until the maximum number of inputs (Supplementary Figure A.7b). The motivation behind this procedure was to generalize the crossover behavior at lower numbers of inputs while enforcing monotonicity as the number of inputs increased, since a decrease in crossover probability would not realistically model factors such as user fatigue increasing with more inputs. The resulting PCHIP was used to generate input errors in our realistic SCINET simulation. Specifically, at input number i, the correct posterior matching response was corrupted with a Bernoulli error (statistically independent of all previous and future

errors) with bias given by the PCHIP value at input i.

Even though input errors were generated according to the PCHIP, during each posterior matching iteration the simulator modeled a binary symmetric channel with a *fixed* crossover probability. This simulates the real-world effect of the trained classifier producing a cross-validation error that is used as the BSC crossover probability estimate for each trial, yet during the trial the BCI's actual input error statistics may change with additional inputs. We set the simulator's fixed crossover estimate as the average input error across all inputs and all virtual swarm trials, which evaluated to 21.8%. This value serves as an estimate of the aggregate error to be expected over the course of a virtual swarm trial.

Once the PCHIP error generator was fit and the fixed crossover probability set, the posterior matching simulation was run for 10,000 trials. At the start of each trial, a configuration was selected uniformly at random from the dictionary to serve as a target for posterior matching. We implemented the same stopping criteria for each trial as in the virtual swarm trials performed by the subject: the posterior convergence threshold was selected from the same lookup table of thresholds using the same procedure, and a maximum of 50 inputs per simulation trial was enforced. When comparing simulation results against empirical results from virtual swarm control in Figure 3.4d, trials were binned by convergence time as follows: "Short" trials converged between 1 and 12 inputs (inclusive); "Medium" trials converged between 13 and 18 inputs (inclusive); and "Long" trials converged between 19 and 50 inputs (inclusive). The number of trials converged in each bin were 24 Short, 25 Medium, and 21 Long virtual swarm trials, and 2,786 Short, 3,748 Medium, and 3,466 Long simulated trials.

The subject also demonstrated two successful trials on the physical Robotarium system, but the quantity of these trials was limited due to laboratory demand for the system and practical considerations such as robot battery life. We implemented the same EEG training and experimental setup procedures as in the virtual swarm sessions, with the only difference being that physical robots responded to user commands rather than virtual robots.

Generalized simulation results To generalize the performance of SCINET to arbitrary dictionaries, the simulator from Figure 3.4d was modified slightly. To fully evaluate the tradeoff between achieved configuration accuracy and required number of inputs for each dictionary size, we disabled convergence for both posterior matching and stepwise search (see Section A.3.1 for a mathematical description of stepwise search) and instead output an instantaneous configuration estimate after each issued input. After k inputs, this instantaneous estimate was taken as the configuration with maximum posterior probability, i.e., $\boldsymbol{z}_{j^*}$ where $j^* = \arg\max_{1 \leq j \leq N_d} \alpha_j(k)$ (see Section A.3.1). This maximum a posteriori (MAP) estimate is distinct from the guess produced during each algorithm interaction with the user, and is used only for analytical purposes to produce an error estimate. By outputting an instantaneous guess after each input and computing its configuration accuracy, we can directly observe the tradeoff between obtainable configuration accuracy and number of inputs for each algorithm and dictionary size.

In Figures 3.5a to 3.5c, a fixed 10% crossover probability was assumed by each algorithm during posterior updating, and the same crossover probability was used to generate input errors. In Figures 3.5d to 3.5f, as in the comparison against virtual swarm trials the PCHIP error profile of Figure 3.4c was used to generate Bernoulli input errors at each number of inputs, while each algorithm assumed a crossover probability of 21.8% for posterior updating. Each simulation — posterior matching or stepwise search, each run with fixed or non-stationary crossover probabilities — was repeated for 1,000 trials, with the target configuration selected uniformly at random from the dictionary at the beginning of each trial.

To evaluate performance on various dictionary sizes, the dictionary size parameter N_d in posterior matching and stepwise search was varied over simulations (see Section A.3.1 for parameter definition). Each setting of N_d corresponds to a different number of controllable dictionary degrees of freedom. To establish this relationship, we consider a dictionary with b characters for each of r alphabets, corresponding to r degrees of freedom. The total

number of strings in the dictionary is then $N_d = b^r$. To select an alphabet size b, we used the rounded harmonic mean of our alphabet sizes (i.e., 5,2,3,2) which evaluates to 3 characters. We generated dictionaries with $r = 2, 4, 6, 8$ degrees of freedom, corresponding to sizes of $N_d = 9,\ 81,\ 729,$ and 6,561 respectively. Note that each algorithm operates on the total order of strings in the dictionary without regard to individual alphabets, and the only parameter that affects simulation results is the dictionary size, rather than the exact alphabet size or degrees of freedom. However, formulating the dictionary size parameter in terms of alphabet size and degrees of freedom allows us to draw connections as in Figure 3.5 between these various parameters. We also performed the same experiments using an alphabet size of $b = 5$ (see Supplementary Figure A.14) and resulting dictionary sizes of $N_d = 25,\ 625,\ 15,625,$ and 390,625. This experiment corresponds to a more conservative relationship between degrees of freedom and dictionary size; keeping degrees of freedom fixed and increasing alphabet size results in a larger dictionary, and therefore more strings to search over (and more user inputs required) to control the same number of degrees of freedom. For this reason, we call the $b = 3$ case the "standard" degrees of freedom estimate, and $b = 5$ the "conservative" degrees of freedom estimate.

In Figures 3.5a and 3.5d, ITR was calculated from the error-free accuracy in Figures 3.5b and 3.5d respectively. At k inputs issued, let P_k denote the error-free accuracy, which is calculated as the number of trials where the kth instantaneous estimate (i.e., $\boldsymbol{z}_{j^*}$) equals the ground truth target configuration, divided by the total number of simulation trials (1,000). ITR, denoted after k inputs as R_k, is then calculated in units of bits as [53]

$$R_k = \log_2 N_d + P_k \log_2 P_k + (1 - P_k) \log_2 \frac{1 - P_k}{N_d - 1}.$$

ITR represents the aggregate amount of information about the target configuration conveyed after k inputs from the user to the swarm. ITR can be also interpreted mathematically as the bit rate over a discrete memoryless channel where the target is selected with probability P_k,

and any remaining configuration is erroneously selected with an equal probability of $\frac{1-P_k}{N_d-1}$.

To calculate absolute deviation in Figures 3.5c and 3.5f, let $Z_{j^*}(k)$ and Z_t denote the unit interval representations (see Section A.3.1) of the MAP estimate after k inputs and the target configuration, respectively. Then absolute deviation, or "dictionary distance," is calculated as $|Z_{j^*}(k) - Z_t|$, and averaged over all trials for each simulation.

A.3 Brain-computer Interfacing Supplementary Material

A.3.1 Supplementary Methods

Cheating Detection

We now describe the procedure used to flag and examine online user study participants suspected of selecting answers at random. To automatically flag suspicious sets of responses, we established a set of benchmarks evaluating response time, overall accuracy, accuracy per character, and accuracy over the course of the HIT; a participant failing any of these benchmarks resulted in a manual approval or rejection of their HIT. A participant was flagged for suspicious duration if they completed the entire task (including reading instructions and submitting answers) in under 25 minutes, which may indicate that a sincere effort was not made to answer each query carefully. This duration threshold was estimated by the authors as the approximate time our HIT might take to complete at a reasonable completion pace.

When evaluating cheat detection queries (6 total), a participant was flagged if they correctly answered exactly 2, 3, or 4 queries. Conversely, a participant passed this benchmark if they answered 5 or 6 cheat detection questions correctly — performing as expected — or if they answered only 0 or 1 test queries correctly, as might be the case if they made sincere efforts but had a reversed understanding of the dictionary ordering.

A participant was flagged for suspicious overall accuracy if their p-value for a two-tailed hypothesis test exceeded a threshold of 10%, with a null hypothesis of chance selection (50% probability of correct selection per query). We estimated this p-value with a normal

approximation to a binomial distribution with a 50% bias over 150 queries (including cheat detection queries). The p-value is computed as the probability of an overall accuracy at or further from chance than the measured accuracy. Specifically, for c correct responses over 150 shape pair queries and letting Φ denote the standard normal cumulative distribution function (c.d.f.), this p-value is calculated as

$$p_1(c) = 2\min\left(\Phi\left(\frac{\frac{c}{150} - 0.5}{\sqrt{\frac{0.25}{150}}}\right), 1 - \Phi\left(\frac{\frac{c}{150} - 0.5}{\sqrt{\frac{0.25}{150}}}\right)\right). \tag{A.5}$$

Two additional benchmark's were evaluated involving p-value evaluation per critical character, and error performance across the HIT duration. In the former, p-values were calculated as in eq. (A.5) when binning trials (including the six test questions) by critical character. These p-values were multiplied to form a "net" p-value, which flagged a participant when it exceeded 10%. To evaluate error performance throughout the duration of a HIT, an ordinary least-squares regression was fit to the cumulative number of correct responses throughout the course of the HIT. If the r^2 value of the linear model fell below 0.64, then the participant was flagged. The motivation behind this test was to detect participants who performed well initially, but then decided to select random answers for the remainder of the HIT; such a participant would show a highly nonlinear error performance over the course of the HIT, unlike participants who answered consistently according to their understanding of the task. Additionally, such nonlinear behavior would not be accounted for by a varying difficulty level over multiple queries, since the sequence of queries was presented in a shuffled order. No participants were flagged for this linear model test, and so we omit its details from the discussion here. Only one participant (participant 145) was flagged for their "net" p-value surpassing 10%, but this participant was also flagged for total accuracy (eq. (A.5)) and so we only discuss the latter.

Participants 2, 131, and 145 were flagged for cheat detection query responses (each answering 4 test queries correctly out of 6), and participants 131 and 145 were additionally

flagged for overall accuracy. Participant 2 had a duration of 54 minutes, and so was approved due to an assumption of genuine effort due to their extensive completion time. Participants 131 and 145 had completion times of 85 and 58 minutes respectively, and so were also approved due to extensive completion time. Participant 33 was flagged for total accuracy only; due to a duration of 65 minutes and answering all 6 cheat detection queries correctly, they were approved. Participants 4, 41, 43, 51, 61, 63, 68, 71, 80, 86, 95, 106, 125, 139, 142, and 144 were flagged for duration only. None of these participants had an accuracy below 96% (over all 150 queries), and all had a duration of at least 14 minutes. Since the durations were still significant and all performed at high accuracy, these participants flagged for durations were approved since their behavior did not indicate random selection. Overall, all 150 participants were approved in this study.

Common Spatial Patterns Filtering

Common Spatial Patterns (CSP) filtering is a supervised spatial filtering method that maximizes the difference in filtered signal variances between two classes [205, 206]. This separation is useful for motor imagery detection, which uses signal power (i.e., signal variance) as the classification feature. We briefly summarize our implementation of the CSP algorithm [205, 206] below, with notation and derivations drawn largely from Ramoser et al. [205].

Let $\{X_{c,i}\}_{i=1}^{N_c}$ denote the training set of N_c temporally filtered EEG signals for class $c \in \{l, r\}$ (for "left" and "right"), where each $X_{c,i}$ is a $T \times d$ matrix of T EEG samples over d channels. Let $\mu_{c,i} = \frac{1}{T}\sum_{t=1}^{T} X_{c,i}[t,:]$ denote the spatial mean of signal block i for class c, where $X_{c,i}[t,:]$ denotes the tth row of matrix $X_{c,i}$. We define the zero-mean signal matrix $\widetilde{X}_{c,i}$ by subtracting the spatial mean from each sample, i.e., $\widetilde{X}_{c,i}[t,:] = X_{c,i}[t,:] - \mu_{c,i}$. Then, we compute the averaged covariance matrices for both classes as

$$C_l = \frac{1}{N_l}\sum_{i=1}^{N_l} \frac{\widetilde{X}_{l,i}^T \widetilde{X}_{l,i}}{\mathrm{Tr}(\widetilde{X}_{l,i}^T \widetilde{X}_{l,i})} \qquad C_r = \frac{1}{N_r}\sum_{i=1}^{N_r} \frac{\widetilde{X}_{r,i}^T \widetilde{X}_{r,i}}{\mathrm{Tr}(\widetilde{X}_{r,i}^T \widetilde{X}_{r,i})}$$

and form the composite covariance matrix as

$$C = C_l + C_r. \tag{A.6}$$

We then factor C into its eigendecomposition $C = U\Lambda U^T$, where the eigenvalues in diagonal matrix Λ are sorted in descending order and the columns of U are orthogonal eigenvectors. Note that Λ only has at most $d-1$ positive eigenvalues. To see this, note that since the raw EEG signals are processed with CAR referencing, each $X_{c,i}$ has rank at most $d-1$ since by definition its columns are linearly dependent, i.e., $\sum_{k=1}^{d} X_{c,i}[t,k] = 0$ for all $1 \leq t \leq T$. $\widetilde{X}_{l,i}$ also has rank at most $d-1$ since

$$\begin{aligned}
\sum_{k=1}^{d} \widetilde{X}_{c,i}[t,k] &= \sum_{k=1}^{d} \big[X_{c,i}[t,k] - \mu_{c,i}[k]\big] \\
&= \sum_{k=1}^{d} \left[X_{c,i}[t,k] - \frac{1}{T}\sum_{j=1}^{T} X_{c,i}[j,k]\right] \\
&= \sum_{k=1}^{d} X_{c,i}[t,k] - \frac{1}{T}\sum_{j=1}^{T}\left(\sum_{k=1}^{d} X_{c,i}[j,k]\right) \\
&= 0 - \frac{1}{T}\sum_{j=1}^{T} 0 \\
&= 0.
\end{aligned}$$

Next, consider the $d \times T(N_l + N_r)$ matrix given by the horizontal concatenation of all $\{\widetilde{X}_{c,i}^T\}_{i=1}^{N_c}$, $c \in \{l, r\}$, i.e., $\widetilde{X} = [\widetilde{X}_{l,1}^T, \widetilde{X}_{l,2}^T, \ldots \widetilde{X}_{l,N_l}, \widetilde{X}_{r,1}^T, \widetilde{X}_{r,2}^T, \ldots \widetilde{X}_{r,N_r}]$. Clearly $\widetilde{X}$ has rank at most $d-1$, since $\sum_{j=1}^{d} \widetilde{X}[j,:] = 0$ by construction. Therefore the column space of $\widetilde{X}$, i.e., the span of $\{\widetilde{X}_{c,i}^T\}_{i=1}^{N_c}$, $c \in \{l, r\}$, has dimension at most $d-1$. By construction, the columns of C (eq. (A.6)) lie in this same column space, and so C has rank at most $d-1$.

Therefore, before the whitening stage of CSP, we truncate Λ to the top $d-1$ eigenvalues, resulting in $d-1 \times d-1$ matrix $\widehat{\Lambda}$, and truncate $\widehat{U}$ as the first $d-1$ columns of U. We then form the whitening transformation $P = \sqrt{\widehat{\Lambda}^{-1}}\widehat{U}^T$, so that $PCP^T = \sqrt{\widehat{\Lambda}^{-1}}\widehat{U}^T\widehat{U}\widehat{\Lambda}\widehat{U}^T\widehat{U}\sqrt{\widehat{\Lambda}^{-1}} =$

I. Let $S_l = PC_lP^T$ and $S_r = PC_rP^T$. Then S_l and S_r share eigenvectors with eigenvalues that sum to unity, i.e.,

$$S_l = B\lambda_l B^T \qquad S_r = B\lambda_r B^T \qquad \lambda_l + \lambda_r = I.$$

To see this, note that $S_l + S_r = P(C_l + C_r)P^T = PCP^T = I$, and suppose that b is an eigenvector for S_l with eigenvalue λ_l. Then $S_l b = \lambda_l b$, and so

$$b = Ib = (S_l + S_r)b = S_l b + S_r b = \lambda_l b + S_r b \implies S_r b = (1 - \lambda_l)b$$

and therefore b is an eigenvector for S_r with eigenvalue $1 - \lambda_l$. This implies that projecting onto the eigenvectors of S_l with the largest eigenvalues will result in variance separation between whitened data from classes l and r: whitened data from l will remain mostly unattenuated, while whitened data from class r will have attenuated signal energy. The reverse relationship is true when projecting onto the top eigenvectors of S_r instead.

The final CSP filtering matrix is then constructed as $W = B^T P$, where the columns of B are sorted in decreasing order from largest to smallest corresponding eigenvalues of S_l. This transformation whitens incoming data and projects it onto each eigenvector of S_l, such that the resulting vector has components with separated energy levels for each class. This $d - 1 \times d$ filter is applied to a $T \times d$ signal X as $Z = XW^T$, where Z is of size $T \times d - 1$. In SCINET, we apply a truncated CSP filter that only uses the top two spatial filters for each class. Letting w_i^T denote the ith row of W, we use only rows $1, 2, d-2, d-1$ of the filter (corresponding to the top two eigenvectors of each class), i.e., we apply filter

$$\widehat{W} = \begin{bmatrix} — & w_1^T & — \\ — & w_2^T & — \\ — & w_{d-2}^T & — \\ — & w_{d-1}^T & — \end{bmatrix}.$$

Stepwise Search

Stepwise search is a Bayesian algorithm that tracks a probabilistic estimate of the user's desired configuration by incrementing or decrementing guesses one string at a time to navigate the dictionary. The initialization of stepwise search is identical to that of posterior matching.

Guess selection At $k = 0$, the swarm guess is initialized as $n(0) = \lfloor N_d/2 \rceil$ and $\hat{\boldsymbol{z}}(0) = \boldsymbol{z}_{n(k)}$, where the $\lfloor x \rceil$ operator rounds x to the nearest integer. For $k > 0$, the guess is updated as

$$n(k) = \max(\min(n(k-1) + (2Y_k - 1), N_d), 1) \qquad \hat{\boldsymbol{z}}(k) = \boldsymbol{z}_{n(k)}.$$

Recalling that $Y_k \in \{0, 1\}$, this guessing rule is equivalent to incrementing or decrementing the guessed configuration by one string position in the direction indicated by the received input, while accounting for edge cases (receiving $Y_k = 0$ for $n(k-1) = 1$ maintains $n(k) = 1$, and receiving $Y_k = 1$ for $n(k-1) = N_d$ maintains $n(k) = N_d$). The noisy user input and posterior update stages are identical to those in posterior matching.

Supplementary Figures

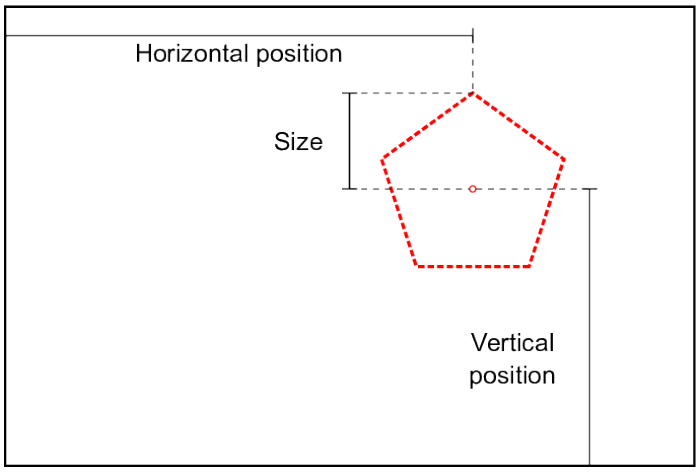


Figure A.1: Polygon dictionary parameters, specified relative to the swarm arena dimensions.

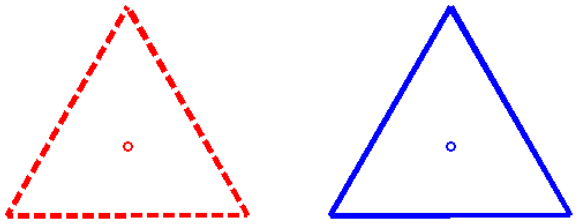
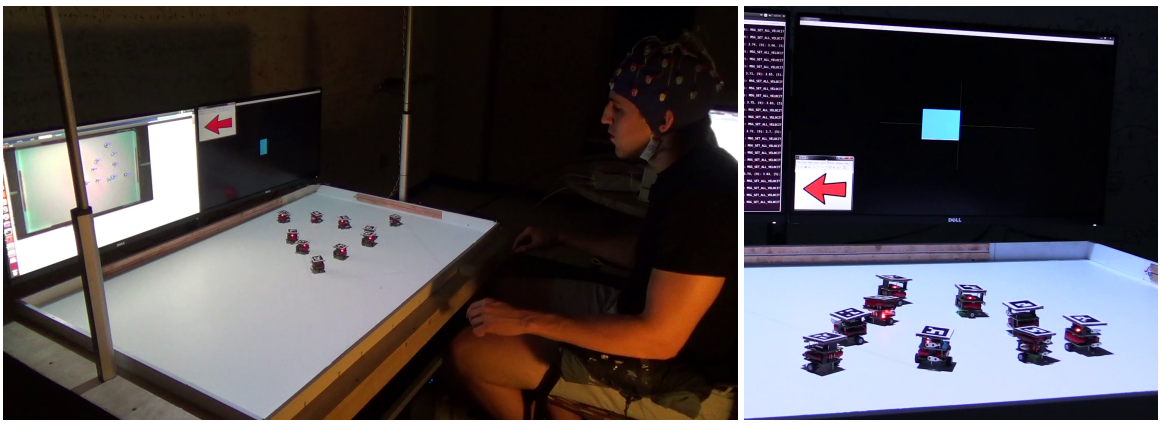


Figure A.2: **Shape query for HIT cheating detection.** This query appeared 6 times in each participant's query set, randomly scattered among regular queries. Participants were not told that these queries were used for cheat detection, although they were told that random selection without a "good faith" effort would be detected (without stating how) and their HIT submission would then be rejected.

(a) (b) (c)

Figure A.3: **Gaussian mixture modeling for swarm density coverage. a,** Gaussian mixture model displayed over a triangle target configuration. Each individual component of the GMM is stylized with a green outline. **b,** Gaussian mixture model density function, without stylization. As described in Diaz-Mercado et al. [90], the robot swarm executes a distributed, low-level algorithm to cover the specified coverage density function, where each robot uses only local information. **c,** Gaussian mixture model density with virtual robots performing swarm coverage.



(a) (b)

Figure A.4: **Physical swarm control.** The BCI user observes the current swarm guess to decide their next motor imagery input according to the rules of posterior matching. When prompted by an audible beep, the user fixates on a blue feedback bar and imagines their intended motor imagery command; the bar responds in real-time to indicate the command being classified, which can aid the user in issuing a reliable command. After each command is issued during a synchronous window of 5 seconds, another beep sounds and an arrow indicates to the user their classified input. While the virtual swarm trials were conducted in a different room with different monitors than those shown here, the software interface used to elicit user motor imagery commands and present feedback was identical to that presented here.

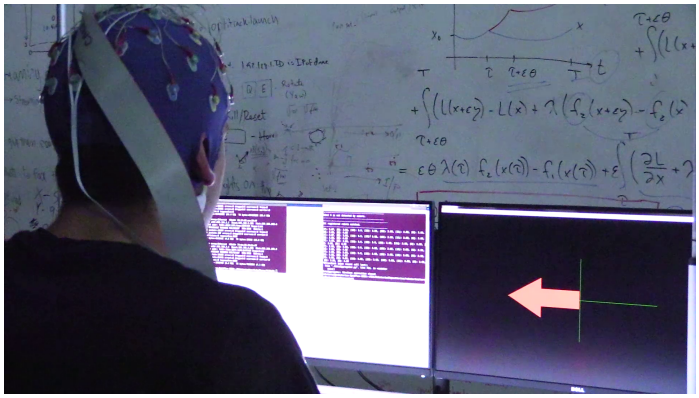


Figure A.5: **Motor imagery training.** During motor imagery classifier training, the user imagines left or right hand motor imagery movements according to a synchronized visual cue.

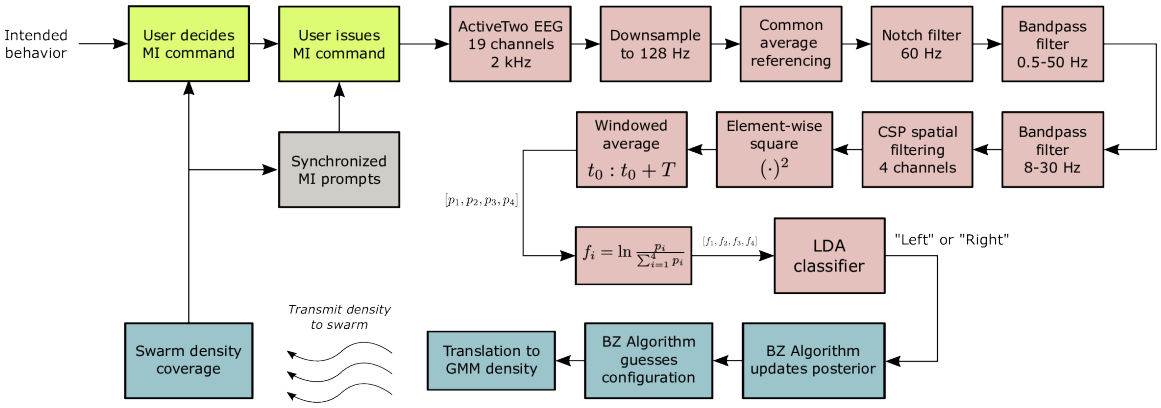


Figure A.6: **Full SCINET feedback system.** The user (green boxes, top left) observes the current swarm configuration along with synchronized motor imagery prompts (see Supplementary Figure A.4) to decide on a motor imagery command for communicating their intended behavior according to the rules of posterior matching. In a signal processing pipeline (pink boxes, top right) beginning with EEG scalp recording and ending with an LDA classifier, the user's motor imagery command is classified from raw scalp measurements. The detected motor imagery command is input into the BZ algorithm, which updates the configuration guess in the dictionary. This configuration is translated to a Gaussian mixture model density (see Supplementary Figure A.3) and transmitted to the swarm for distributed density coverage.

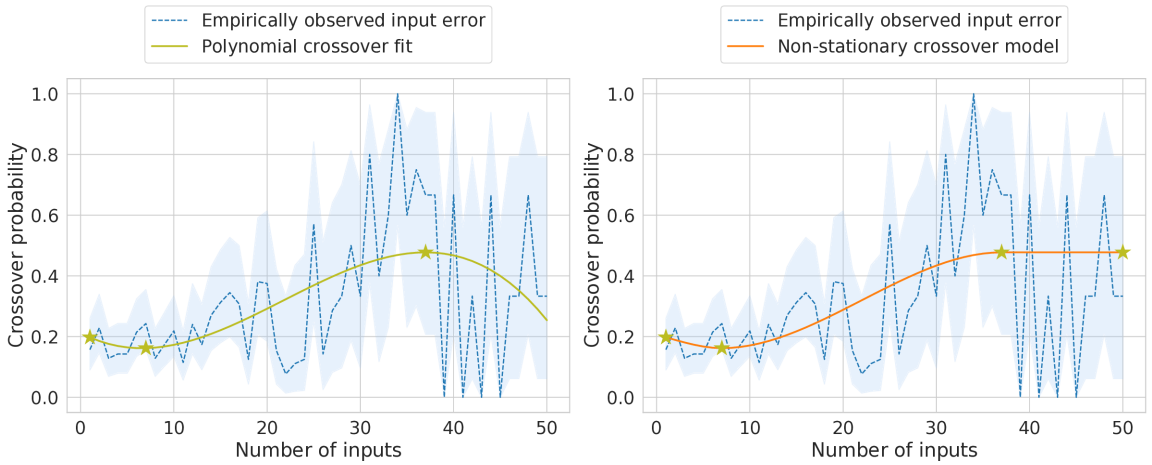


Figure A.7: **Modeling a non-stationary input profile from empirical crossover data. a,** Least-squares cubic fit to empirical crossover data. The model is highlighted with stars at the first input value, the minimum value, and the maximum value. **b,** A piecewise cubic Hermite interpolating polynomial (PCHIP) was fit to the first, minimum, and maximum value points from **a**. We clamped the maximum value to remain fixed until the maximum of 50 inputs, so that the resulting PCHIP error model is non-decreasing. This behavior models the fact that the BCI user may experience fatigue as the number of inputs increases, possibly resulting in non-improving input error statistics after issuing many inputs.

A.3.2 Supplementary Discussion

Online User Study Critical Character Analysis

In Supplementary Figure A.8, we analyze dictionary sorting proficiency for each individual participant in the online user study. Figure A.8a plots each individual participant's accuracy in sorting shape pairs, binned by critical character; for each subject these values are connected by straight lines, for visualization purposes. Each participant's piecewise linear curve is colored by their *overall* sorting accuracy across all 144 (not including test queries) shape pairs, as indicated by the color bar. Generally speaking, the lowest performing subjects (dark colors) performed poorly across all critical character comparisons. Interestingly, several subjects performed well for horizontal position, dropped in performance for vertical position, and increased in performance for number of sides and size comparisons. Several subjects sorted vertical position below 50% accuracy, implying that they sorted this character consistently, but in the reversed alphabet order of precedence.

To more formally analyze the trends over each individual participant, for each participant we performed a least-squares linear fit to their accuracies across critical characters, with categorical values converted to regressors as 0 (horizontal position), 1 (vertical position), 2 (number of sides), and 3 (size). In Figure A.8b, we present a scatter plot of the regression slope of each participant's linear model, plotted against each model's intercept at the horizontal position. As in Figure A.8a, participants are colored coded by their overall sorting accuracy. Only two participants both sorted early string characters with high accuracy (large horizontal position intercept) and decreased in performance for deeper characters (negative regression slope). Otherwise, participants mostly performed accurately across all characters (high horizontal position intercept, flat slope), or performed moderately for early characters and increased in accuracy, with a positive regression slope. Only a few participants performed poorly for early characters and continued to perform poorly for deeper characters.

In Figures A.8c and A.8d, we plot a histogram and empirical cumulative distribution function of regression slopes across all participants. As can be observed from Figure A.8d, only approximately 30% of participants have negative regression slopes, and only 10% of participants have regression slopes of 1% accuracy decrease or worse per additional character. These results collectively suggest that overall, the performance of individual participants did not decrease noticeably as character depth increased. This suggests that as the number of characters in each heterogeneous dictionary string is increased, users are still able to both identify and make accurate comparisons with respect to each critical character.

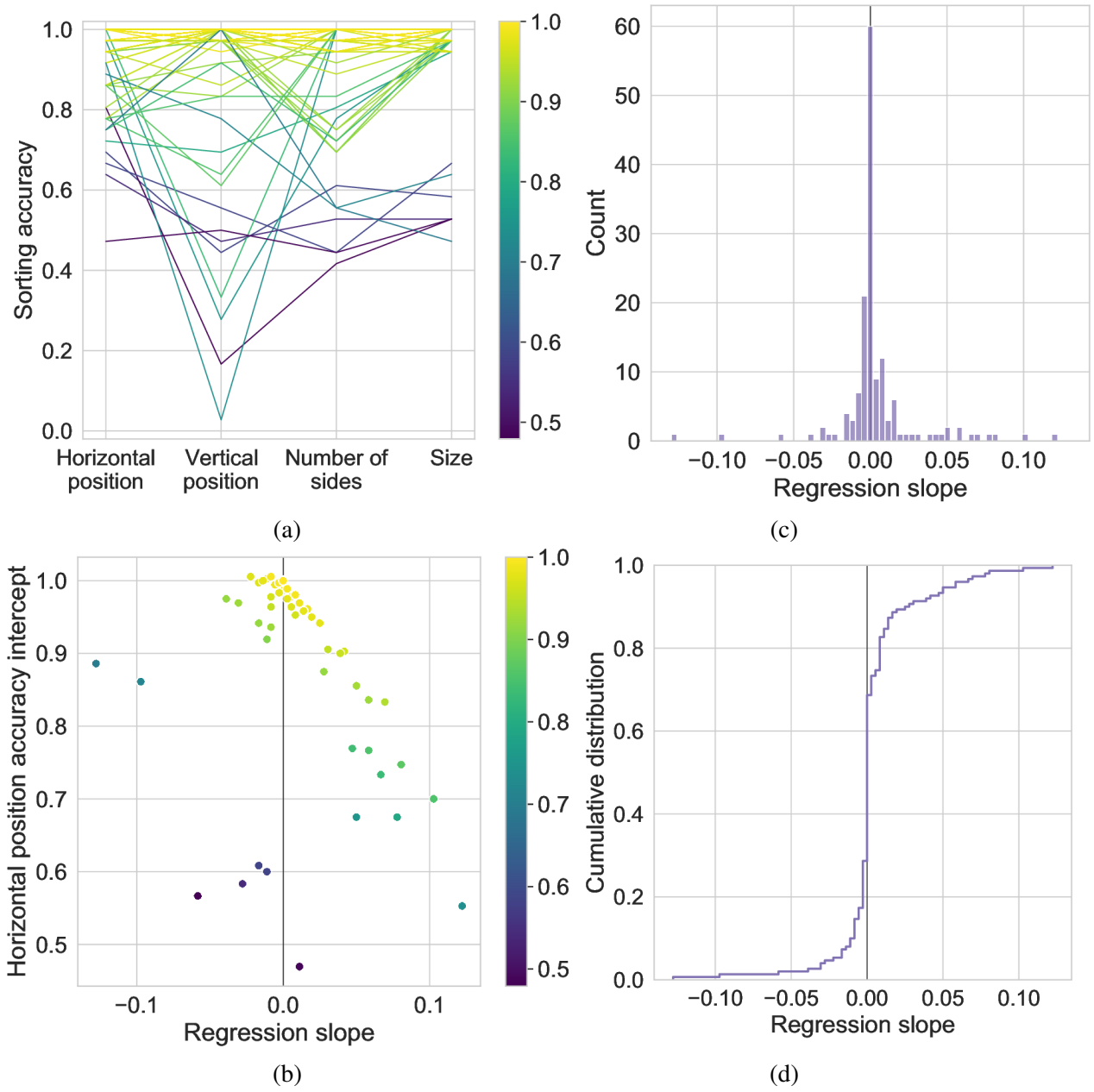
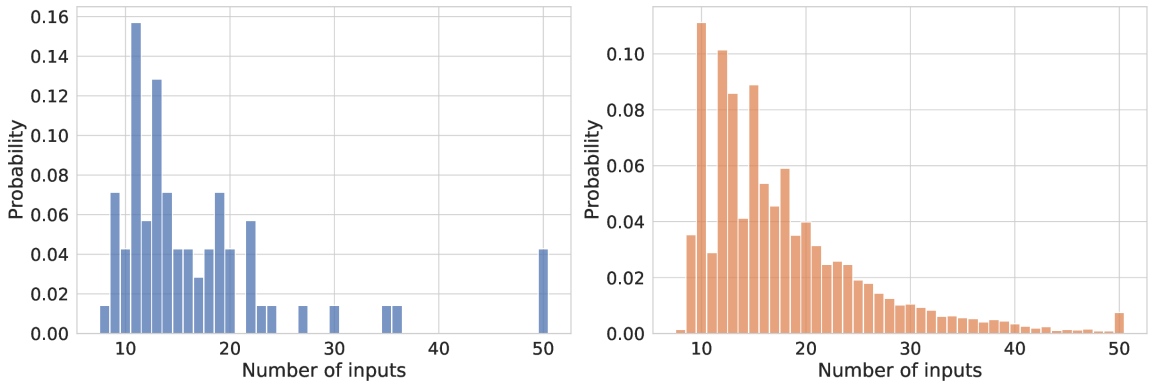


Figure A.8: **Trend line analysis of individual participant performance over critical character comparisons of increasing depth.** **a,** Each individual participant's sorting accuracy is plotted as a connected line over accuracies calculated with respect to each critical character comparison. Each line is colored by the participant's overall sorting accuracy across all queries. **b,** After fitting a linear least squares model to each participant's piecewise curve in **a**, we plot the slope and intercept at the horizontal position for each participant's linear regression model. **c,** Histogram of individual participant regression slopes. **d** Empirical cumulative distribution plot of individual participant regression slopes. Only approximately 30% of participants have negative regression slopes, and only 10% of participants have slopes of 1% accuracy decrease or worse per additional character.

Virtual Swarm Evaluation

Figure A.9 compares the histograms of the number of inputs required for convergence in each experimental and simulated trial. These histograms generally agree in shape, with right skewed distributions and peaks at the maximum number of inputs, reflecting that several trials "timed out" before convergence. This overall agreement corroborates the simulation's realistic modeling of the overall system.



(a) (b)

Figure A.9: Histogram of number of inputs until convergence for virtual swarm control (**a**) and simulated (**b**) trials.

Figure A.10 plots the number of recorded samples at every number of inputs, across all 70 virtual swarm trials. Since no trials cross the convergence threshold until after at least 8 inputs, 70 input samples were recorded at every number of inputs at or below this point. As trials begin to converge after 8 inputs, fewer recorded samples are available for larger number of inputs due to trials converging and halting input recording. Due to this decreasing sample size at larger numbers of inputs, the empirical EEG crossover probability in Figure 3.4c is most accurately estimated at lower numbers of inputs. As the number of inputs increases, crossover probability is less accurately estimated due to smaller sample sizes, reflected in the larger error bars in the crossover probability estimate.

In Figure A.11, we analyze the data in Figure 3.4d in terms of absolute deviation rather than error-free configuration accuracy. As in Figure 3.4d, virtual swarm and simulated trials

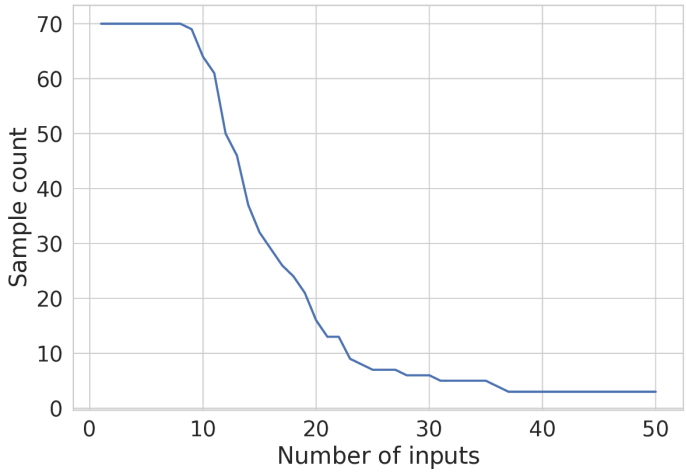


Figure A.10: **Number of samples at each number of issued inputs, aggregated over all virtual swarm trials.** The number of samples observed at each number of inputs decreases as trials converge to selected configurations. This results in larger error bars in Figure 3.4c and higher late trial variability in Supplementary Figures A.12 and A.13.

were separated into the same "Short" (between 1 and 12 inputs until convergence, inclusive), "Medium" (between 13 and 18 inputs until convergence, inclusive), and "Long" (between 19 and 50 inputs until convergence, inclusive) trial bins. Then, we analyzed the absolute deviation trajectories (see Section A.2 for a description of absolute deviation) within each bin by comparing a trial's target configuration to the posterior median guessed by the swarm after every input; this differs from the absolute deviation calculation in Figure 3.5, which instead calculates absolute deviation with respect to an instantaneous MAP configuration estimate. Figures A.11a to A.11c plot absolute deviations for both virtual swarm and simulated trials within Short, Medium, and Long bins, respectively. Figure A.11d plots absolute deviation over all trials. In each figure, vertical red lines visually indicate the span of each bin range, depicting the range of inputs in which all trials within the bin converged. The virtual swarm control and simulation trials have absolute deviations that mostly agree in the first two bins, with increased differences between empirical and simulated trials in the Long bin. Both empirical and simulated results in the Long bin experience increasing absolute deviations, due to the fact that input errors increase toward chance for larger numbers of inputs (see Figure 3.4c). Regardless, the simulated system in its entirety matches the behavior of the

experimental trials, indicating that the simulator used here can reliably account for realistic experimental factors such as user fatigue.

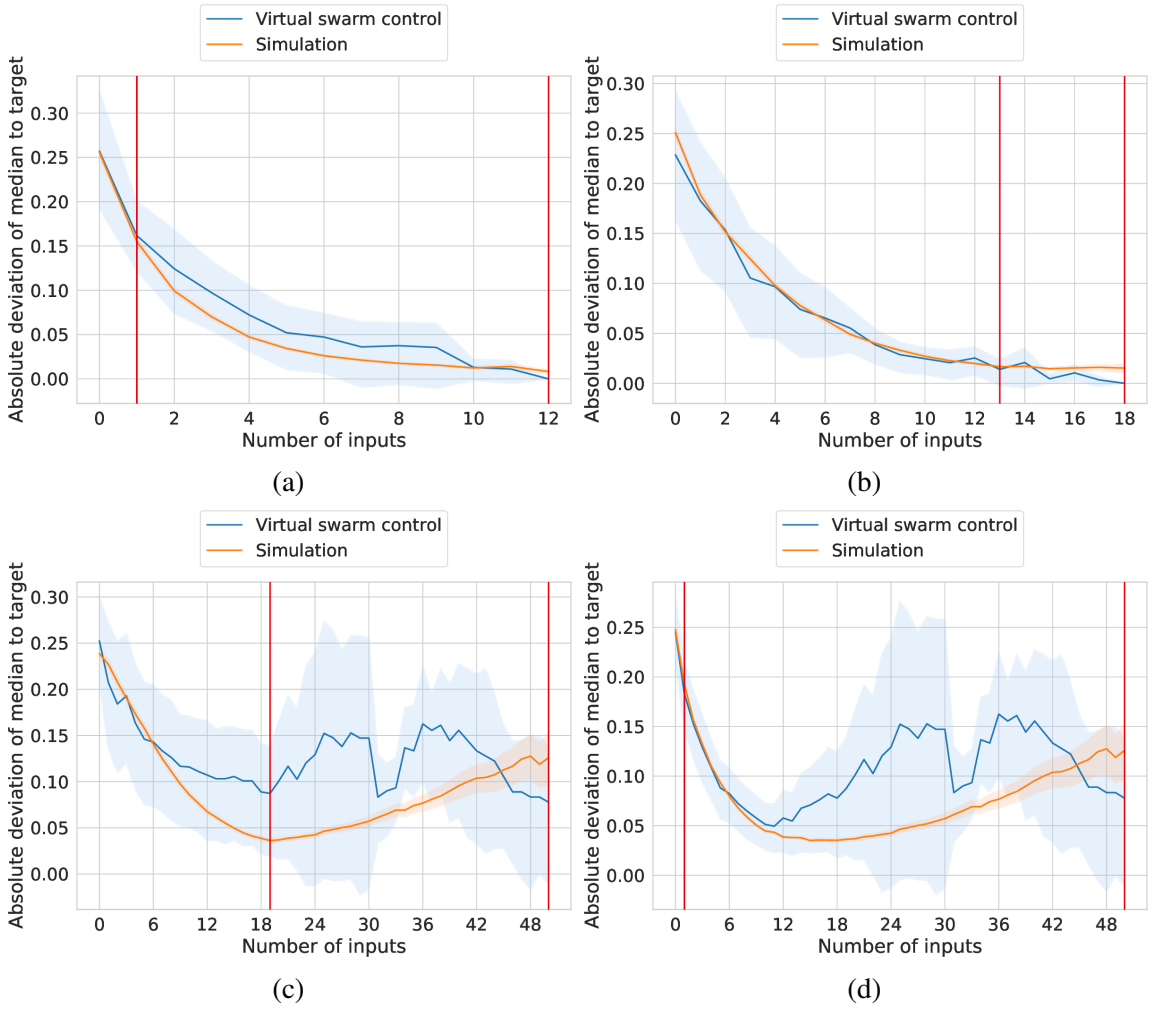


Figure A.11: Absolute deviation between guessed swarm configuration after each number of inputs (guessed as posterior median) in comparison to target configuration, for both virtual swarm control and simulated trials. The crossover model from Figure 3.4c was used to generate non-stationary errors in all 10,000 simulation trials. Each pane depicts a subset of trials binned by number of inputs until swarm convergence. The bin range for each pane is depicted through two vertical lines at the bin edges (inclusive). Each trial set, for both virtual swarm control and simulated trials, is plotted as mean absolute deviation with error bars depicting 95% bootstrap confidence intervals over 10,000 samples (separate resampling for every number of inputs). **a,** Trials converging between 1 and 12 inputs, inclusive. **b,** Trials converging between 13 and 18 inputs, inclusive. **c,** Trials converging between 19 and 50 inputs, inclusive. **d,** All trials. In each pane, values at 0 inputs indicate absolute error at initialization, before a trial begins.

Signal feature analysis In Figure 3.4c, the empirically observed crossover probability degrades in quality (approached chance) at the number of issued inputs increases. As demonstrated in Figure 3.4d, this behavior can be accounted for in a posterior matching simulator that assumes a fixed crossover probability while errors are generated according to a non-stationary input error profile. Below, we further analyze the empirically observed input degradation. In particular, we analyze the behavior of the LDA classifier component of our motor imagery detection pipeline, and measure signal quality by observing the changing classifier confidence over time.

Let $f \in \mathbb{R}^4$ denote the EEG feature vector for a given input (see Figure A.6), and let $\mu \in \mathbb{R}^4$ and $\tau \in \mathbb{R}$ denote the hyperplane weights and offset respectively of the trained LDA classifier [209]. Let $Y \in \{0, 1\}$ denote a classification result of left-hand ($Y = 0$) or right-hand ($Y = 1$) motor imagery detection, determined by the sign of the distance to the classifier hyperplane, i.e., $Y = \text{sign}(\mu^T f - \tau)$. In LDA, a standard result [209] is that the log-ratio of the class posterior distribution is given by

$$\log \frac{P(Y = 1 \mid f)}{P(Y = 0 \mid f)} = \mu^T f - \tau.$$

Denote $X \in \{0, 1\}$ to be the ground truth (i.e., correct) motor imagery input that the user should issue, according to the rules of posterior matching. One way to measure the quality of the classifier's decision is to evaluate the ratio of the probability that it detects the ground truth input correctly (i.e., $P(Y = X \mid f)$) over the probability that it detects the ground truth incorrectly (i.e., $P(Y \neq X \mid f)$) It is easy to show that the log of this ratio

takes a convenient form:

$$\begin{aligned}
\log \frac{P(Y = X \mid f)}{P(Y \neq X \mid f)} &= X \log \frac{P(Y = 1 \mid f)}{P(Y = 0 \mid f)} + (1 - X) \log \frac{P(Y = 0 \mid f)}{P(Y = 1 \mid f)} \\
&= X \log \frac{P(Y = 1 \mid f)}{P(Y = 0 \mid f)} + (X - 1) \log \frac{P(Y = 1 \mid f)}{P(Y = 0 \mid f)} \\
&= (2X - 1) \log \frac{P(Y = 1 \mid f)}{P(Y = 0 \mid f)} \\
&= (2X - 1)(\mu^T f - \tau). \qquad \text{(A.7)}
\end{aligned}$$

When the classifier is more confident in the correct class (which is unavailable to the classifier at classification time, but is available in post hoc analysis) than the incorrect class, then this log-ratio will be positive. Conversely, if the classifier is confident in the *incorrect* decision, then this log-ratio will be negative. If the classifier is "unsure" in its decision, then this log-ratio will be close to 0 since the classifier assigns equal probability to both the correct and incorrect decision. These scenarios can also be interpreted geometrically by considering the equivalent log-ratio form in eq. (A.7): the log-ratio of probability of a correct to an incorrect decision corresponds to the signed distance from the feature vector to the hyperplane, where the sign is determined by whether the classifier is correct or incorrect in its decision. Decisions that are confident and correct will have a positive signed distance, ambiguous decisions a signed distance of 0, and confident but incorrect decisions will have a negative signed distance.

Supplementary Figure A.12 depicts the distribution of this log-ratio (or signed classifier distance) over increasing numbers of inputs, on the same input data evaluated in Figure 3.4c. After each number of inputs, we calculate the log ratio in eq. (A.7) with respect to the corresponding EEG feature vector. Initially, the classifier is confident in correct decisions, and this confidence decreases gradually towards zero as more inputs are issued. This metric is a direct measure of feature degradation, due to the direct correspondence between the log-ratio of classifier probabilities and the distance from each feature vector to the decision

boundary. In other words, as the number of inputs increases, the processed EEG signal vectors are on average closer to the decision boundary, indicating that features are no longer being well separated according to the classifier geometry established during motor imagery training.

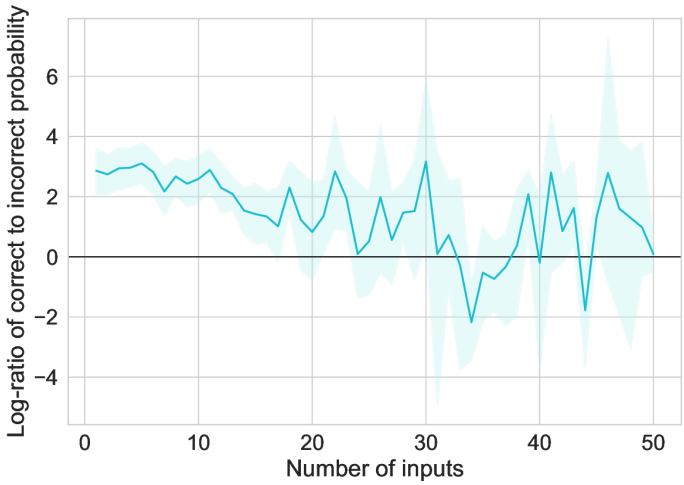
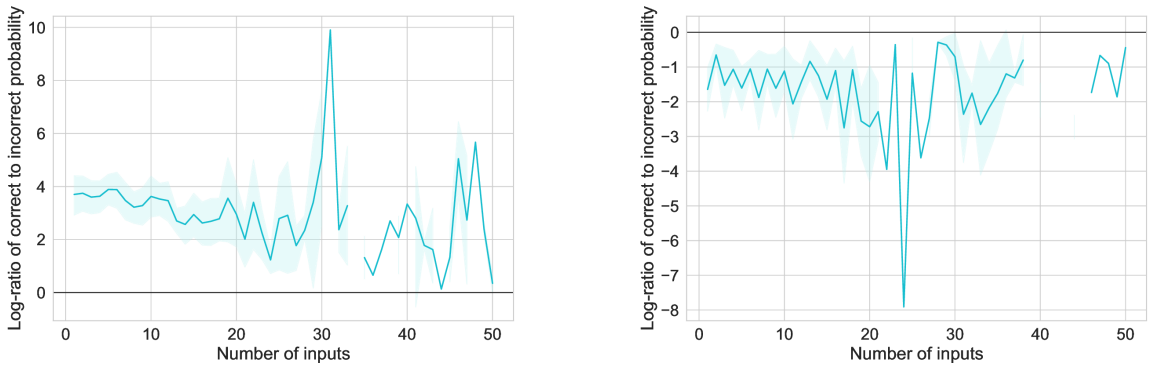


Figure A.12: Log-ratio of classifier probability assigned to the correct input over the probability assigned to the incorrect input, plotted against the number of inputs issued in a trial. Results are aggregated over all virtual swarm trials and plotted as mean log-ratio with error bars depicting 95% bootstrap confidence intervals over 10,000 samples (separate resampling for every number of inputs). When taken as a measure of classifier confidence, the log-ratio's steady decline indicates decreasing classifier confidence in its decisions, as the classifier's probability assignment to the correct input approaches 0.5. Equivalently, this log-ratio measures the signed distance of each feature vector to the classifier hyperplane, signed such that positive distances indicate correct classification.

For completeness, we also perform the same analysis when grouping inputs by correct or incorrect classification (Supplementary Figure A.13). In Figure A.13a, we evaluate the log-ratio of classifier probabilities only on inputs that were classified correctly. Although the classifier is making correct decisions on this data group, it is clear from the gradual log-ratio decline that the classifier's correct decisions are made with less confidence as the number of inputs increases to approximately 25 inputs. This corresponds directly to the increasing crossover probability observed in Figure 3.4c. Conversely, in Figure A.13b, we evaluate log-ratio for only *incorrectly* classified inputs. In this case, the log probability ratio becomes more negative for increasing numbers of inputs up to approximately 25 inputs,



(a) Inputs classified correctly

(b) Inputs classified incorrectly

Figure A.13: Log-ratio of classifier probability assigned to the correct input over the probability assigned to the incorrect input, grouped by inputs that were classified correctly (**a**) or incorrectly (**b**). Results are aggregated over all recorded inputs in each group and plotted as mean log-ratio with error bars depicting 95% bootstrap confidence intervals over 10,000 samples (separate resampling for every number of inputs). **a,** When only analyzing inputs that were detected correctly by the classifier, we still observe a decline in confidence. **b,** We also analyze classifier confidence when grouped over inputs that were detected incorrectly. Both **a** and **b** have missing data, since at certain numbers of issued inputs all samples were either detected correctly or incorrectly. Error bars are omitted at numbers of inputs with only a single sample.

indicating that the classifier becomes more confident in its incorrect decisions. Interestingly, this behavior would indicate a degree of class reversal in the statistics of the extracted EEG features. Both Figures A.13a and A.13b have missing data, since at certain numbers of inputs all signals were detected either correctly or incorrectly.

Overall, the analysis in this section indicates that during test time (i.e., when issuing inputs for swarm control, rather than for motor imagery training), the EEG feature statistics shift after increasing numbers of inputs in a way that no longer aligns with or perhaps experiences a reversal with respect to the linear classifier trained before each session. While the focus of this work is high-complexity control *despite* the presence of such noisy inputs, these results suggest that the increasing crossover probability observed in Figure 3.4c could possibly be mitigated through a calibration step during the effector control period. During such a step, effector control could be briefly halted to collect a small number of additional supervised motor imagery inputs using the same protocol during training, such that the CSP

and LDA parameters can be readjusted. This calibration step could either be set ahead of time (e.g., after 25 inputs), or could be automatically triggered when the LDA feature distance (i.e., $|\mu^T f - \tau|$) falls below a prespecified threshold across multiple inputs, indicating a possible mismatch between the current EEG statistics and the original classification pipeline learned during session training. Finally, the analysis in this section does not take into account the possibility of user error rather than classifier error; it is possible that an "incorrect" trial as analyzed above is in fact the result of correct motor imagery classification, issued with an incorrect user input. To fully disambiguate these sources of error, additional data collection is necessary where after issuing each motor imagery input the user, via another means such as a keypress or speech (if they are able to do so), indicates which hand movement they intended to convey.

Extended Evaluation of Generalized Dictionary Simulations

In Figure A.14, we perform the same analysis as in Figure 3.5 except with a simulated alphabet size of $b = 5$ (see Section A.2 for simulation details) rather than $b = 3$. We refer to the case of $b = 5$ as the "conservative" degrees of freedom estimate, recalling that we refer to $b = 3$ as the "standard" estimate. Since the posterior matching and stepwise search simulations track mathematical partitions of the unit interval and operate directly on the total dictionary order of strings rather than distinguishing between individual alphabet orders (which is performed by the human user), the only parameter affected by alphabet size relevant for simulation is the overall dictionary size N_d. In other words, distinct alphabet sizes b_1 and b_2 and numbers of degrees of freedom r_1 and r_2 that happen to produce the same dictionary size $N_d = b_1^{r_1} = b_2^{r_2}$ will yield the same simulation results, since both scenarios map to the same partitions of the unit interval. Because of this, Figure A.14 with $b = 5$ can be seen as mathematically equivalent to Figure 3.5 with $b = 3$, except with different dictionary sizes for each number of degrees of freedom.

What does differ between these figures is the translation of each dictionary size to

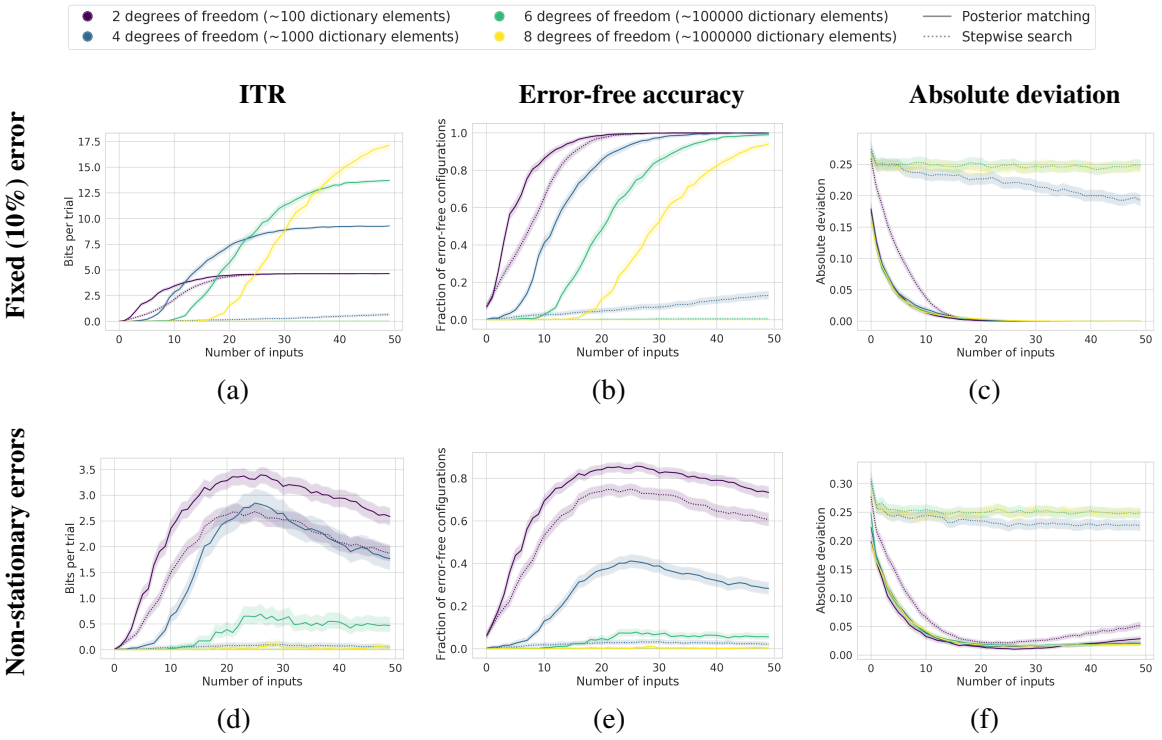


Figure A.14: Performance as a function of number of inputs and dictionary size across both fixed and non-stationary input errors, with conservative degrees of freedom estimates. The same performance metrics (ITR, error-free accuracy, absolute deviation) with corresponding error bars are displayed here as in the standard degrees of freedom estimate in Figure 3.5.

estimated degrees of freedom, since a fixed number of degrees of freedom for a larger alphabet size corresponds to a larger dictionary than for smaller alphabets (see Section A.2 for details on the relationship between dictionary size and alphabet size). For this reason, Figure A.14 serves as a more conservative estimate of the tradeoffs involved in controlling more degrees of freedom. In particular, since each number of degrees of freedom corresponds to a larger dictionary than in Figure 3.5, more inputs are needed to control a conservatively estimated number of degrees of freedom than would be required for the standard estimate. While this additional cost in number of inputs is apparent in Figure A.14 in comparison to Figure 3.5, the same overall trend holds that posterior matching significantly outperforms stepwise search across all metrics, and that with enough refinements posterior matching can obtain high configuration accuracies at high estimated degrees of freedom.

In Figures A.15 and A.16, we evaluate additional simulation performance metrics based on alphabet-wise convergence for both standard and conservative degrees of freedom estimates. These metrics evaluate the performance of each algorithm in driving each *individual* character to its correct target rather than measuring convergence of the entire string at once, and provide another perspective on alphabet-wise convergence not captured by the latter.

Expanding on notation from Section A.3.1, let $\widehat{\boldsymbol{z}} = \{\widehat{\sigma}^i\}_{i=1}^r$ denote a configuration estimate with ith character $\widehat{\sigma}^i \in 1, 2, \ldots b$, and let $\boldsymbol{z}_t = \{\sigma_t^i\}_{i=1}^r$ denote the target configuration. Both configurations have r degrees of freedom each with alphabets of size b. Our first alphabet performance metric is "alphabet accuracy," which for a single configuration estimate measures the fraction its characters that equal the corresponding characters in the target configuration:

$$\mathrm{AlphAcc}(\widehat{\boldsymbol{z}}, \boldsymbol{z}_t) = \frac{1}{r} \sum_{i=1}^{r} \delta(\widehat{\sigma}^i, \sigma_t^i),$$

where $\delta(x, y) = 1$ if $x = y$, and 0 otherwise. Figures A.15a, A.15d, A.16a and A.16d plot alphabet accuracy averaged over multiple trials, calculated with respect to the instantaneous MAP configuration estimate after every number of inputs.

Next, we calculate "alphabet deviation," which in a similar spirit to absolute deviation (or dictionary distance) calculates the sum of absolute deviations within each *alphabet*. As in dictionary distance, the alphabet deviation measures the rate of convergence of each alphabet character to its respective target character, rather than measuring a binary notion of correct or incorrect convergence. Alphabet deviation is calculated as

$$\mathrm{AlphDev}(\widehat{\boldsymbol{z}}, \boldsymbol{z}_t) = \sum_{i=1}^{r} \left| \frac{\widehat{\sigma}^i - \sigma_t^i}{b} \right|.$$

We normalize the absolute deviation within each alphabet by the alphabet size b such that the resulting metric has a range between 0 and r that does not depend on alphabet size. Figures A.15b, A.15e, A.16b and A.16e plot alphabet deviation averaged over multiple

trials, calculated with respect to the instantaneous MAP configuration estimate after every number of inputs.

For completeness, we also calculate a normalized version of alphabet deviation which we call "normalized alphabet deviation." Normalized alphabet deviation is simply equal to alphabet deviation normalized by the number of degrees of freedom. This yields an alphabet deviation metric that scales between 0 and 1 for any number of degrees of freedom r, and allows for a more equalized comparison between different degrees of freedom.

$$\text{NormAlphDev}(\widehat{\boldsymbol{z}}, \boldsymbol{z}_t) = \frac{1}{r} \sum_{i=1}^{r} \left| \frac{\widehat{\sigma}^i - \sigma_t^i}{b} \right|.$$

Figures A.15c, A.15f, A.16c and A.16f plot normalized alphabet deviation averaged over multiple trials, calculated with respect to the instantaneous MAP configuration estimate after every number of inputs.

Figure A.15 depicts these alphabet-wise metrics for the standard degrees of freedom estimate across both fixed and non-stationary errors. The most direct point of comparison for alphabet accuracy in Figures A.15a and A.15d is error-free accuracy with respect to the entire string, as depicted in Figures 3.5b and 3.5e. In the fixed error case, both string-wise and alphabet-wise metrics capture similar behavior. In the case of non-stationary errors, measuring alphabet accuracy appears to penalize larger numbers of degrees of freedom less harshly than accuracy with respect to the entire string. Intuitively, alphabet accuracy accounts for the fact that early characters may have converged successfully to the target while later characters are still being refined, which is a subtlety that is ignored when assessing error-free accuracy at the string level.

Alphabet deviation and normalized alphabet deviation depict similar trends to one another; we focus on normalized alphabet deviation due to its attractive normalization between different degrees of freedom. When comparing normalized alphabet deviation in Figures A.15c and A.15f to string-wise dictionary distance in Figures 3.5c and 3.5f, we can

observe characteristics of normalized alphabet deviation not captured by dictionary distance. In particular, since posterior matching operates through bisections of the entire dictionary, it is able to quickly refine a configuration's equivalent point on the unit interval to the correct numerical neighborhood, regardless of number of degrees of freedom (see Section A.3.1). This is reflected in dictionary distance by the clustering of all degrees of freedom at low deviation values in Figures 3.5c and 3.5f. Unlike dictionary distance, normalized absolute deviation accounts for individual convergence within each alphabet, which may still be large for later characters even if posterior matching has converged well when measured by overall dictionary distance. As can be observed in Figures A.15c and A.15f, larger numbers of degrees of freedom require more inputs to refine character selection within a larger number of alphabets. Similar trends as these can be observed when using conservative degrees of freedom estimates in Figure A.16 and comparing to the corresponding string-wise metrics in Figure A.14.

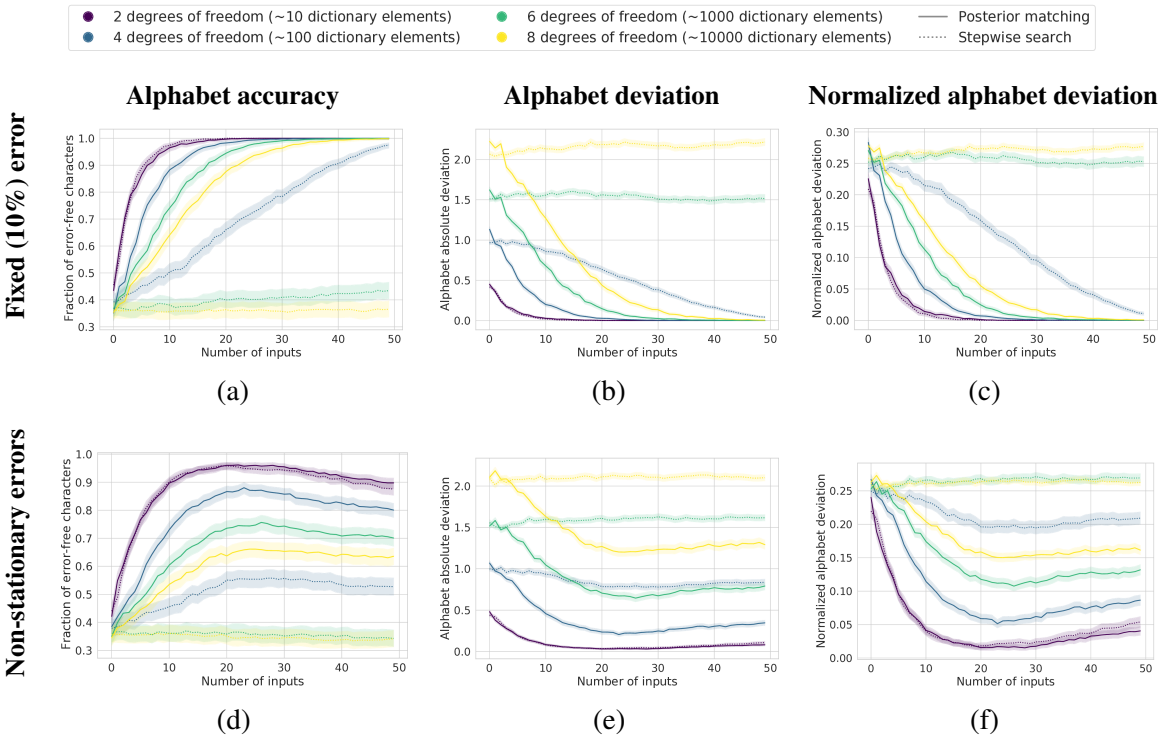


Figure A.15: Alphabet-wise performance metrics as a function of number of inputs and dictionary size across both fixed and non-stationary input errors, with standard degrees of freedom estimates. Alphabet accuracy (**a,d**) measures the fraction of error-free guessed characters, plotted at its mean value with a 95% Wilson confidence interval. Alphabet deviation (**b,e**) measures the sum of absolute deviations *within* each alphabet of a configuration guess, plotted at its mean value with error bars depicting 95% bootstrap confidence intervals over 10,000 samples (separate resampling for every number of inputs). Normalized alphabet deviation (**c,f**) calculates alphabet deviation, but normalizes each value by the number of degrees of freedom, plotted at its mean value with error bars depicting 95% bootstrap confidence intervals over 10,000 samples (separate resampling for every number of inputs).

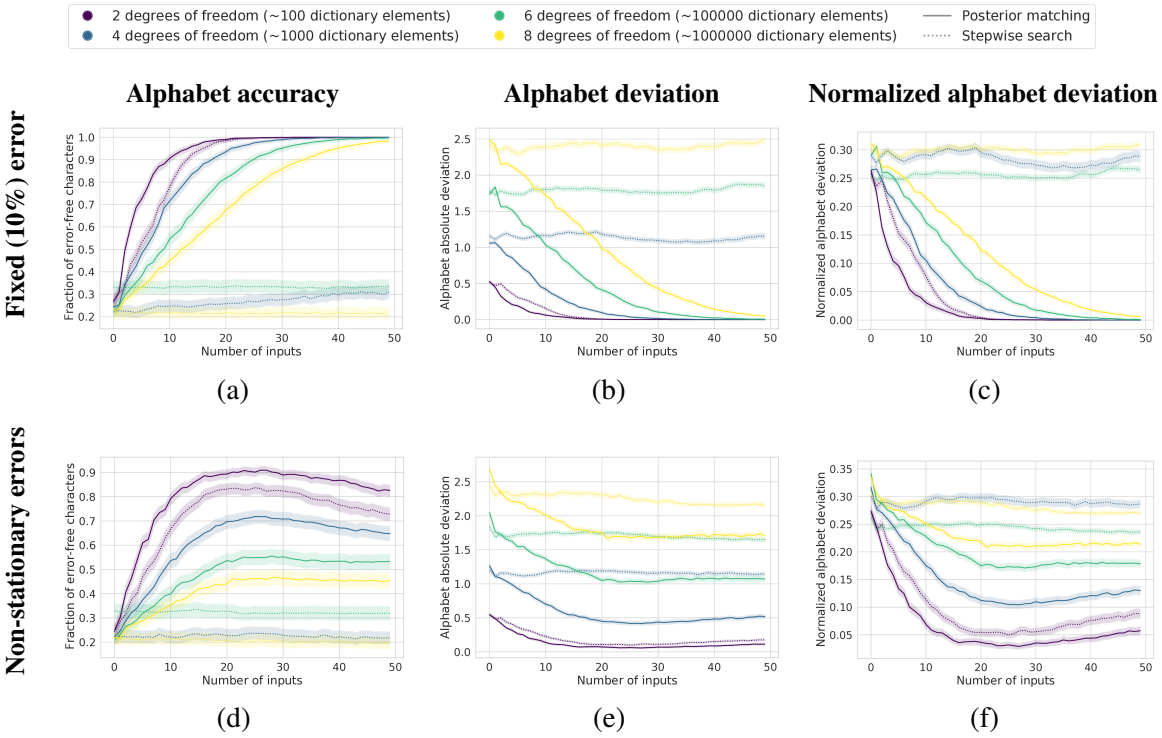


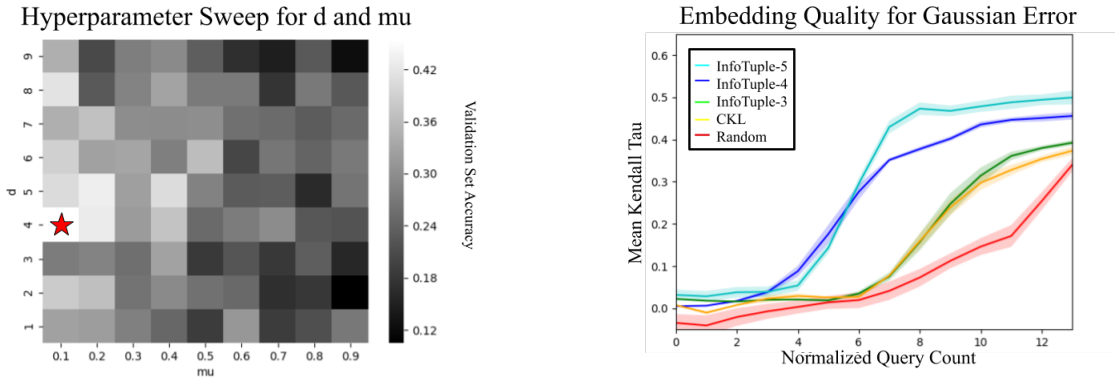
Figure A.16: Alphabet-wise performance metrics as a function of number of inputs and dictionary size across both fixed and non-stationary input errors, with conservative degrees of freedom estimates. All plotted metrics and error bars are otherwise identical to those in Figure A.15.

APPENDIX B

EXPERIMENTAL DETAILS IN TUPLEWISE SIMILARITY LEARNING

B.1 Experimental Details

For each of the human-subject experiments, μ was set to 0.1 and d was set to 4 per the hyperparameter search shown in Figure B.1a. The validation set for this search was an additional 500 heldout triplets from the Food10k dataset. In the synthetic experiments provided, μ was set to 0.5 and d was set to 2 to match the dimensionality of the generating distribution. The stochastic oracle had a high noise level, inverting 33% of tuple responses. Higher tuple sizes were strongly correlated with both higher performance and higher robustness to error (even when normalized by the effective number of pairwise queries), indicating performance gains for InfoTuple that are not simply due to increasing tuple sizes. A heuristic was used to pick a number of samples for the Monte Carlo estimation of the mutual information, with $\frac{N}{10}$ samples being used in practice.



(a) (b)

Figure B.1: Supplementary experiments for tuplewise similarity learning. (a) Hyperparameter sweep for Food10k dataset. Experimental values of $d = 4$ and $\mu = 0.1$ were found to be the most effective on a held-out validation set of triplets. (b) Synthetic experiment results using an oracle with Gaussian noise. Results were broadly consistent with those of the Plackett-Luce oracle in spite of the mismatch between the oracle noise and the embedding.

Figure 5.3c in Chapter 5 shows empirical performance for query selection algorithms on predicting labels from held out triplet queries in the Mechanical Turk dataset described. Experimental horizons for human subject experiments were chosen based on estimates of the initial steps of convergence and had to be limited due to high experimental costs. Turk subjects were presented with queries in batches of 25, with one repeated tuple across the batch as a test for validity. If the repeat query was not answered the same way by the user both times it was asked the batch was discarded. Order effects were controlled for by shuffling queries prior to presenting them to users for labeling, ensuring that any queries presented to multiple users would appear in different orders and that the test queries would also appear differently each time.

B.2 Oracle Details

Two different models of oracle noise were used in our synthetic experiments, Plackett-Luce noise and Gaussian noise. These models were chosen to be different from the one we use to estimate mutual information in order to demonstrate the robustness of our method. In Chapter 5 we describe the selection process used by the Plackett-Luce oracle noise, which works by assigning latent scores to objects on the basis of their distances in some synthetic "ground truth" embedding space. The Gaussian noise model, instead of applying noise directly at the level of the ranking responses, applies noise at the level of the oracle's representation of the "ground truth" embedding by adding Gaussian noise to the coordinates of each point drawn from the "ground truth" embedding before imputing a ranking from distances in the oracle's noisy interpretation of the space. For the Plackett-Luce error model results shown in Chapter 5, 33% of individual rankings were inverted.

B.3 Computational Complexity

The computational complexity of the embedding calculation is that of a typical MDS algorithm- for any $M \in \mathbb{R}^{d \times N}$ an approximate solution can be found in $O(N)$ for $d < N$

[211]. Our case has an N far greater than d while still being of manageable size, allowing for a fast linear-time approximation.

With respect to the entropy calculation itself, the inner loop computing the mutual information from a given tuple is computable in $O(N_f k^2)$. However, the computational complexity for a given algorithm iteration is dominated by the $O(\omega\binom{N}{k-1})$ cost of generating and iterating over large pools of candidate tuples, meaning that the run-time is heavily dependent on the choice of the sampling rate ω and distance sample size N_f, and the question of how to efficiently estimate similar mutual information quantities without the use of Monte Carlo methods remains open.

APPENDIX C

PROOFS AND ADDITIONAL DETAILS IN PAIRWISE SEARCH

First, we begin with an additional lemma:

Lemma C.0.1. *Let X_i be a marginal distribution of W. The density of X_i is then*

$$p_{X_i|y^i}(x) = \frac{1}{\sigma_i} p_{Z_i}\left(\frac{X_i - \mathbb{E}[X_i|y^i]}{\sigma_i}\right) \leq \frac{1}{\sigma_i},$$

where $\sigma_i = \sqrt{\mathbb{E}[(X_i - \mathbb{E}[X_i|y^i])^2|y^i]}$ and $Z_i = \frac{X_i - \mathbb{E}[X_i|y^i]}{\sigma_i}$.

Proof. Since X_i is a marginal of a log-concave distribution, X_i is also log-concave. Furthermore, Z_i is a zero-mean, unit-variance (i.e., isotropic) log-concave random variable with density $p_{Z_i}(z)$. Then Lemma C.0.1 follows because one-dimensional isotropic log-concave densities are upper bounded by one [24]. □

A direct consequence of Lemma C.0.1 is that for any $a > 0$,

$$\begin{aligned} P(|X_i| < a \,|\, y^i) &= \int_{-a}^{a} p_{X_i|y^i}(x)dx \\ &\leq \frac{1}{\sigma_i}\int_{-a}^{a} dx \leq \frac{2a}{\sigma_i} \end{aligned}$$

implying that

$$P(|X_i| \geq a \,|\, y^i) \geq 1 - \frac{2a}{\sigma_i}. \tag{C.1}$$

C.1 Proof of Lemma 6.3.1

Proof. Letting Σ_W denote the $d \times d$ covariance matrix of random vector $W \in \mathbb{R}^d$, from Theorem 8.6.5 in [27], we have the upper bound

$$h(W) \leq \frac{1}{2} \log_2((2\pi e)^d |\Sigma_W|). \tag{C.2}$$

Now assume the distribution P_W of W is log-concave, let $W_1, W_2 \sim P_W$ be i.i.d. and let $\widetilde{W} := W_1 - W_2$. Let $p_{\widetilde{W}}$ and p_W denote the respective densities of $\widetilde{W}$ and W. We have by Proposition 3.5 of [25], for all $z \in \mathbb{R}^d$,

$$p_{\widetilde{W}}(z) = p_W(z) \star p_W(-z), \tag{C.3}$$

where $\star$ is the convolution operator, is also log-concave. Since covariances add for independent random vectors, $\Sigma_{\widetilde{W}} = 2\Sigma_W$.

By Theorem 4 of [212], for $d \geq 2$

$$h(\widetilde{W}) \geq \frac{d}{2} \log_2 \frac{|\Sigma_{\widetilde{W}}|^{1/d}}{c(d)},$$

where $c(d) = e^2 d^2/(4\sqrt{2}(d+2))$. From Corollary 2.3 of [213],

$$h(\widetilde{W}) = h(W_1 - W_2) \leq h(W) + d \log_2 e,$$

which implies

$$\begin{aligned} h(W) \geq h(\widetilde{W}) - d \log_2 e &\geq \frac{d}{2} \log_2 \frac{|\Sigma_{\widetilde{W}}|^{1/d}}{c(d)} - d \log_2 e \\ &\geq \frac{d}{2} \log_2 \frac{|2\Sigma_W|^{1/d}}{e^2 c(d)} \end{aligned} \tag{C.4}$$

The result follows combining eq. (C.2) and eq. (C.4). □

C.2 Proof of Theorem 6.3.2

$$\mathbb{E}_{Y^i}[h_i(W)] = h_0(W) - \sum_{j=1}^{i} I(W; Y_j|Y^{j-1}) \tag{C.5}$$

$$\geq -i \tag{C.6}$$

from the chain rule for mutual information with $h_0(W) = 0$ and $I(W; Y_j|Y^{j-1}) \leq 1$ [27], and

$$\mathbb{E}_{Y^i}[h_i(W)] \leq \frac{1}{2}\,\mathbb{E}_{Y^i} \log_2((2\pi e)^d |\Sigma_{W|Y^i}|) \tag{C.7}$$

$$\leq \frac{1}{2} \log_2((2\pi e)^d |\mathbb{E}_{Y^i}\, \Sigma_{W|Y^i}|) \tag{C.8}$$

from Lemma 6.3.1 with Jensen's inequality and the concavity of $\log|A|$ for any matrix A in the positive definite cone [159]. Rearranging, we have

$$\frac{2^{-2i}}{(2\pi e)^d} \leq |\mathbb{E}_{Y^i}\, \Sigma_{W|Y^i}| \tag{C.9}$$

$$\leq \frac{\operatorname{Tr}\left(\mathbb{E}_{Y^i}[\Sigma_{W|y^i}]\right)^d}{d^d} \tag{C.10}$$

$$= \frac{(\mathbb{E}_{W,Y^i}[\|W - \mathbb{E}[W|Y^i]\|_2^2])^d}{d^d} \tag{C.11}$$

$$\leq \frac{(\mathbb{E}_{W,Y^i}[\|W - \widehat{W}_i\|_2^2])^d}{d^d} \tag{C.12}$$

where eq. (C.10) is from the AM–GM inequality, eq. (C.11) is due to the linearity of trace and expectation, and the last inequality is due to that fact that expected value is the MMSE estimator, from which the MSE lower bound follows.

C.3 Proof of Proposition 6.3.3

Proof. Consider the 'equiprobable' query scheme, with $P(Y_i = 1|y^{i-1}) = \frac{1}{2}$ for hyperplane query given by weights a_i, threshold τ_i, and noise constant k. Letting $X_i = a_i^T W - \tau_i$, we have

$$\begin{aligned} I(W; Y_i|y^{i-1}) &= H(Y_i|y^{i-1}) - H(Y_i|y^{i-1}, W) \\ &= H(Y_i|y^{i-1}) - H(Y_i|y^{i-1}, W, X_i) \end{aligned}$$

since X_i is a deterministic function of W

$$= H(Y_i|y^{i-1}) - H(Y_i|y^{i-1}, X_i)$$

since $p(Y_i|y^{i-1}, W, X_i) = p(Y_i|y^{i-1}, X_i)$

$$= I(X_i; Y_i|y^{i-1}).$$

Revisiting mutual information, we have

$$I(X_i; Y_i|y^{i-1}) = \mathbb{E}\left[\log_2 \frac{p(Y_i|X_i, y^{i-1})}{p(Y_i|y^{i-1})}\right] \tag{C.13}$$

$$= E_{X_i}[(1 - h_b(f(kX_i))) \,|\, y^{i-1}] \tag{C.14}$$

$$= E_{X_i}[(1 - h_b(f(k|X_i|)))|y^{i-1}] \tag{C.15}$$

since $1 - h_b(f(kX_i))$ is symmetric. From Markov's inequality with $1 - h_b(f(k|X_i|))$ being monotonically increasing, for any $a > 0$,

$$\geq (1 - h_b(f(ka)))P(|X| > a \,|\, y^{i-1}) \tag{C.16}$$

$$\text{(from eq. (C.1))} \quad \geq (1 - h_b(f(ka))) \left(1 - \frac{2a}{\sigma_i}\right) \tag{C.17}$$

$$= \left(1 - h_b\left(f\left(\frac{kc\sigma_i}{2}\right)\right)\right)(1 - c) \tag{C.18}$$

by letting $a = \frac{c\sigma_i}{2}$ for any $0 \leq c \leq 1$ □

C.4 Proof of Theorem 6.3.4

Entropy Properties: Let $h(W|y^i)$ denote the posterior entropy after observing i queries. With a uniform prior distribution over the hypercube $[-\frac{1}{2}, \frac{1}{2}]$, we have that $h(W|y^0) = 0$ and $h(W|y^i) \leq 0$ for $\forall i$ since the uniform distribution maximizes entropy over this bounded space.

After query i, let the eigenvalues of the posterior covariance matrix be denoted in decreasing order as $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_d$. In the equiprobable, max-variance scheme, query a_i is in the direction of maximal eigenvector, so the product of the noise constant and query standard deviation at iteration i is given by $k\sqrt{a_i^T \Sigma_{W|y^i} a_i} = k\|a_i\|\sqrt{\lambda_1} \geq k_{\min}\sqrt{\lambda_1}$. From the monotonicity of the mutual information lower bound on equiprobable queries, we have

$$I(W; Y_i|y^{i-1}) \geq L_{c,k_{\min}}(\sqrt{\lambda_1}) \tag{C.19}$$

From rearranging terms in Lemma 6.3.1 along with $|\Sigma_{W|y^i}| = \prod_{i=1}^d \lambda_i$, we have

$$\frac{2^{2h(W|y^i)}}{(2\pi e)^d} \leq |\Sigma_{W|y^i}| = \prod_{i=1}^d \lambda_i \leq \lambda_1^d \tag{C.20}$$

$$\implies \lambda_1 \geq \frac{2^{\frac{2h(W|y^i)}{d}}}{2\pi e} \tag{C.21}$$

For compactness of notation, let

$$\tilde{L}_{c,k_{\min}}(h) = L_{c,k_{\min}}\left(\frac{2^{\frac{h}{d}}}{\sqrt{2\pi e}}\right) \tag{C.22}$$

Since $L_{c,k_{\min}}$ is monotonically increasing, we have

$$I(W;Y_i|y^{i-1}) \geq \tilde{L}_{c,k_{\min}}(h(W|y^i)) \tag{C.23}$$

Combined with the 1 bit upper bound on mutual information along with $I(W;Y_i|y^{i-1}) = h(W|y^{i-1}) - \mathbb{E}_{Y_i|y^{i-1}}[h(W|y^i)]$, we have

$$\begin{aligned} h(W|y^{i-1}) - 1 &\leq \mathbb{E}_{Y_i|y^{i-1}}[h(W|y^i)] \\ &\leq h(W|y^{i-1}) - \tilde{L}_{c,k_{\min}}(h(W|y^{i-1})) \end{aligned} \tag{C.24}$$

To bound the entropy deviations from one measurement to the next, we need the following lemma:

Lemma C.4.1. *For the equiprobable query scheme,*

$$|h(W|y^i) - h(W|y^{i-1})| \leq \gamma(d) \quad \forall i \geq 0$$

where $\gamma(d) = 8d + \frac{d}{2}\log_2(2\pi e d) + 1$.

The proof of Lemma C.4.1 is highly technical and so we relegate it to the end of the supplementary materials.

Martingale Properties: We note our martingale argument is similar in style to [38]. Let $Z_i = -h(W|y^i)$. From the previous section we have $Z_0 = 0,\ Z_i \geq 0\ \forall i \geq 0$, $|Z_i - Z_{i-1}| \leq \gamma(d)$ from Lemma C.4.1, and $Z_{i-1} + \tilde{L}_{c,k_{\min}}(-Z_{i-1}) \leq E_{Z_i|y^{i-1}}[Z_i] \leq Z_{i-1} + 1$.

Since Z_{i-1} is a deterministic function of y^{i-1} $\forall i$ along with the law of total expectation,

$$\begin{aligned}\mathbb{E}[Z_i|Z_0,\ldots,Z_{i-1}] &= \mathbb{E}_{Y^{i-1}|Z_0,\ldots,Z_{i-1}}\,\mathbb{E}[Z_i|Z_0,\ldots,Z_{i-1},y^{i-1}] \\ &= \mathbb{E}_{Y^{i-1}|Z_0,\ldots,Z_{i-1}}\,\mathbb{E}[Z_i|y^{i-1}]\end{aligned}$$

which implies

$$\begin{aligned}\mathbb{E}[Z_i|Z^{i-1}] &\geq \mathbb{E}_{Y^{i-1}|Z_0,\ldots,Z_{i-1}}[Z_{i-1}+\tilde{L}_{c,k_{\min}}(-Z_{i-1})] \\ &= Z_{i-1}+\tilde{L}_{c,k_{\min}}(-Z_{i-1})\end{aligned}$$

and

$$\begin{aligned}\mathbb{E}[Z_i|Z_0,\ldots,Z_{i-1}] &\leq \mathbb{E}_{Y^{i-1}|Z_0,\ldots,Z_{i-1}}[Z_{i-1}+1] \\ &= Z_{i-1}+1\end{aligned}$$

Since $\tilde{L}_{c,k_{\min}}(-Z_{i-1}) > 0$, we have $\mathbb{E}[Z_i|Z^{i-1}] \geq Z_{i-1}$. For all $i \geq 0$, $|Z_i| < \infty$ since $|Z_i| = |Z_0 + \sum_{j=1}^{i} Z_j - Z_{j-1}| \leq \sum_{j=1}^{i}|Z_j - Z_{j-1}| \leq i\gamma(d) < \infty$. Therefore, Z_i is a submartingale.

Let $\tau > 0$ define a stopping threshold and corresponding stopping time $T = \min\{i : Z_i \geq \tau\}$ Considering $\mathbb{E}[Z_i|Z^{i-1}] \leq Z_{i-1}+1$ and taking the expectation over Z^{i-1} on both sides and expanding with the tower rule, we have

$$\begin{aligned}\mathbb{E}[\mathbb{E}[Z_i|Z^{i-1}]] &\leq \mathbb{E}[Z_{i-1}]+1 \\ \mathbb{E}[Z_i] &\leq \mathbb{E}\,\mathbb{E}[Z_{i-1}|Z^{i-2}]+1 \\ \mathbb{E}[Z_i] &\leq \mathbb{E}[Z_{i-2}]+1+1 \\ &\cdots \\ \mathbb{E}[Z_i] &\leq i\end{aligned}$$

which implies

$$T \geq \mathbb{E}[Z_T] \geq \tau$$

where the last inequality follows by definition, so $\mathbb{E}[T] \geq \tau$. Note that this is true for *any* query selection scheme since mutual information is always upper bounded by 1 bit.

To lower bound the expected stopping time, observe $\tilde{L}_{c,k_{\min}}(-z)$ is monotonically decreasing in z, and $Z_i \leq \tau$ for $i < T$, so we have in this range that $\tilde{L}_{c,k_{\min}}(-Z_i) > \tilde{L}_{c,k_{\min}}(-\tau)$. Using this fact, we construct a separate submartingale that equals Z_i up to and including the stopping time and has the same properties listed above. Specifically, let

$$U_i = \begin{cases} Z_i & i \leq T \\ U_{i-1} + \tilde{L}_{c,k_{\min}}(-\tau) & i > T. \end{cases} \tag{C.25}$$

Clearly for $i \leq T$, $U_i = Z_i$, and if T_U is defined as $T_U = \min\{i : U_i \geq \tau\}$, by observation $T_U = T$. U_i also satisfies $|U_i - U_{i-1}| < \gamma(d)$, and $U_{i-1} + \tilde{L}_{c,k_{\min}}(-\tau) \leq E[U_i|U^{i-1}] \leq U_{i-1} + 1$.

We have

$$E[U_i|U^{i-1}] \geq U_{i-1} + \tilde{L}_{c,k_{\min}}(-\tau) \tag{C.26}$$

$$\frac{E[U_i|U^{i-1}]}{\tilde{L}_{c,k_{\min}}(-\tau)} \geq \frac{U_{i-1}}{\tilde{L}_{c,k_{\min}}(-\tau)} + 1 \tag{C.27}$$

$$\frac{E[U_i|U^{i-1}]}{\tilde{L}_{c,k_{\min}}(-\tau)} - i \geq \frac{U_{i-1}}{\tilde{L}_{c,k_{\min}}(-\tau)} - (i-1) \tag{C.28}$$

We then have a submartingale given by $U_i^{(\text{sub})} = \frac{U_i}{\tilde{L}_{c,k_{\min}}(-\tau)} - i$.

Assume for the time being that the optional stopping theorem can be applied to this submartingale (proved in the sequel)—for any stopping time S satisfying $S \leq T$, $\mathbb{E}[U_S^{\text{sub}}] \leq \mathbb{E}[U_T^{\text{sub}}]$. Specifically, if τ_S is a stopping threshold satisfying $\tau_S \leq \tau$ such that $S = \min\{i :$

$U_i \geq \tau_S\}$, then (for brevity, letting $l(u) = \tilde{L}_{c,k_{\min}}(-u)$)

$$\frac{\mathbb{E}[U_S]}{l(\tau)} - \mathbb{E}[S] \leq \frac{\mathbb{E}[U_T]}{l(\tau)} - \mathbb{E}[T] \tag{C.29}$$

which implies

$$\begin{aligned} \frac{\mathbb{E}[U_S]}{l(\tau_S)} - \mathbb{E}[S] &= \frac{l(\tau)}{l(\tau_S)} \left[\frac{\mathbb{E}[U_S]}{l(\tau)} - \mathbb{E}[S]\right] - \\ &\qquad \left(1 - \frac{l(\tau)}{l(\tau_S)}\right) \mathbb{E}[S] \end{aligned} \tag{C.30}$$

$$\leq \frac{l(\tau)}{l(\tau_S)} \left[\frac{\mathbb{E}[U_T]}{l(\tau)} - \mathbb{E}[T]\right] - \left(1 - \frac{l(\tau)}{l(\tau_S)}\right) \mathbb{E}[S] \tag{C.31}$$

More generally, let $\Delta > 0$ be given and set stopping threshold $\tau_i = i\Delta$, with corresponding stopping time T_i. Define $P_i = \frac{U_{T_i}}{l(\tau_i)} - T_i$. Letting $r_i = \frac{l(\tau_i)}{l(\tau_{i-1})}$ and letting $T = T_i$ and $S = T_{i-1}$, by rearranging the above we have

$$\mathbb{E}[P_i] \geq \frac{\mathbb{E}[P_{i-1}]}{r_i} + \frac{(1 - r_i)}{r_i} \mathbb{E}[T_{i-1}] \tag{C.32}$$

Noting that $\mathbb{E}[T_0] = 0$ since a threshold of τ_0 results in stopping at $T_0 = 0$ and $E[P_0] = \frac{U_{T_0}}{l(\tau_0)} - \mathbb{E}[T_0] = 0$, we continue this bound recursively

$$\begin{aligned} \mathbb{E}[P_i] &\geq \frac{\mathbb{E}[P_{i-2}]}{r_i r_{i-1}} + \frac{(1 - r_{i-1})}{r_i r_{i-1}} \mathbb{E}[T_{i-2}] \\ &\quad + \frac{(1 - r_i)}{r_i} \mathbb{E}[T_{i-1}] \dots \\ &= \sum_{j=1}^{i-1} \frac{1 - r_{j+1}}{\prod_{k=j+1}^{i} r_k} \mathbb{E}[T_j] \\ &= \sum_{j=1}^{i-1} \frac{l(\tau_j) - l(\tau_{j+1})}{l(\tau_i)} \mathbb{E}[T_j] \end{aligned}$$

since $\prod_{k=j+1}^{i} r_k = \frac{l(\tau_i)}{l(\tau_{i-1})}\frac{l(\tau_{i-1})}{l(\tau_{i-2})}\cdots\frac{l(\tau_{j+1})}{l(\tau_j)} = \frac{l(\tau_i)}{l(\tau_j)}$

$$
\begin{aligned}
&= \frac{1}{l(\tau_i)}\sum_{j=1}^{i-1}\frac{l(j\Delta) - l(j\Delta+\Delta)}{\Delta}\Delta\,\mathbb{E}[T_j] \\
&\geq \frac{1}{l(\tau_i)}\sum_{j=1}^{i-1}\frac{l(\tau_j) - l(\tau_j+\Delta)}{\Delta}\tau_j\Delta
\end{aligned}
$$

since $\mathbb{E}[T_j] \geq \tau_j = j\Delta$. Now let $\tau > 0$ be given (with corresponding stopping time defined as T) and let $\Delta \to 0$, choosing i appropriately such that $\tau = \tau_i = i\Delta$

$$
\begin{aligned}
&\geq -\frac{1}{l(\tau)}\int_0^\tau \left(\frac{d}{dx}l(x)\right)x dx \\
&= \frac{1}{l(\tau)}\int_0^\tau l(x)dx - \tau \\
&\implies \frac{\mathbb{E}[U_T]}{l(\tau)} - \mathbb{E}[T] \geq \frac{1}{l(\tau)}\int_0^\tau l(x)dx - \tau \\
\implies \mathbb{E}[T] &\leq \tau + \frac{\mathbb{E}[U_T]}{l(\tau)} - \frac{1}{l(\tau)}\int_0^\tau l(x)dx \\
&\leq \tau + \frac{\tau+1}{l(\tau)} - \frac{1}{l(\tau)}\int_0^\tau l(x)dx
\end{aligned}
$$

since $\mathbb{E}[U_T] = \mathbb{E}[\mathbb{E}[U_T|U^{T-1}]] \leq \mathbb{E}[U_{T-1}] + 1 \leq \tau + 1$

All together we have

$$
\tau \leq \mathbb{E}[T] \leq \tau + \frac{\tau+1}{l(\tau)} - \frac{1}{l(\tau)}\int_0^\tau l(x)dx \tag{C.33}
$$

Now, suppose we'd like to stop the algorithm when the posterior covariance determinant crosses below a threshold, corresponding to a low posterior volume. Denote this threshold as ε, and define the stopping time T_ε as $\min\{i : |\Sigma_{W|y^i}|^{\frac{1}{d}} < \varepsilon\}$. By rearranging the upper

bound in Lemma 6.3.1 we have the necessary condition

$$h_i(W) \leq \frac{d}{2} \log_2(2\pi e \varepsilon) \tag{C.34}$$

Letting $\tau_1 = \frac{d}{2} \log_2(\frac{1}{2\pi e \varepsilon})$ be the entropic stopping threshold with stopping time T_1, from eq. (C.33) this results in (with $\mathbb{E}[T_\varepsilon] \geq \mathbb{E}[T_1]$ since this is a necessary condition)

$$\mathbb{E}[T_\varepsilon] \geq \mathbb{E}[T_1] \geq \tau_1 \tag{C.35}$$

Similarly, by rearranging the lower bound in Lemma 6.3.1 we observe that a sufficient condition for this stopping criterion is

$$h_i(W) \leq \frac{d}{2} \log_2 \frac{2\varepsilon}{e^2 c_d} \tag{C.36}$$

where $c_d = (e^2 d^2)/(4\sqrt{2}(d+2))$. Letting $\tau_2 = \frac{d}{2} \log_2 \frac{e^2 c_d}{2\varepsilon}$ be the entropic stopping threshold with stopping time T_2, we have from eq. (C.33) (with $\mathbb{E}[T_\varepsilon] \leq \mathbb{E}[T_2]$ since this is only a sufficient condition):

$$\mathbb{E}[T_\varepsilon] \leq \mathbb{E}[T_2] \leq \tau_2 + \frac{\tau_2 + 1}{l(\tau_2)} - \frac{1}{l(\tau_2)} \int_0^{\tau_2} l(x) dx \tag{C.37}$$

Combining these, we have the theorem result.

Verifying Optional Stopping Theorem: Consider a submartingale of the form $P_i = \frac{Q_i}{C} - i$ for some $C > 0$, where Q_i is also a submartingale satisfying $Q_i = 0$, $Q_i \geq 0$ for $i \geq 0$, and

$|Q_{i+1} - Q_i| \leq B$ for some $B > C > 0$. This implies

$$\begin{aligned}
|P_i - P_{i-1}| &= \left| \frac{Q_i}{C} - i - \frac{Q_{i-1}}{C} + (i-1) \right| \\
&= \frac{|Q_i - Q_{i-1} - C|}{C} \\
&\leq \frac{|Q_i - Q_{i-1}|}{C} + 1 \\
&\leq \frac{B}{C} + 1 =: B' < \infty
\end{aligned}$$

Let stopping time T_Q be defined as $\min\{i : Q_i > \tau\}$ for some threshold $0 < \tau < \infty$. This implies a stopping time on P_i given by $T_P = \min\{i : P_i > \frac{\tau}{C} - i\}$, with $T \coloneqq T_Q = T_P$. We have from Theorem 5.2.6 of [214] that $P_{T\wedge i}$ and $Q_{T\wedge i}$ are also submartingales.

Consider $\sup \mathbb{E}\, Q^+_{T\wedge i} = \sup \mathbb{E}\, Q_{T\wedge i} \leq \tau + B < \infty$, by definition. From Theorem 5.2.8 of [214], as $i \to \infty$, $Q_{T\wedge i}$ converges a.s. to a limit Q with $\mathbb{E}\,|Q| < \infty$ (and hence $|Q| < \infty$ a.s.). This also implies $|Q_{T\wedge i}| \overset{\text{a.s.}}{\to} |Q|$.

Similarly, $\sup \mathbb{E}\, P^+_{T\wedge i} = \sup \mathbb{E}\left[\left\{ \frac{Q_{T\wedge i}}{C} - (T \wedge i) \right\}^+ \right] \leq \sup \mathbb{E}\left[\frac{Q^+_{T\wedge i}}{C} \right] \leq \frac{\tau + B}{C} < \infty$, so as $i \to \infty$, $P_{T\wedge i}$ converges a.s. to a limit P with $\mathbb{E}\,|P| < \infty$ (and hence $|P| < \infty$ a.s.). This also implies $|P_{T\wedge i}| \overset{\text{a.s.}}{\to} |P|$.

We have

$$\begin{aligned}
T \wedge i &= \left| (T \wedge i) - \frac{Q_{T\wedge i}}{C} + \frac{Q_{T\wedge i}}{C} \right| \\
&\leq \left| (T \wedge i) - \frac{Q_{T\wedge i}}{C} \right| + \frac{|Q_{T\wedge i}|}{C} \\
&= |P_{T\wedge i}| + \frac{|Q_{T\wedge i}|}{C}
\end{aligned}$$

Since the right side converges a.s. to a limit $|P| + \frac{|Q|}{C} =: L$ and $L < \infty$ a.s., for all large enough i, $T \wedge i < L$ a.s. which implies $T < L$ a.s. and therefore $\mathbb{E}[T] < \infty$. Combining this fact with $|P_{i+1} - P_i| \leq B'$, Theorem 5.7.5 of [214] gives that $P_{T\wedge i}$ is uniformly integrable. Then, from Theorem 5.7.4 of [214], for any stopping time $L \leq T$, $\mathbb{E}[P_L] \leq \mathbb{E}[P_T]$.

C.5 Proof of Theorem 6.3.5

To lower bound the complexity of T_ε, we substitute the definition of τ_1 into eq. (C.35), which is true for any query scheme:

$$\mathbb{E}[T_\varepsilon] \geq \frac{d}{2}\log_2\left(\frac{1}{2\pi e\varepsilon}\right) \tag{C.38}$$

$$\implies \mathbb{E}[T_\varepsilon] = \Omega\left(d\log\frac{1}{\varepsilon}\right) \tag{C.39}$$

To upper bound the complexity of T_ε, note that $\tau_2 - \frac{1}{l(\tau_2)}\int_0^{\tau_2} l(x)dx \leq 0$ from the mean value theorem, so $\mathbb{E}[T_\varepsilon] \leq \frac{\tau_2+1}{l(\tau_2)}$. Also note that

$$\begin{aligned} L_{c,k}(\sigma) &= \left(1 - h_b\left(f\left(\frac{ck\sigma}{2}\right)\right)\right)(1-c) \\ &\geq \left(1 - \operatorname{sech}\left(\frac{ck\sigma}{4}\right)\right)(1-c) \qquad \text{(C.40)} \\ &\geq \frac{c^2k^2\sigma^2}{32 + c^2k^2\sigma^2}(1-c) \qquad \text{(C.41)} \end{aligned}$$

where eq. (C.40) is from $h_b(p) \leq 2\sqrt{p(1-p)}$, and eq. (C.41) is from $\operatorname{sech}(x) \leq \frac{2}{2+x^2}$.

Plugging in the definition for τ_2 into $l(\tau_2)$ we have

$$l(\tau_2) = L_{c,k_{\min}}\left(\frac{2^{\frac{-\tau_2}{d}}}{\sqrt{2\pi e}}\right) = L_{c,k_{\min}}\left(\sqrt{\frac{\varepsilon}{\pi e^3 c_d}}\right) \tag{C.42}$$

so

$$l(\tau_2) \geq \frac{c^2k_{\min}^2}{32\pi e^3 c_d\frac{1}{\varepsilon} + c^2k_{\min}^2}(1-c) \tag{C.43}$$

which implies

$$\mathbb{E}[T_\varepsilon] \leq \frac{\left(\frac{d}{2}\log_2\frac{e^2c_d}{2\varepsilon} + 1\right)\left(32\pi e^3 c_d\frac{1}{\varepsilon} + c^2k_{\min}^2\right)}{(1-c)c^2k_{\min}^2} \tag{C.44}$$

$$\implies \mathbb{E}[T_\varepsilon] = O\left(d\log\frac{1}{\varepsilon} + \left(\frac{1}{\varepsilon k_{\min}^2}\right)d^2\log\frac{1}{\varepsilon}\right) \tag{C.45}$$

C.6 Proof of Proposition 6.3.6

Proof. We first bound $p_1 := P(Y = 1)$. Recall that for some fixed k, $f(x) = (1 + e^{-kx})^{-1}$. First note that

$$\begin{aligned}\int_a^b f(x)dx &= \int_a^b \frac{1}{k}\frac{ke^{kx}}{1+e^{kx}}dx \\ &= \frac{1}{k}\int_a^b \frac{u'}{u}dx = \frac{1}{k}\int_{u(a)}^{u(b)} \frac{1}{u}du \\ &= \frac{1}{k}\ln\frac{1+e^{kb}}{1+e^{ka}}.\end{aligned}$$

We have that $P(Y = 1) = \mathbb{E}[P(Y = 1|X = x)] = \mathbb{E}[f(x)]$. Note that $\forall x$, $(1 + e^{-kx}) \leq 1$. Then,

$$\begin{aligned}p_1 = \mathbb{E}[f(x)] &= \int f(x)p_X(x)dx \\ &= \int_{x\leq 0} f(x)p_X(x)dx + \int_{x>0} f(x)p_X(x)dx \\ &\leq \frac{1}{\sigma_X}\int_{-\infty}^{0} f(x)dx + \int_{x>0} f(x)p_X(x)dx \\ &\leq \frac{1}{\sigma_X k}\ln\frac{1+e^{k0}}{1} + \int_{x>0} 1p_X(x)dx \\ &\leq \frac{\ln 2}{\sigma_X k} + P(X > 0) \leq \frac{\ln 2}{\sigma_X k} + 1 - \frac{1}{e},\end{aligned}$$

where we use $p_X(x) \leq 1/\sigma_X$ and the final inequality follows from $P(X \leq 0) \geq \frac{1}{e}$ for zero-mean log-concave X [24]. Using a similar argument it can be shown that $\mathbb{E}[f(x)] \geq 1/e - \ln 2/(\sigma_X k)$. Combining these, we have

$$\frac{1}{e} - \frac{\ln 2}{\sigma_X k} \leq p_1 \leq 1 - \left(\frac{1}{e} - \frac{\ln 2}{\sigma_X k}\right). \tag{C.46}$$

Now we turn to lower bounding $I(X;Y) := H(Y) - H(Y|X)$. The second term can

be written

$$\begin{aligned} H(Y|X) &= \mathbb{E}_X\, H(Y|X=x) \\ &= \int_{-\infty}^{\infty} h_b(f(x)) p_X(x) dx \\ &\leq \frac{1}{\sigma_X} \int_{-\infty}^{\infty} h_b(f(x)) dx. \end{aligned} \tag{C.47}$$

where the inequality follows from Lemma C.0.1. Since

$$\begin{aligned} & H(Y|X=x) \\ &\quad = -f(x)\log_2 f(x) - (1-f(x))\log_2(1-f(x)) \\ &\quad = \frac{1}{1+e^{-kx}} \log_2(1+e^{-kx}) \\ &\qquad + \frac{e^{-kx}}{1+e^{-kx}} \log_2((1+e^{-kx})/e^{-kx}) \\ &\quad = \frac{1+e^{-kx}}{1+e^{-kx}} \log_2(1+e^{-kx}) \\ &\qquad - \frac{e^{-kx}}{1+e^{-kx}} \log_2(e^{-kx}) \\ &\quad = \log_2(1+e^{-kx}) + \frac{kxe^{-kx}\log_2(e)}{1+e^{-kx}}, \end{aligned}$$

which is an even function, we have (omitting details of the integration)

$$\begin{aligned} H(Y|X) &\leq \frac{2}{\sigma_X} \int_0^{\infty} \log_2(1+e^{-kx}) \\ &\qquad + \frac{kxe^{-kx}\log_2(e)}{1+e^{-kx}} dx \\ &= \frac{\pi^2(\log_2 e)}{3k\sigma_X} \end{aligned} \tag{C.48}$$

For the second term, note that $H(Y=1) = h_b(p_1)$. The binary entropy function is symmetric about, and monotonically decreasing from $p = 1/2$. Therefore,

$$H(Y) = h_b(p_1) \geq h_b\left(\frac{1}{e} - \frac{\ln 2}{\sigma_X k}\right) \tag{C.49}$$

Combining eq. (C.48) and eq. (C.49) gives the desired result. □

C.7 Proof of Lemma C.4.1

Proof. Since $p(W|y^i)$ is log-concave, and by Jensen's inequality,

$$\begin{aligned} -h(W|y^i) &= \mathbb{E}_{W|y^i}[\log_2 p(W|y^i)] \\ &\leq \log_2 p(\mathbb{E}[W|y^i]|y^i) \\ &\leq \log_2 \sup_w p(w|y^i). \end{aligned}$$

Without loss of generality, we may suppose $\mathbb{E}[W|y^i] = 0$, and let $V = \Sigma_{W|y^i}^{-\frac{1}{2}} W$ and $W \sim P_{W|y^i}$, such that $\mathbb{E}[V] = 0$ and $\mathbb{E}[VV^T] = \Sigma_{W|y^i}^{-\frac{1}{2}} \mathbb{E}[WW^T] \Sigma_{W|y^i}^{-\frac{1}{2}} = \Sigma_{W|y^i}^{-\frac{1}{2}} \Sigma_{W|y^i} \Sigma_{W|y^i}^{-\frac{1}{2}} = I$ and therefore V is isotropic. From [215] we have that $p_V(v) \leq 2^{8d} d^{\frac{d}{2}}$. From the density of a linear transformation of a random variable we have

$$p_{W|y^i}(w) = \frac{p_V(\Sigma_{W|y^i}^{-\frac{1}{2}} w)}{|\Sigma_{W|y^i}^{\frac{1}{2}}|} \leq \frac{2^{8d} d^{\frac{d}{2}}}{|\Sigma_{W|y^i}|^{\frac{1}{2}}}.$$

Therefore, for our query strategy we have (with $f_i(W)$ denoting the logistic response model for the query at iteration i)

$$\begin{aligned} p(w|y^i) &= p(w|y_i = y, y^{i-1}) \\ &= \frac{f_i(W)y + (1 - f_i(W))(1-y)}{p(y_i = y|y^{i-1})} p(W|y^{i-1}) \\ &\leq \frac{(1)y + (1-(0))(1-y)}{p(y_i = y|y^{i-1})} p(W|y^{i-1}) \\ &= \frac{1}{p(y_i = y|y^{i-1})} p(W|y^{i-1}) \\ &\leq \frac{1}{p(y_i = y|y^{i-1})} \frac{2^{8d} d^{\frac{d}{2}}}{|\Sigma_{W|y^{i-1}}|^{\frac{1}{2}}} \\ \implies \sup_w p(w|y^i) &\leq \frac{1}{p(y_i = y|y^{i-1})} \frac{2^{8d} d^{\frac{d}{2}}}{|\Sigma_{W|y^{i-1}}|^{\frac{1}{2}}}, \end{aligned}$$

which implies

$$\log_2 \sup_w p(w|y^i) \leq 8d + \frac{d}{2}\log_2 d - \frac{1}{2}\log_2|\Sigma_{W|y^{i-1}}| - \log_2(p(y_i = y|y^{i-1})),$$

and hence

$$\begin{aligned} h(W|y^i) &\geq \frac{1}{2}\log_2|\Sigma_{W|y^{i-1}}| + \log_2(p(y_i = y|y^{i-1})) \\ &\quad - \left(8d + \frac{d}{2}\log_2 d\right) \\ &\geq \frac{1}{2}\log_2((2\pi e)^d|\Sigma_{W|y^{i-1}}|) \\ &\quad - \frac{1}{2}\log_2(2\pi e)^d + \log_2(p(y_i = y|y^{i-1})) \\ &\quad - \left(8d + \frac{d}{2}\log_2 d\right) \\ &\geq h(W|y^{i-1}) + \log_2(p(y_i = y|y^{i-1})) \\ &\quad - \left(8d + \frac{d}{2}\log_2(2\pi e d)\right) \quad \text{from eq. (C.2).} \end{aligned}$$

For equiprobable queries $p(y_i = y|y^{i-1}) = 1/2$, and so we have

$$h(W|y^{i-1}) - h(W|y^i) \leq \gamma(d). \tag{C.50}$$

where $\gamma(d) = 8d + \frac{d}{2}\log_2(2\pi e d) + 1$.

To obtain the other direction, let $h_y^{i-1} = h(W|Y_i = y, y^{i-1})$, $y_m = \arg\min_{y\in\{0,1\}} h_y^{i-1}$, $y_M = 1 - y_m$. Note that $h_{y_M}^{i-1} \geq h_{y_m}^{i-1}$. We have

$$\begin{aligned} h(W|Y_i, y^{i-1}) &= \frac{1}{2}h_m^{i-1} + \frac{1}{2}h_M^{i-1} \\ &\geq \frac{1}{2}(h(W|y^{i-1}) - \gamma(d)) + \frac{1}{2}h_M^{i-1} \\ &\geq \frac{1}{2}(h(W|y^{i-1}) - \gamma(d)) + \frac{1}{2}h(W|y^i) \end{aligned}$$

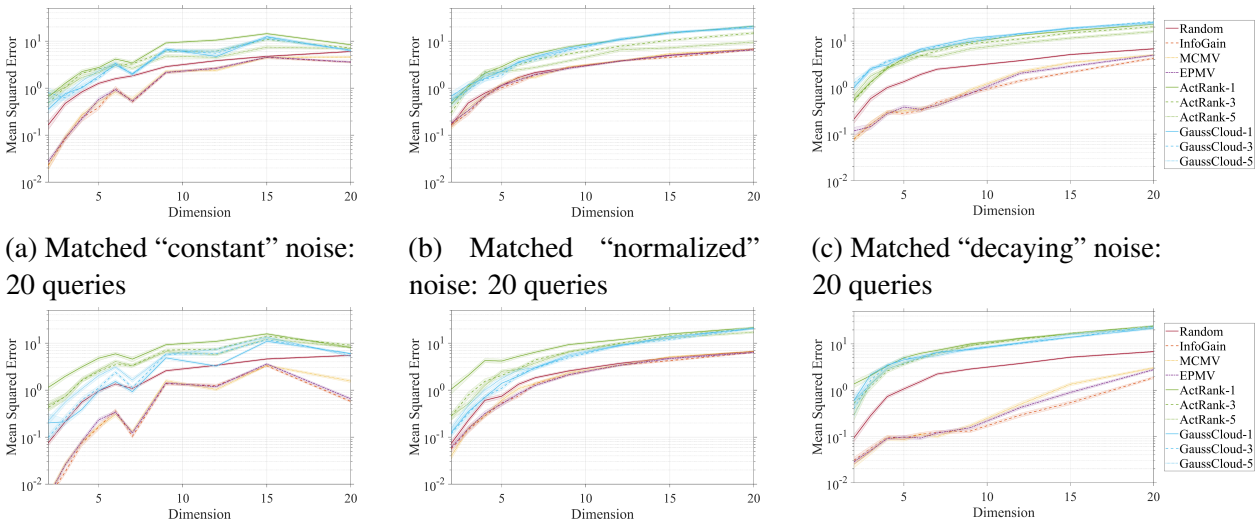
where the first inequality follows from eq. (C.50) and the second inequality follows from the definition of h_M. From the non-negativity of mutual information, we have that $h(W|Y_i, y^{i-1}) \leq h(W|y^{i-1})$, implying

$$h(W|y^{i-1}) \geq \frac{1}{2}(h(W|y^{i-1}) - \gamma(d)) + \frac{1}{2}h(W|y^i)$$
$$h(W|y^{i-1}) - h(W|y^i) \geq -\gamma(d) \tag{C.51}$$

Combining eq. (C.51) with eq. (C.50) we have the desired result. □

C.8 Additional Experiments

Performance Across Dimensions: Figure C.1 plots MSE against embedding dimension averaged across all trials at both 20 and 60 queries asked. For all dimensions across all experiments, the learned Yummly `Food-10k` embedding was centered and scaled by a constant amount such that the unit hypercube of user preference points would be contained in the embedding of items, allowing for a rich pool of pairs to be selected from for any user point. This scaling constant was heuristically set to $\sqrt{d}/(3\tilde{\lambda}^{1/2})$, where $\tilde{\lambda}$ is the smallest eigenvalue of the covariance matrix of embedding items. This scaling is motivated by setting the smallest variance direction of the embedding to align with the furthest point of the unit cube at a distance of $\sqrt{d}$ from the origin. For each learned embedding, responses to the Yummly `Food-10k` training triplets were predicted by selecting the closer of the two comparison items to the reference item, using the embedding to measure distances. For a given embedding, we refer to the fraction of incorrectly predicted triplet responses as the *triplet error fraction*, which we plot for reference against embedding dimension in Figure C.2. For all experiments, $\beta = 10^{-3}$ and results are averaged over 50 trials.



(a) Matched "constant" noise: 20 queries

(b) Matched "normalized" noise: 20 queries

(c) Matched "decaying" noise: 20 queries

(d) Matched "constant" noise: 60 queries

(e) Matched "normalized" noise: 60 queries

(f) Matched "decaying" noise: 60 queries

Figure C.1: Mean squared error performance across dimensions at a fixed number of answered queries, plotted with $\pm$ one standard error.

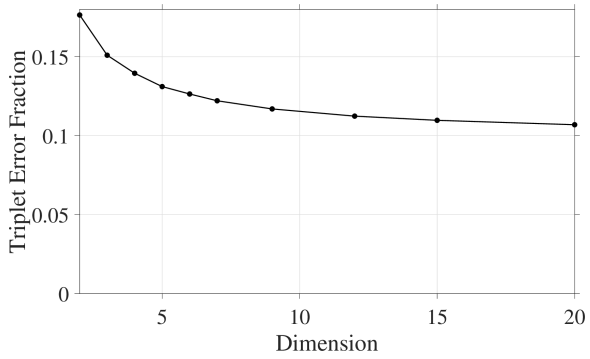
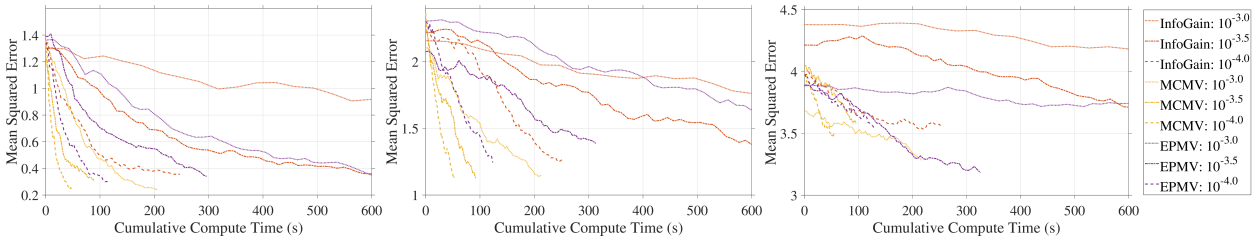


Figure C.2: Triplet error fraction versus embedding dimension.

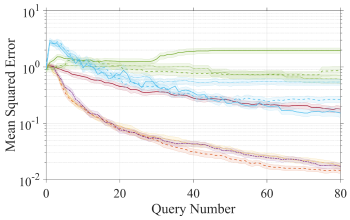
Speed Plot Comparison Figure C.3 plots MSE against cumulative compute time for matched logistic noise with "normalized" noise constant for $d \in \{4, 7, 12\}$ in a smaller scale experiment of 60 queries per trial, and 40 trials per dimension. Specifically, MSE and average cumulative compute time were calculated for *each* number of queries asked, and these two values plotted against each other directly in a range up to 600 seconds. We evaluated all three of our methods (InfoGain, MCMV, EPMV) at various pair pool downsampling rates of $\beta \in \{10^{-3}, 10^{-3.5}, 10^{-4}\}$, as listed in the figure legend next to each method. Each experiment was run on an Intel Xeon CPU E5-2680 v4 2.40 GHz processor.



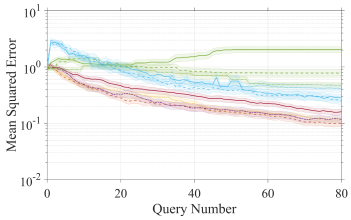
(a) $d = 4$ (b) $d = 7$ (c) $d = 12$

Figure C.3: Mean squared error performance against cumulative compute time (s) for matched, "normalized" logistic noise at various pair downsampling rates. Error bars have been omitted for visual clarity.

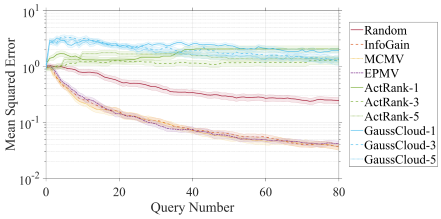
Additional Experimental Results In this section, MSE is evaluated for both matched and mismatched noise at $d \in \{3, 5, 7, 9, 12\}$ in Figures C.4 to C.8. The model for k_{pq} on mismatched Gaussian noise is chosen as the maximum-likelihood model ("constant," "normalized", "decaying") on the training triplets, calculated separately for each embedding dimension. For all experiments, $\beta = 10^{-3}$ and results are averaged over 50 trials.



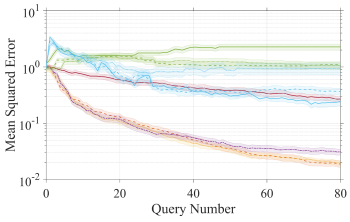
(a) "Constant" model, matched, $d = 3$



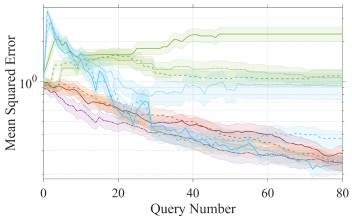
(b) "Normalized" model, matched, $d = 3$



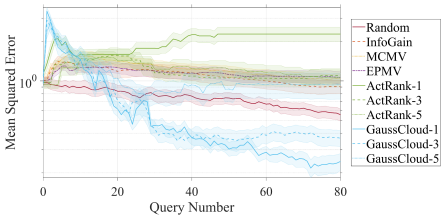
(c) "Decaying" model, matched, $d = 3$



(d) "Constant" model, mismatched, $d = 3$

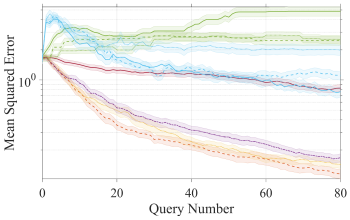


(e) "Normalized" model, mismatched, $d = 3$

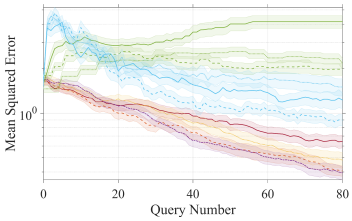


(f) "Decaying" model, mismatched, $d = 3$

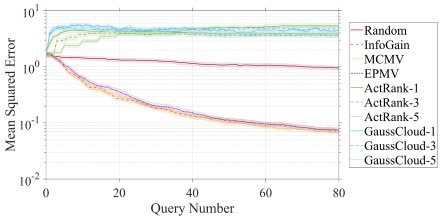
Figure C.4: Mean squared error performance versus number of queries asked for pairwise search in 3 dimensions, plotted with $\pm$ one standard error. All mismatched noise is Gaussian with a "constant" noise constant.



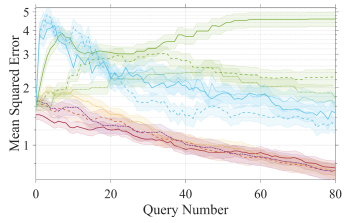
(a) "Constant" model, matched, $d = 5$



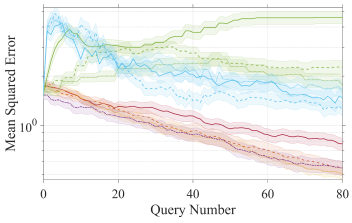
(b) "Normalized" model, matched, $d = 5$



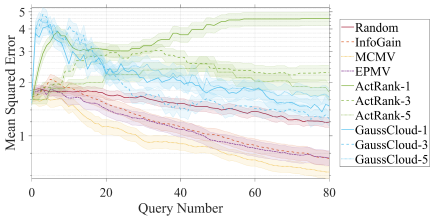
(c) "Decaying" model, matched, $d = 5$



(d) "Constant" model, mismatched, $d = 5$

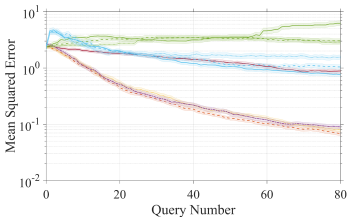


(e) "Normalized" model, mismatched, $d = 5$

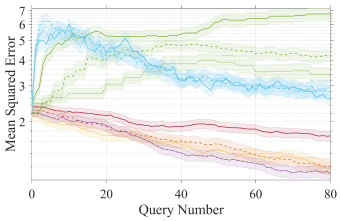


(f) "Decaying" model, mismatched, $d = 5$

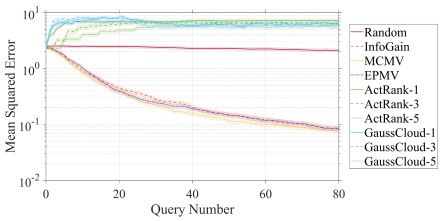
Figure C.5: Mean squared error performance versus number of queries asked for pairwise search in 5 dimensions, plotted with $\pm$ one standard error. All mismatched noise is Gaussian with a "normalized" noise constant.



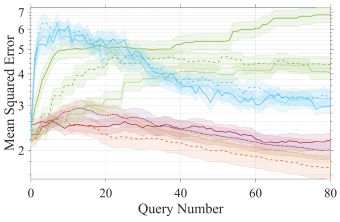
(a) "Constant" model, matched, $d = 7$



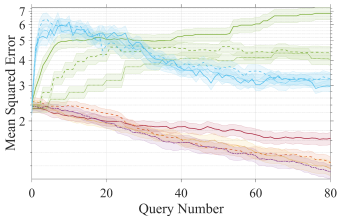
(b) "Normalized" model, matched, $d = 7$



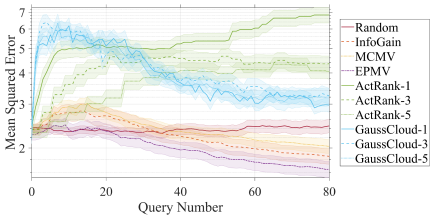
(c) "Decaying" model, matched, $d = 7$



(d) "Constant" model, mismatched, $d = 7$

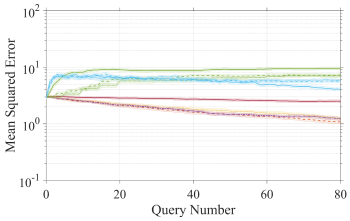


(e) "Normalized" model, mismatched, $d = 7$

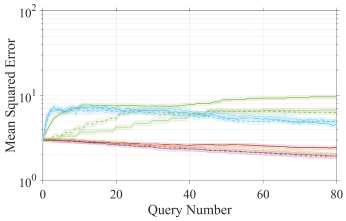


(f) "Decaying" model, mismatched, $d = 7$

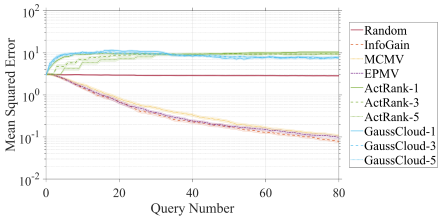
Figure C.6: Mean squared error performance versus number of queries asked for pairwise search in 7 dimensions, plotted with $\pm$ one standard error. All mismatched noise is Gaussian with a "normalized" noise constant.



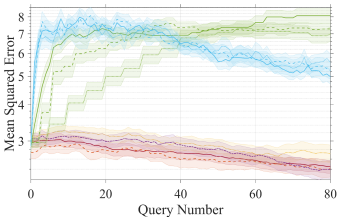
(a) "Constant" model, matched, $d = 9$



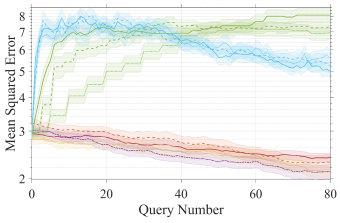
(b) "Normalized" model, matched, $d = 9$



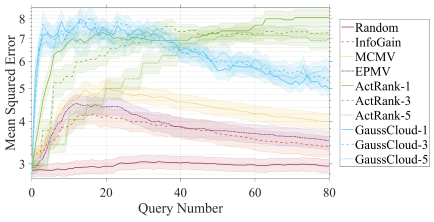
(c) "Decaying" model, matched, $d = 9$



(d) "Constant" model, mismatched, $d = 9$

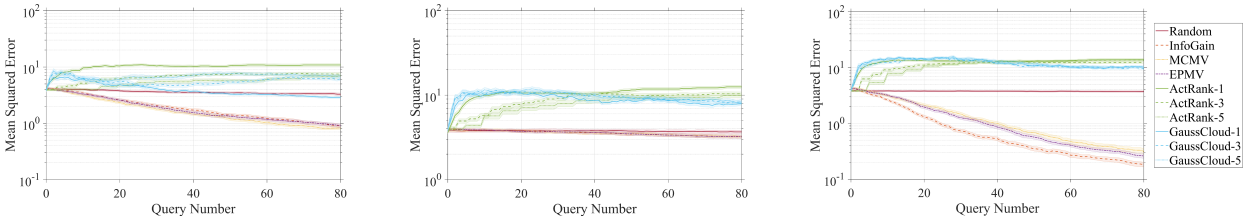


(e) "Normalized" model, mismatched, $d = 9$



(f) "Decaying" model, mismatched, $d = 9$

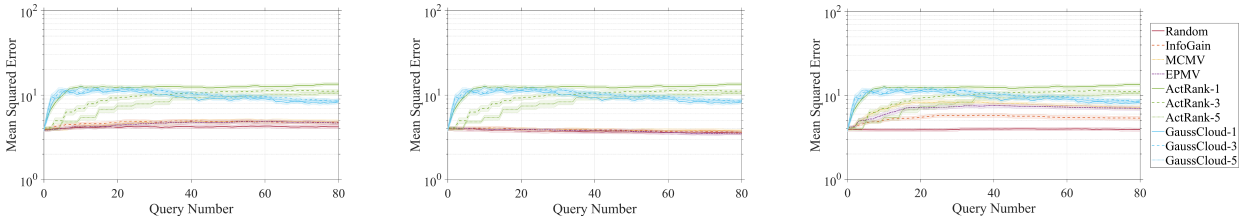
Figure C.7: Mean squared error performance versus number of queries asked for pairwise search in 9 dimensions, plotted with $\pm$ one standard error. All mismatched noise is Gaussian with a "normalized" noise constant.



(a) "Constant" model, matched, $d = 12$

(b) "Normalized" model, matched, $d = 12$

(c) "Decaying" model, matched, $d = 12$



(d) "Constant" model, mismatched, $d = 12$

(e) "Normalized" model, mismatched, $d = 12$

(f) "Decaying" model, mismatched, $d = 12$

Figure C.8: Mean squared error performance versus number of queries asked for pairwise search in 12 dimensions, plotted with $\pm$ one standard error. All mismatched noise is Gaussian with a "normalized" noise constant.

APPENDIX D

PROOFS AND ADDITIONAL DETAILS IN FEEDBACK CODING FOR ACTIVE LEARNING

D.1 Proofs of Analytical Results

D.1.1 Proof of Proposition 2.1

Proof. Our proof follows closely to that of [177] for the capacity of the one-bit quantized Gaussian channel. We start by writing $I(L;Y) = H(Y) - H(Y \mid L)$, where H denotes the entropy of a discrete random variable [27]. $H(Y)$ is maximized at 1 bit, when $p(Y=1) = p(Y=-1) = 0.5$. Expanding $H(Y \mid L)$, we have $H(Y \mid L) = \mathbb{E}_{p_L}[h_b(p(Y=1 \mid L))] = \mathbb{E}_{p_L}[h_b(f(L))]$.

For distribution p_L, consider its symmetrized distribution $\widetilde{p}_L(\ell) = \frac{1}{2}p_L(\ell) + \frac{1}{2}p_L(-\ell)$ and the expectation of any even function $e(\cdot)$ over $\widetilde{p}_L(\ell)$:

$$
\begin{aligned}
E_{\widetilde{p}_L}[e(L)] &= \int_{-\infty}^{\infty} \left(\frac{1}{2}p_L(\ell) + \frac{1}{2}p_L(-\ell)\right) e(\ell) d\ell \\
&= \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)e(\ell)d\ell + \frac{1}{2}\int_{-\infty}^{\infty} p_L(-\ell)e(\ell)d\ell \\
&= \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)e(\ell)d\ell + \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)e(-\ell)d\ell \qquad \text{change of variables} \\
&= \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)e(\ell)d\ell + \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)e(\ell)d\ell \qquad e(\ell) \text{ is even} \\
&= \frac{1}{2}E_{p_L}[e(L)] + \frac{1}{2}E_{p_L}[e(L)] \\
&= E_{p_L}[e(L)]
\end{aligned}
$$

Observe that h_b is symmetric about 0.5, i.e. for $x \in [-0.5, 0.5]$, $h_b(0.5+x) = h_b(0.5-x)$. Combining this with the fact that $f(\ell) - 0.5$ is an odd function (i.e. $f(-\ell) - 0.5 =$

$-(f(\ell)-0.5))$, we have

$$h_b(f(-\ell)) = h_b(f(-\ell)-0.5+0.5) = h_b(-(f(\ell)-0.5)+0.5) = h_b((f(\ell)-0.5)+0.5) = h_b(f(\ell))$$

and so $h_b(f(\ell))$ is an even function. Therefore, the conditional entropy $H(Y \mid L)$ is equivalent when L is distributed as p_L or $\widetilde{p}_L$, i.e. $E_{\widetilde{p}_L}[h_b(f(L))] = E_{p_L}[h_b(f(L))]$.

We also have

$$\begin{aligned}
E_{\widetilde{p}_L}[f(L)] &= E_{\widetilde{p}_L}[f(L)-0.5]+0.5 \\
&= \int_{-\infty}^{\infty}\left(\frac{1}{2}p_L(\ell)+\frac{1}{2}p_L(-\ell)\right)(f(\ell)-0.5)d\ell + 0.5 \\
&= \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)(f(\ell)-0.5)d\ell + \frac{1}{2}\int_{-\infty}^{\infty} p_L(-\ell)(f(\ell)-0.5)d\ell + 0.5 \\
&= \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)(f(\ell)-0.5)d\ell + \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)(f(-\ell)-0.5)d\ell + 0.5 \quad \text{change of variables} \\
&= \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)(f(\ell)-0.5)d\ell - \frac{1}{2}\int_{-\infty}^{\infty} p_L(\ell)(f(\ell)-0.5)d\ell + 0.5 \qquad (f(\ell)-0.5) \text{ is odd} \\
&= 0.5
\end{aligned}$$

and so under $\widetilde{p}_L$, $p(Y=1) = \mathbb{E}_{\widetilde{p}_L}[f(L)] = 0.5$ and $H(Y)$ is maximized at 1 bit.

Combining these facts, we have

$$I(\widetilde{p}_L, f) = 1-E_{\widetilde{p}_L}[h_b(f(L))] = 1-E_{p_L}[h_b(f(L))] \geq h_b(E_{p_L}[f(L)])-E_{p_L}[h_b(f(L))] = I(p_L, f)$$

and so symmetrizing a distribution can only increase $I(L;Y)$. Furthermore, since ℓ^2 is even we have $\mathbb{E}_{\widetilde{p}_L}[L^2] = \mathbb{E}_{p_L}[L^2]$. Therefore, when evaluating the capacity of channel with transition probability f under power constraint P, we only consider symmetric distributions since for every $p_L \in \mathcal{C}_P$ there exists a symmetric distribution $\widetilde{p}_L \in \mathcal{C}_P$ satisfying $I(\widetilde{p}_L, f) \geq I(p_L, f)$. We solve for the capacity-achieving distribution over the set of

symmetric distributions in $\mathcal{C}_P$:

$$p_L^* = \underset{\substack{\mathbb{E}_{p_L}[L^2]\leq P \\ p_L(\ell)=p_L(-\ell)}}{\arg\max} \; I(p_L, f) \tag{D.1}$$

$$= \underset{\substack{\mathbb{E}_{p_L}[L^2]\leq P \\ p_L(\ell)=p_L(-\ell)}}{\arg\max} \; 1 - E_{p_L}[h_b(f(L))]$$

$$= \underset{\substack{\mathbb{E}_{p_L}[L^2]\leq P \\ p_L(\ell)=p_L(-\ell)}}{\arg\min} \; E_{p_L}[h_b(f(L))] \tag{D.2}$$

Since $h_b(f(\ell))$ is even, $h_b(f(\ell)) = h_b(f(|\ell|)) = h_b(f(\sqrt{\ell^2}))$. Omitting calculations, we have

$$\frac{d^2}{du^2} h_b(f(\sqrt{u})) = (\log_2 e) \frac{\tanh(\frac{\sqrt{u}}{2}) \operatorname{sech}^2(\frac{\sqrt{u}}{2})}{16\sqrt{u}}$$

which is non-negative for $u > 0$ and therefore $h_b(f(\sqrt{u}))$ (which is continuous on $u \geq 0$) is convex on $u \geq 0$. We then have

$$E_{p_L}[h_b(f(L))] = E_{p_L}\left[h_b\big(f\big(\sqrt{L^2}\big)\big)\right] \overset{(a)}{\geq} h_b\Big(f\Big(\sqrt{E_{p_L}[L^2]}\Big)\Big) \overset{(b)}{\geq} h_b\big(f\big(\sqrt{P}\big)\big)$$

where Jensen's inequality is used in (a) [27], with equality if and only if L^2 is constant, and (b) results from the power constraint $E_{p_L}[L^2] \leq P$ and the fact that $h_b(f(\sqrt{u}))$ is monotonically decreasing for $u \geq 0$. For symmetric p_L, equality in (a) is achieved if $p_L = B_t$ for some $t > 0$. By setting $t = \sqrt{P}$, equality in (b) is also achieved, and so $B_{\sqrt{P}}$ minimizes eq. (D.2) (and therefore maximizes eq. (D.1)). The maximum value in eq. (D.1), which is equal to capacity C, is then

$$I(B_{\sqrt{P}}, f) = 1 - \mathbb{E}_{B_{\sqrt{P}}}[h_b(f(L))] = 1 - \frac{1}{2} h_b(f(\sqrt{P})) - \frac{1}{2} h_b(f(-\sqrt{P})) = 1 - h_b(f(\sqrt{P})).$$

□

D.1.2 Proof of Proposition 2.2

Proof. Since p_θ is log-concave, then $p_{\theta|\mathcal{L}_{n-1}}(\theta) \propto p_\theta(\theta)\prod_{i=1}^{n-1} p(Y = y_i \mid x_i, \theta)$ is also log-concave since it is the product of log-concave functions. Since marginals of log-concave distributions are log-concave, $L_n = x_n^T\theta$ is log-concave for any x_n under the distribution $p_{\theta|\mathcal{L}n-1}$. However, we know from Proposition 2.1 that p_L^* for logistic regression is a sum of mass points, which is not log-concave. Therefore no x_n exists which can induce p_L^* from h. □

D.1.3 Proof of Theorem 3.1

Proof. In the following, suppose that $p_L \in \mathcal{C}_P$, and let $H_{p_L}(Y) = h_b(\mathbb{E}_{p_L}[f(L)])$ and $H_{p_L}(Y \mid L) = \mathbb{E}_{p_L}[h_b(f(L))]$. $f(\ell)$ is K_1-Lipschitz, where $K_1 = 0.25$, and $h_b(f(\ell))$ is K_2-Lipschitz, where $K_2 \approx 0.32$.

$$\begin{aligned}
|I(p_L, f) - I(B_t, f)| &= |H_{p_L}(Y) - H_{p_L}(Y \mid L) - (H_{B_t}(Y) - H_{B_t}(Y \mid L))| \\
&\leq |H_{p_L}(Y) - H_{B_t}(Y))| + |H_{p_L}(Y \mid L) - H_{B_t}(Y \mid L)| \\
&= |h_b(\mathbb{E}_{p_L}[f(L)]) - h_b(\mathbb{E}_{B_t}[f(L)])| + \left|\int_\ell h_b(f(\ell))p_L(\ell)d\ell - \int_\ell h_b(f(\ell))B_t(\ell)d\ell\right|
\end{aligned}$$

Assume that there exists $\varepsilon \in (0, 0.5)$ such that $\varepsilon \leq \mathbb{E}_{p_L}[f(L)] \leq 1 - \varepsilon$. For $\ell \in (\varepsilon, 1-\varepsilon)$, h_b is $\log_2 \frac{1-\varepsilon}{\varepsilon}$-Lipschitz. Since $\varepsilon < \mathbb{E}_{p_L}[f(L)] < 1 - \varepsilon$ by assumption and B_t satisfies $\varepsilon < \mathbb{E}_{B_t}[f(L)] < 1 - \varepsilon$ since $\mathbb{E}_{B_t}[f(L)] = 0.5$, we have

$$\begin{aligned}
|h_b(\mathbb{E}_{p_L}[f(L)]) - h_b(\mathbb{E}_{B_t}[f(L)])| &\leq \log_2\Big(\frac{1-\varepsilon}{\varepsilon}\Big)|\mathbb{E}_{p_L}[f(L)] - \mathbb{E}_{B_t}[f(L)]| \\
&= \log_2\Big(\frac{1-\varepsilon}{\varepsilon}\Big)\left|\int_\ell f(\ell)p_L(\ell)d\ell - \int_\ell f(\ell)B_t(\ell)d\ell\right|
\end{aligned}$$

which implies

$$|I(p_L, f) - I(B_t, f)| \leq \log_2\Big(\frac{1-\varepsilon}{\varepsilon}\Big)\left|\int_\ell f(\ell)p_L(\ell)d\ell - \int_\ell f(\ell)B_t(\ell)d\ell\right| + \left|\int_\ell h_b(f(\ell))p_L(\ell)d\ell - \int_\ell h_b(f(\ell))B_t(\ell)d\ell\right| \tag{D.3}$$

To continue, we use the following result from [179]: defining $P_1(\mathbb{R}) \coloneqq \{\mu' : \mathbb{E}_{\mu'}[|L|] < \infty\}$, for any $\mu, \nu \in P_1(\mathbb{R})$ we have

$$\sup_{\|f\|_{\text{Lip}} \leq 1} \int_\ell f(\ell)\mu(\ell)d\ell - \int_\ell f(\ell)\nu(\ell)d\ell = W_1(\mu, \nu).$$

Therefore, for any K-Lipschitz function g we have that $\frac{g}{K}$ is 1-Lipschitz and so

$$\begin{aligned}
\left|\int_\ell g(\ell)\mu(\ell)d\ell - \int_\ell g(\ell)\nu(\ell)d\ell\right| &= K\left|\int_\ell \frac{g(\ell)}{K}\mu(\ell)d\ell - \int_\ell \frac{g(\ell)}{K}\nu(\ell)d\ell\right| \\
&= K\max\left\{\int_\ell \frac{g(\ell)}{K}\mu(\ell)d\ell - \int_\ell \frac{g(\ell)}{K}\nu(\ell)d\ell, \int_\ell \frac{-g(\ell)}{K}\mu(\ell)d\ell - \int_\ell \frac{-g(\ell)}{K}\nu(\ell)d\ell\right\} \\
&\leq K \sup_{\|f\|_{\text{Lip}} \leq 1} \int_\ell f(\ell)\mu(\ell)d\ell - \int_\ell f(\ell)\nu(\ell)d\ell \\
&\leq KW_1(\mu, \nu) \\
&\leq KW_2(\mu, \nu) \qquad \text{(D.4)}
\end{aligned}$$

where the last inequality is from $W_1(\mu, \nu) \leq W_2(\mu, \nu)$ [179].

To apply this inequality to both expressions in eq. (D.3), we first verify that $p_L, B_t \in P_1(\mathbb{R})$. $\mathbb{E}_{B_t}[|L|] = t < \infty$, and

$$\mathbb{E}_{p_L}[|L|] = \mathbb{E}_{p_L}\left[\sqrt{L^2}\right] \overset{(a)}{\leq} \sqrt{\mathbb{E}_{p_L}[L^2]} \overset{(b)}{\leq} \sqrt{P} < \infty$$

where (a) results from Jensen's inequality with the concavity of $\sqrt{\cdot}$, and (b) is since $\mathbb{E}_{p_L}[L^2] \leq P$ by assumption and $\sqrt{\cdot}$ is monotonically increasing. Applying eq. (D.4) separately to both terms in eq. (D.3), we have

$$|I(p_L, f) - I(B_t, f)| \leq \left(K_1 \log_2\left(\frac{1-\varepsilon}{\varepsilon}\right) + K_2\right) W_2(p_L, B_t) \qquad \text{(D.5)}$$

Finally, we compute a valid value of ε for all $p_L \in \mathcal{C}_P$. First note that $f(\ell) < 0.5$ for $\ell < 0$ and $f(\ell) \geq 0.5$ for $\ell \geq 0$, implying that $f(\ell) \leq f(|\ell|)\ \forall \ell$. Next note that $f(\sqrt{u})$ is

concave on $u \geq 0$, since for any $u, v \in [0, \infty)$ and any $0 < \phi < 1$

$$f(\sqrt{\phi u + (1-\phi)v}) \geq f(\phi\sqrt{u} + (1-\phi)\sqrt{v})$$

since f is monotonically increasing and $\sqrt{\cdot}$ is concave.

$$\geq \phi f(\sqrt{u}) + (1-\phi) f(\sqrt{v})$$

since f is concave on $\mathbb{R}_{\geq 0}$. This can be shown by considering

$$\frac{d^2}{du^2} f(u) = \frac{e^u}{(1+e^u)^3}(1-e^u) \leq 0 \; \forall u \geq 0$$

Combining these facts, we have

$$\begin{aligned}
\mathbb{E}_{p_L}[f(L)] &\leq \mathbb{E}_{p_L}[f(|L|)] && \text{since } f(\ell) \leq f(|\ell|) \\
&= \mathbb{E}_{p_L}\left[f\left(\sqrt{L^2}\right)\right] \\
&\leq f\left(\sqrt{\mathbb{E}_{p_L}[L^2]}\right) && \text{from Jensen's inequality with the concavity of } f(\sqrt{\cdot}) \\
&\leq f(\sqrt{P})
\end{aligned}$$

since $f(\sqrt{\cdot})$ is monotonically increasing, and by assumption $E_{p_L}[L^2] \leq P$. Similarly, $\mathbb{E}_{p_L}[1 - f(L)] \leq f(\sqrt{P})$, and therefore we can set $\varepsilon = 1 - f(\sqrt{P})$. Applying this choice of ε to eq. (D.5) we have

$$|I(p_L, f) - I(B_t, f)| \leq \left(K_1 \log_2\left(\frac{f(\sqrt{P})}{1 - f(\sqrt{P})}\right) + K_2\right) W_2(p_L, B_t)$$

and can set $K_P = K_1 \log_2\left(\frac{f(\sqrt{P})}{1-f(\sqrt{P})}\right) + K_2$ to obtain $|I(p_L, f) - I(B_t, f)| \leq K_P W_2(p_L, B_t)$.

Recall that $C = \max_{p_L \in \mathcal{C}_P} I(p_L, f) = I(B_{\sqrt{P}}, f)$ and $\widetilde{C}_n = \max_{x \in \mathcal{U}_n} I(p_{L_n|\mathcal{L}_{n-1}}, f)$. By assumption, P is selected such that $p_{L_n|\mathcal{L}_{n-1}} \in \mathcal{C}_P$ for any $x \in \mathcal{U}_n$, which implies

$I(p_{L_n|\mathcal{L}_{n-1}}, f) \leq C$ for any $x \in \mathcal{U}_n$ and hence $\widetilde{C}_n \leq C$. Combining these facts, we have

$$\widetilde{C}_n - I(p_{L_n|\mathcal{L}_{n-1}}, f) \leq C - I(p_{L_n|\mathcal{L}_{n-1}}, f) = |I(B_{\sqrt{P}}, f) - I(p_{L_n|\mathcal{L}_{n-1}}, f)| \leq K_P W_2(p_{L_n|\mathcal{L}_{n-1}}, B_{\sqrt{P}}).$$

□

D.1.4 Proof of Proposition 3.2

Proof. Adopting notation from [216], let S denote a finite set of points in $\mathbb{R}$, and $w \colon S \to \mathbb{R}$ a weight vector. Define $\mathrm{Vor}_S^w(p) = \{\ell : \|\ell - p\|_2^2 - w(p) \leq \|\ell - q\|_2^2 - w(q) \ \ \forall q \in S\}$.

Let μ be a given probability measure with density p_L. Consider $S = \{-t, t\}$, with the corresponding measure $B_t = \sum_{p \in S} \frac{1}{2}\delta_p = \frac{1}{2}\delta_{-t} + \frac{1}{2}\delta_t$. Let $w^*(-t) = 2t \operatorname{med}_{p_L}(L)$, and $w^*(t) = -2t \operatorname{med}_{p_L}(t)$. We have

$$\begin{aligned}
\mathrm{Vor}_S^{w^*}(-t) &= \{\ell : \|\ell + t\|_2^2 - w^*(-t) \leq \|\ell - q\|_2^2 - w^*(q) \ \ \forall q \in \{-t, t\}\} \\
&= \{\ell : \|\ell + t\|_2^2 - w^*(-t) \leq \|\ell - t\|_2^2 - w^*(t)\} \\
&= \{\ell : \|\ell + t\|_2^2 - 2t \operatorname{med}_{p_L}(L) \leq \|\ell - t\|_2^2 + 2t \operatorname{med}_{p_L}(L)\} \\
&= \{\ell : \ell \leq \operatorname{med}_{p_L}(L)\}
\end{aligned}$$

and similarly $\mathrm{Vor}_S^{w^*}(t) = \{\ell : \ell \geq \operatorname{med}_{p_L}(L)\}$. We have

$$\int_{\mathrm{Vor}_S^{w^*}(-t)} p_L(\ell) d\ell = \int_{\ell \leq \operatorname{med}_{p_L}(L)} p_L(\ell) d\ell = \frac{1}{2}$$

and similarly $\int_{\mathrm{Vor}_S^{w^*}(t)} p_L(\ell) d\ell = \frac{1}{2}$. Therefore, w^* is *adapted* to (μ, B_t). By Theorem 2 of [216], a map $T_S^{w^*} : \mathbb{R} \to \mathbb{R}$ exists which realizes an optimal transport between μ and B_t. By

[216] Theorem 1, we have

$$
\begin{aligned}
W_2^2(\mu, B_t) &= \int_{\mathrm{Vor}_S^{w^*}(-t)} \|\ell + t\|_2^2 \; p_L(\ell) d\ell + \int_{\mathrm{Vor}_S^{w^*}(t)} \|\ell - t\|_2^2 \; p_L(\ell) d\ell \\
&= \int_{\ell \leq \mathrm{med}_{p_L}(L)} \|\ell + t\|_2^2 \; p_L(\ell) d\ell + \int_{\ell \geq \mathrm{med}_{p_L}(L)} \|\ell - t\|_2^2 \; p_L(\ell) d\ell \\
&= \mathbb{E}_{p_L}[L^2] + t^2 - 2t \bigg(\int_{\ell \geq \mathrm{med}_{p_L}(L)} \ell \; p_L(\ell) d\ell - \int_{\ell \leq \mathrm{med}_{p_L}(L)} \ell \; p_L(\ell) d\ell \bigg) \\
&= \mathbb{E}_{p_L}[L^2] + t^2 - 2t \bigg(\int_{\ell \geq \mathrm{med}_{p_L}(L)} (\ell - \mathrm{med}_{p_L}(L)) \; p_L(\ell) d\ell + \int_{\ell \leq \mathrm{med}_{p_L}(L)} (\mathrm{med}_{p_L}(L) - \ell) \; p_L(\ell) d\ell \bigg) \\
&= \mathbb{E}_{p_L}[L^2] + t^2 - 2t \bigg(\int_{\ell \geq \mathrm{med}_{p_L}(L)} |\ell - \mathrm{med}_{p_L}(L)| \; p_L(\ell) d\ell + \int_{\ell \leq \mathrm{med}_{p_L}(L)} |\ell - \mathrm{med}_{p_L}(L)| \; p_L(\ell) d\ell \bigg) \\
&= \mathbb{E}_{p_L}[L^2] + t^2 - 2t \, \mathbb{E}_{p_L}[|L - \mathrm{med}_{p_L}(L)|] \qquad \square
\end{aligned}
$$

D.1.5 Proof of Corollary 3.2.1

Proof. Let $p_L \sim \mathcal{N}(\mu, \sigma^2)$. We have $\mathbb{E}_{p_L}[L^2] = \mathbb{E}_{p_L}[L]^2 + \mathrm{Var}_{p_L}(L) = \mu^2 + \sigma^2$, and $\mathbb{E}_{p_L}[|L - \mathrm{med}_{p_L}(L)|] = \mathbb{E}_{p_L}[|L - \mu|] = \sigma\sqrt{\frac{2}{\pi}}$ [217]. Hence $W_2^2(p_L, B_t) = \mathbb{E}_{p_L}[L^2] + t^2 - 2t \, \mathbb{E}_{p_L}[|L - \mathrm{med}_{p_L}(L)|] = \mu^2 + \sigma^2 + t^2 - 2\sqrt{\frac{2}{\pi}} t\sigma$. Completing the square, we have the desired result. $\square$

D.2 Experiment Details

D.2.1 Selection of Power Constraint

Recall that APM-LR minimizes an objective function consisting of a mixture of two terms, reprinted below:

$$
\pi_n(\mathcal{L}_{n-1}) = \underset{x \in \mathcal{U}_n}{\arg\min} \;\; (\mu_n^T x)^2 + \left(\sqrt{x^T \Sigma_n x} - \sqrt{\frac{2}{\pi} P_n} \right)^2 . \tag{D.6}
$$

The first term in eq. (D.6), which is independent of P_n, encourages x to lie orthogonal to the hyperplane posterior mean, μ_n. For all such x satisfying $\mu_n^T x = 0$, we have $\mathbb{E}[L_n] = \mu_n^T x =$

0 and

$$\mathbb{E}[L_n^2] = (\mu_n^T x)^2 + x^T \Sigma_n x = x^T \Sigma_n x \leq B^2 \lambda_1(\Sigma_n)$$

where expectations are taken with respect to $p_{L_n|\mathcal{L}_{n-1}}$. Therefore $P_n = B^2\lambda_1(\Sigma_n)$ is a valid power constraint for the set of examples that induce zero-mean input distributions. This set arguably contains the "best" candidate examples, since if $(\mu_n^T x)^2 \gg 0$ then the objective in eq. (D.6) will be large. For this reason we set $P_n = B^2\lambda_1(\Sigma_n)$ in our experiments, as opposed to the power constraint of $B^2\lambda_1(\mu_n\mu_n^T + \Sigma_n)$ which is valid for all examples but is loose for examples encouraged by the first term in eq. (D.6).

D.2.2 Dataset Information

In Table D.1 we describe the datasets used in our experiments. Several datasets have multiple classes: in this case, we select a two-class dataset partition by either grouping individual classes together into super-classes, or simply training on a subset of the classes. In our experiments we treat each class partition as its own dataset, and refer to each partition by a nickname. All datasets except for *clouds*, *cross*, and *horseshoe* come from the UCI Machine Learning Repository [188]; several UCI datasets have additional citations, which are listed next to their names.

Table D.1: Full dataset information

Nickname	Dataset	Class partition	# of features	# of examples
vehicle-full	Vehicle Silhouettes [218]	$Y = -1$: 'saab' or 'opel' $Y = 1$: 'bus' or 'van'	18	846
vehicle-cars	Vehicle Silhouettes [218]	$Y = -1$: 'saab' $Y = 1$: 'opel'	18	429
vehicle-transport	Vehicle Silhouettes [218]	$Y = -1$: 'bus' $Y = 1$: 'van'	18	417
letterDP	Letter Recognition	$Y = -1$: 'D' $Y = 1$: 'P'	16	1608
letterEF	Letter Recognition	$Y = -1$: 'E' $Y = 1$: 'F'	16	1543
letterIJ	Letter Recognition	$Y = -1$: 'I' $Y = 1$: 'J'	16	1502
letterMN	Letter Recognition	$Y = -1$: 'M' $Y = 1$: 'N'	16	1575
letterUV	Letter Recognition	$Y = -1$: 'U' $Y = 1$: 'V'	16	1577
letterVY	Letter Recognition	$Y = -1$: 'V' $Y = 1$: 'Y'	16	1550
austra	Australian Credit Approval	$Y = -1$: '0' $Y = 1$: '1'	14	690
wdbc	Breast Cancer Wisconsin (Diagnostic)	$Y = -1$: 'M' $Y = 1$: 'B'	30	569
clouds	Synth1 [174]	$Y = -1$: '-1' $Y = 1$: '1'	2	600
cross	Synth2 [174]	$Y = -1$: '-1' $Y = 1$: '1'	2	600
horseshoe	Synth3 [174]	$Y = -1$: '-1' $Y = 1$: '1'	2	600

D.2.3 Baseline Methods Details

Below we elaborate on the BALD and InfoGain baseline selection methods:

InfoGain We can directly approximate information gain $I(\theta; Y \mid \mathcal{L}_{n-1})$ with a Monte Carlo approximation over s samples from $p_{\theta|\mathcal{L}_{n-1}} \sim \mathcal{N}(\mu_n, \Sigma_n)$:

$$\begin{aligned} I(\theta; Y \mid \mathcal{L}_{n-1}) &= h_b(\mathbb{E}_{p_{\theta|\mathcal{L}_{n-1}}}[f(\theta^T x_n)]) - \mathbb{E}_{p_{\theta|\mathcal{L}_{n-1}}}[h_b(f(\theta^T x_n))] \\ &\approx h_b\left(\frac{1}{s}\sum_{i=1}^{s} f(\theta_i^T x_n)\right) - \frac{1}{s}\sum_{i=1}^{s} h_b\Big(f(\theta_i^T x_n)\Big) \qquad \theta_i \sim p_{\theta|\mathcal{L}_{n-1}} \\ &\approx h_b\left(\frac{1}{s}\sum_{i=1}^{s} f(\theta_i^T x_n)\right) - \frac{1}{s}\sum_{i=1}^{s} h_b\Big(f(\theta_i^T x_n)\Big) \qquad \theta_i \sim \mathcal{N}(\mu_n, \Sigma_n) \end{aligned} \tag{D.7}$$

Our "InfoGain" baseline selects the example $x_n \in \mathcal{U}_n$ that maximizes the expression in eq. (D.7), computed in $O(sd)$ time per candidate example.

BALD Consider a probit regression label distribution $p(Y = 1 \mid L) = \Phi(L)$, where Φ is the standard normal cumulative distribution function. For $p_L \sim \mathcal{N}(\mu, \sigma^2)$, [171] use a Taylor expansion in the BALD algorithm to approximate $I(p_L, \Phi(L))$ as

$$I(p_L, \Phi(L)) \approx h_b\left(\Phi\left(\frac{\mu}{\sqrt{\sigma^2+1}}\right)\right) - \frac{D\exp\left(-\frac{\mu^2}{2(\sigma^2+D^2)}\right)}{\sqrt{\sigma^2+D^2}} \tag{D.8}$$

where $D = \sqrt{\frac{\pi \ln 2}{2}}$. By equalizing derivatives at $L = 0$, we can approximate $f(L) \approx \Phi(kL)$ where $k = \sqrt{\frac{\pi}{8}}$ [180]. Define $\widetilde{L} = kL$ and note that $\widetilde{L} \sim \mathcal{N}(\widetilde{\mu}, \widetilde{\sigma}^2)$ for $\widetilde{\mu} = k\mu$ and

$\widetilde{\sigma}^2 = k^2\sigma^2$. We can then use the BALD approximation in eq. (D.8) for logistic regression:

$$\begin{aligned}
I(p_L, f(L)) &\approx I(p_L, \Phi(kL)) \\
&= h_b(\mathbb{E}_{p_L}(\Phi(kL))) - \mathbb{E}_{p_L}(h_b(\Phi(kL))) \\
&= h_b(\mathbb{E}_{p_{\widetilde{L}}}(\Phi(\widetilde{L}))) - \mathbb{E}_{p_{\widetilde{L}}}(h_b(\Phi(\widetilde{L}))) \\
&= I(p_{\widetilde{L}}, \Phi(\widetilde{L})) \\
&\approx h_b\left(\Phi\left(\frac{k\mu}{\sqrt{k^2\sigma^2+1}}\right)\right) - \frac{D\exp\left(-\frac{k^2\mu^2}{2(k^2\sigma^2+D^2)}\right)}{\sqrt{k^2\sigma^2+D^2}}
\end{aligned}$$

Approximating $p_{\theta|\mathcal{L}_{n-1}} \sim \mathcal{N}(\mu_n, \Sigma_n)$, we have $p_{L_n|\mathcal{L}_{n-1}} \sim \mathcal{N}(\mu_n^T x_n, x_n^T \Sigma_n x_n)$ and so we can approximate

$$I(p_{L_n|\mathcal{L}_{n-1}}, f(L)) \approx h_b\left(\Phi\left(\frac{k\mu_n^T x_n}{\sqrt{k^2 x_n^T \Sigma_n x_n + 1}}\right)\right) - \frac{D\exp\left(-\frac{k^2(\mu_n^T x_n)^2}{2(k^2 x_n^T \Sigma_n x_n + D^2)}\right)}{\sqrt{k^2 x_n^T \Sigma_n x_n + D^2}} \quad \text{(D.9)}$$

where $D = \sqrt{\frac{\pi \ln 2}{2}}$ and $k = \sqrt{\frac{\pi}{8}}$. Our "BALD" baseline method selects the example $x_n \in \mathcal{U}_n$ that maximizes the expression in eq. (D.9), computed in $O(d^2)$ time per candidate example.

Summary For completeness, below we summarize all selection methods used in our experiments. For any method utilizing a normal approximation to the hyperplane posterior,

let $p_{\theta|\mathcal{L}_{n-1}} \sim \mathcal{N}(\mu_n, \Sigma_n)$. Let $\widehat{\theta}_{n-1} = \mathrm{A}(\mathcal{L}_{n-1})$, $D = \sqrt{\frac{\pi \ln 2}{2}}$, and $k = \sqrt{\frac{\pi}{8}}$.

$$\textit{APM-LR}: \quad x_n = \operatorname*{arg\,min}_{x \in \mathcal{U}_n} \ (\mu_n^T x)^2 + \left(\sqrt{x^T \Sigma_n x} - \sqrt{\frac{2}{\pi} P_n} \right)^2 \tag{D.10}$$

$$\textit{Uncertainty}: \quad x_n = \operatorname*{arg\,min}_{x \in \mathcal{U}_n} \ x^T \widehat{\theta}_{n-1}$$

$$\textit{Random}: \quad \text{Select } x_n \text{ uniformly at random from } \mathcal{U}_n$$

$$\textit{MaxVar}: \quad x_n = \operatorname*{arg\,max}_{x \in \mathcal{U}_n} \ x^T \Sigma_n x$$

$$\textit{InfoGain}: \quad x_n = \operatorname*{arg\,max}_{x \in \mathcal{U}_n} \ h_b\left(\frac{1}{s} \sum_{i=1}^{s} f(\theta_i^T x_n) \right) - \frac{1}{s} \sum_{i=1}^{s} h_b\left(f(\theta_i^T x_n) \right) \quad \theta_i \sim \mathcal{N}(\mu_n, \Sigma_n)$$

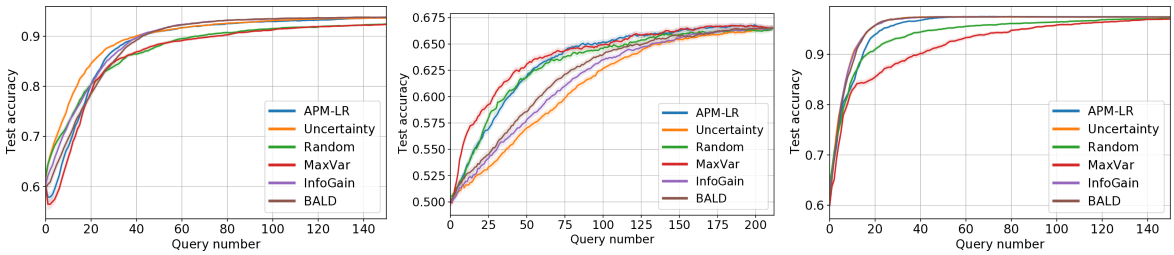
$$\textit{BALD}: \quad x_n = \operatorname*{arg\,max}_{x \in \mathcal{U}_n} \ h_b\left(\Phi\left(\frac{k \mu_n^T x_n}{\sqrt{k^2 x_n^T \Sigma_n x_n + 1}} \right) \right) - \frac{D \exp\left(-\frac{k^2 (\mu_n^T x_n)^2}{2(k^2 x_n^T \Sigma_n x_n + D^2)} \right)}{\sqrt{k^2 x_n^T \Sigma_n x_n + D^2}}$$

D.2.4 Extended Test Accuracy Results

Below we plot average holdout test accuracy against number of queried examples, excluding one initial seed point selected uniformly at random per class. Error bars show ± 1 standard error over 150 trials per method. For visual clarity, we display different numbers of queried examples for each dataset.

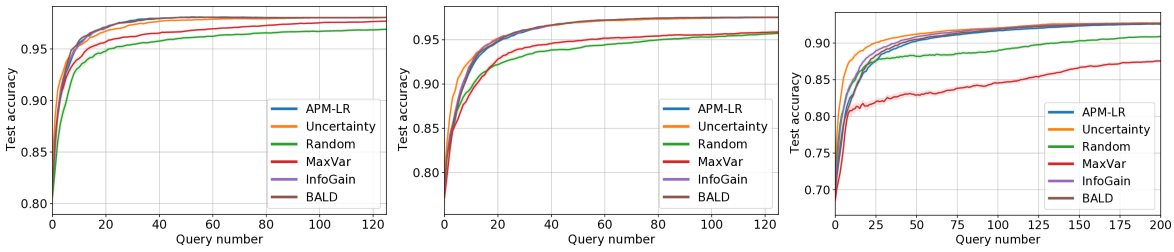
Figure D.1 shows test accuracy across several two-class partitions of the Vehicle Silhouettes dataset (see Table D.1). In *vehicle-cars*, Uncertainty, InfoGain, and BALD fail to perform as well as MaxVar, Random, and APM-LR. As noted in [174], there are cases where Random sampling — or more generally, selection methods that encourage dataset exploration — can outperform methods that maximize information. In *vehicle-cars*, it's possible that the "exploration" component in APM-LR encourages the selection of satisfactory examples, which we investigate further in Section D.2.6.

Figure D.2 shows test accuracy across several two-class partitions of the Letter Recognition dataset. All partitions show similar trends to *letterDP*, which was included in Chapter 7.

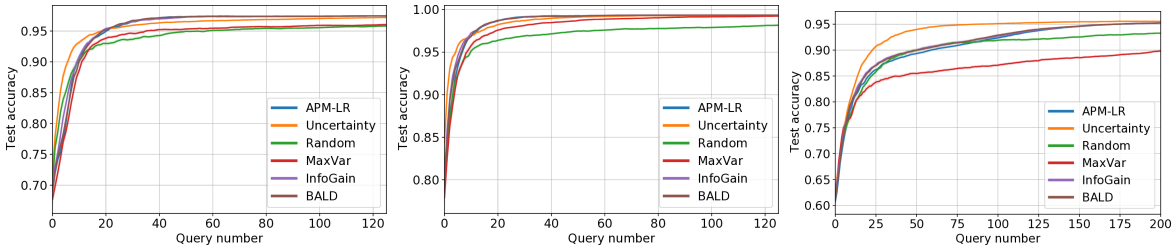


(a) *vehicle-full* (b) *vehicle-cars* (c) *vehicle-transport*

Figure D.1: Test accuracy on "Vehicle Silhouettes"



(a) *letterDP* (b) *letterEF* (c) *letterIJ*

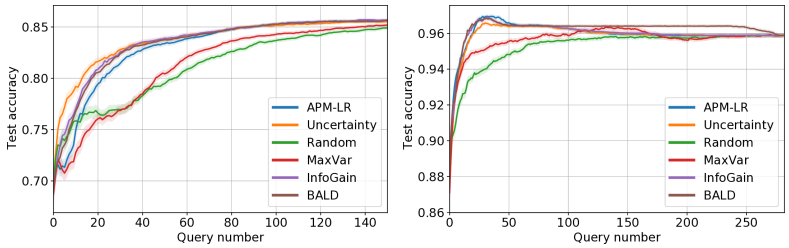


(d) *letterMN* (e) *letterUV* (f) *letterVY*

Figure D.2: Test accuracy on "Letter Recognition"

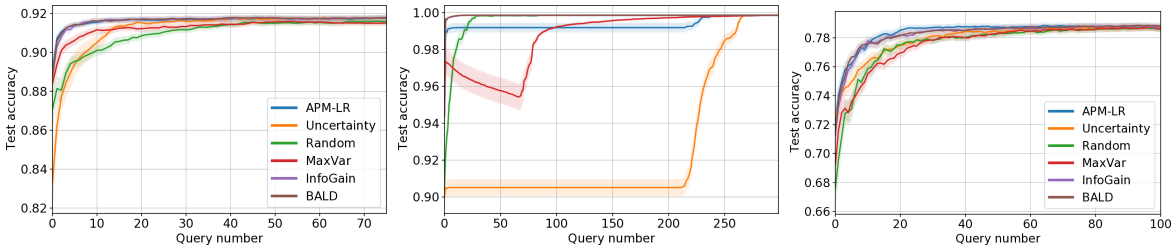
Figure D.3 shows test accuracy across the remaining UCI datasets in Table D.1. On *wdbc*, the active methods appear to have an average test accuracy that peaks early and then gradually decreases. While this behavior merits further investigation, we note that it is possible in some cases for a selected subset of the full data pool to generalize better than when training on the entire pool [219].

Figure D.4 shows test accuracy across several synthetic datasets. On *clouds* and *cross*, Uncertainty sampling is outperformed by the other baseline active learning methods, except MaxVar.



(a) *austra* (b) *wdbc*

Figure D.3: Miscellaneous UCI datasets



(a) *clouds* (b) *cross* (c) *horseshoe*

Figure D.4: Synthetic datasets

D.2.5 Extended Computational Cost Results

All experiments were run on Intel Xeon Gold 6226 CPUs at 2.7 GHz. In Table D.2 we present for all datasets the cumulative compute time (in seconds) needed for each method to select the first 40 examples (excluding seed points). In this first table, we exclude the compute time needed to retrain the logistic regression model and perform the VariationalEM posterior update after each example is selected, since these steps are common to all selection methods. While some methods do not directly utilize the variational posterior in selecting examples, we perform variational posterior updates for all data selection methods since we consider the variational posterior to be part of the Bayesian model produced by the training routine.

Table D.3 isolates the compute time needed for performing VariationalEM at each input, summed over the first 40 examples. Interestingly, methods which are primarily focused on data space exploration (MaxVar, Random) require more time for variational posterior updating than exploitation methods (Uncertainty). Since VariationalEM is an

Table D.2: **Cumulative selection time:** comparison of median cumulative time (s) for each method to select the first 40 examples (excluding seed points).

	APM-LR	Uncertainty	BALD	InfoGain	Random	MaxVar
vehicle-full	0.173	0.077	2.212	7.166	0.003	0.061
vehicle-cars	0.089	0.039	1.078	3.462	0.002	0.030
vehicle-transport	0.087	0.037	1.036	3.306	0.002	0.029
letterDP	0.336	0.149	4.230	12.755	0.005	0.118
letterEF	0.318	0.143	4.044	12.188	0.005	0.113
letterIJ	0.314	0.139	3.941	11.879	0.004	0.110
letterMN	0.331	0.147	4.170	12.531	0.005	0.117
letterUV	0.330	0.145	4.129	12.429	0.005	0.115
letterVY	0.318	0.143	4.063	12.284	0.004	0.114
austra	0.150	0.063	1.770	5.089	0.003	0.050
wdbc	0.125	0.052	1.480	6.415	0.003	0.042
clouds	0.119	0.053	1.522	2.735	0.002	0.041
cross	0.125	0.053	1.521	2.722	0.002	0.040
horseshoe	0.116	0.053	1.517	2.731	0.002	0.040

iterative procedure that we run with an adaptive stopping rule (with convergence defined as the relative variational parameter difference falling below $1\mathrm{e}{-6}$ between iterations), it presumably requires more iterations to adjust to significant changes in the posterior distribution due to variability in examples. Although less accurate of an approximation than VariationalEM, using a Laplace posterior approximation instead would have a constant update time per method [187].

Table D.4 depicts the total compute time needed for selecting each example, performing VariationalEM, and retraining the logistic regression classifier at each iteration, summed over the first 40 examples. The median time needed for retraining the logistic regression classifier lies within 0.01 to 0.03 seconds across all methods and datasets, and therefore contributes only marginally to the total. While the spread of running times is more narrow than it would be when only evaluating selection time, the same general trend holds that InfoGain is more expensive than BALD and APM-LR.

Table D.3: **Cumulative VariationalEM time:** comparison of median cumulative time (s) for each method to perform VariationalEM over the first 40 examples (excluding seed points).

	APM-LR	Uncertainty	BALD	InfoGain	Random	MaxVar
vehicle-full	10.088	4.540	10.118	9.729	7.469	18.064
vehicle-cars	5.420	4.412	5.605	5.475	3.280	4.558
vehicle-transport	9.609	5.814	9.289	9.083	11.216	21.058
letterDP	7.618	6.412	6.904	6.758	10.694	11.851
letterEF	6.866	5.701	6.320	6.160	11.302	10.755
letterIJ	7.367	5.724	6.924	6.708	10.019	9.846
letterMN	8.190	6.281	7.615	7.375	10.082	13.236
letterUV	8.029	6.556	7.137	7.075	10.746	12.585
letterVY	7.463	5.760	7.142	6.910	8.234	9.975
austra	12.513	6.451	12.009	11.645	8.541	13.580
wdbc	17.966	10.880	14.183	13.874	20.763	29.778
clouds	1.221	1.172	1.156	1.322	3.201	5.318
cross	1.386	2.417	1.474	1.537	3.138	4.453
horseshoe	0.996	0.908	0.863	0.931	0.802	1.208

D.2.6 Failure Mode Analysis

While in many cases APM-LR performs comparably to InfoGain, BALD, and Uncertainty while outperforming Random and MaxVar, the main exception in our experiments is on *vehicle-cars* (Figure D.1b), where APM-LR, Random, and MaxVar outperform InfoGain, BALD, and Uncertainty. Conceptually, what differentiates these two classes of methods is that APM-LR, Random, and MaxVar have explicit exploration components to their selection policies, while InfoGain, BALD, and Uncertainty only seek to directly maximize information or uncertainty. As we will demonstrate below, on *vehicle-cars* this difference in exploration correlates with significant differences in generalization performance.

To isolate the effect of each term in APM-LR (eq. (D.10)) — corresponding to exploitation and exploration — we simulated two pseudo-APM policies where only one of the terms is active at once. In *APM-LR-U*, examples are selected that minimize the first term, which

Table D.4: **Cumulative running time:** comparison of median cumulative run time (s) for each method to select each example, perform VariationalEM, and retrain the logistic regression classifier over the first 40 examples (excluding seed points).

	APM-LR	Uncertainty	BALD	InfoGain	Random	MaxVar
vehicle-full	10.288	4.637	12.365	16.943	7.493	18.148
vehicle-cars	5.532	4.474	6.727	8.980	3.306	4.616
vehicle-transport	9.721	5.876	10.341	12.419	11.238	21.116
letterDP	7.992	6.583	11.139	19.534	10.730	11.995
letterEF	7.215	5.868	10.396	18.414	11.330	10.887
letterIJ	7.716	5.892	10.896	18.619	10.048	9.981
letterMN	8.561	6.455	11.813	19.991	10.124	13.374
letterUV	8.399	6.724	11.294	19.552	10.781	12.724
letterVY	7.802	5.931	11.233	19.233	8.260	10.118
austra	12.690	6.538	13.801	16.804	8.574	13.655
wdbc	18.130	10.968	15.711	20.323	20.787	29.842
clouds	1.358	1.241	2.706	4.122	3.224	5.385
cross	1.534	2.490	3.028	4.291	3.159	4.515
horseshoe	1.134	0.978	2.405	3.741	0.819	1.264

has an action similar to uncertainty sampling:

$$\textit{APM-LR-U}: \quad x_n = \operatorname*{arg\,min}_{x \in \mathcal{U}_n} \ (\mu_n^T x)^2.$$

In *APM-LR-V*, examples are selected that minimize the second term, which prefers examples that probe in directions of high posterior variance:

$$\textit{APM-LR-V}: \quad x_n = \operatorname*{arg\,min}_{x \in \mathcal{U}_n} \ \left(\sqrt{x^T \Sigma_n x} - \sqrt{\frac{2}{\pi} P_n} \right)^2.$$

We start in Figure D.5 by plotting generalization performance as in Figure D.1b, with the addition of APM-LR-U and APM-LR-V. In all plots below, error bars are removed for visual clarity, and the query horizon spans the entire training sequence (until the training pool is exhausted). As expected, APM-LR-V performs comparably to MaxVar, since both methods prefer examples that probe in directions of large posterior variance. Similarly, APM-LR-U performs comparably to Uncertainty, since both methods minimize distance to a hyperplane estimate (the former using the posterior mean hyperplane, the latter using a

MAP estimate). These results support the hypothesis that it is the exploration component of APM-LR which leads to improved performance on *vehicle-cars* over non-exploration methods, including its own exploitation variant APM-LR-U.

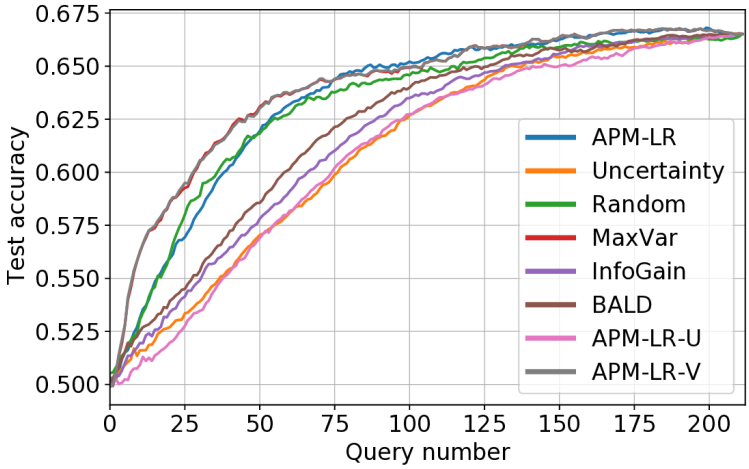


Figure D.5: Test accuracy on *vehicle-cars*, over expanded method set.

We can explore this hypothesis further by directly evaluating metrics for exploitation and exploration of each method. To measure exploitation, in Figure D.6, we plot the average distance from each selected example to the MAP hyperplane estimate. Since distance from the classifier hyperplane directly corresponds to label uncertainty in logistic regression, this distance is a direct measure of how often a policy selects uncertain examples. By definition, Uncertainty begins by querying examples that are closest to the hyperplane estimate, maximally exploiting the estimate to query examples with the highest model uncertainty. The remaining methods vary in their levels of initial distance from the hyperplane estimate, but all eventually query close to their respective estimates, either by design or due to exhausting the full training pool. Notably, the level of initial distance from the hyperplane corresponds almost exactly to test accuracy performance: high-performing MaxVar and APM-LR-V initially query far from their hyperplane estimates, while the poorly performing Uncertainty queries examples close by.

To measure policy exploration, we use two metrics and plot their average values in Figure D.7. In the first metric, we measure the Euclidean distance from each unlabeled

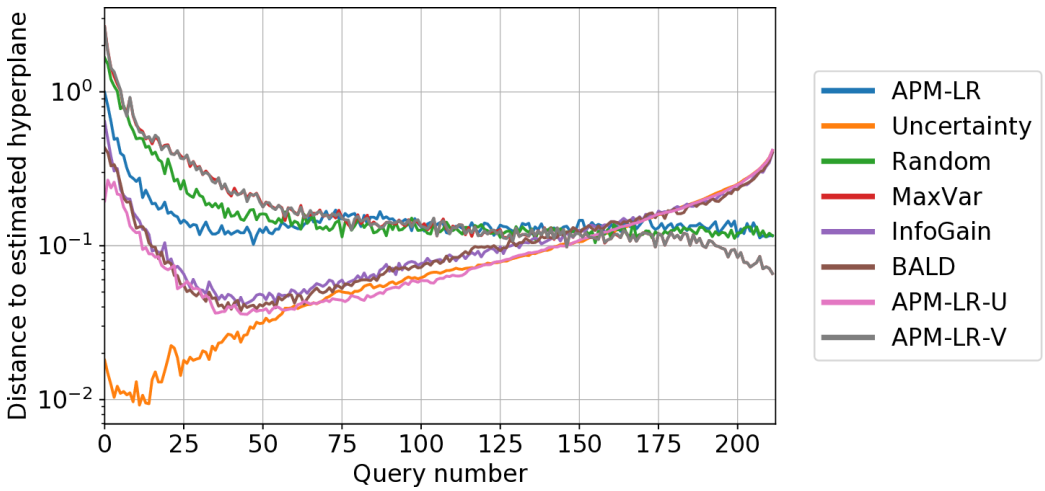
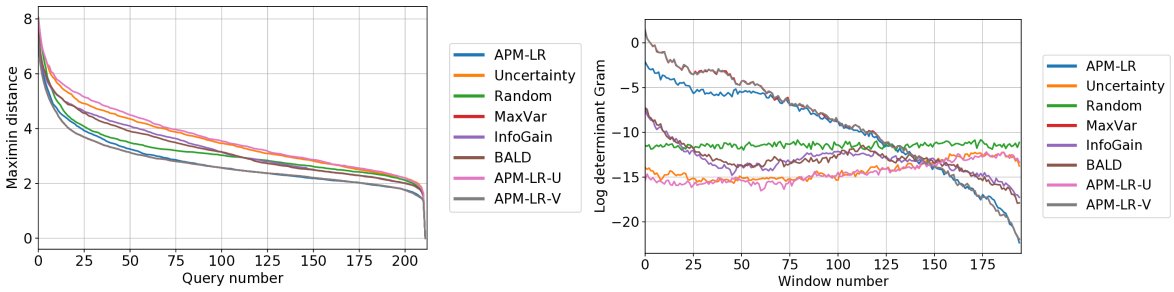


Figure D.6: Exploitation metric for *vehicle-cars*: average distance of selected example to estimated hyperplane. Small distances reflect high levels of policy exploitation since this reflects examples being queried that are uncertain with respect to the current hyperplane estimate.

example to its nearest labeled neighbor, and take the maximum such distance over all unlabeled examples. This quantity measures the worst-case level of isolation of an unlabeled point to its nearest labeled neighbor, with lower values corresponding to higher degrees of policy exploration. A similar quantity is involved in the construction of coresets for active learning to promote diversity among selected examples [45]. As our second metric, we consider windows of d examples (recall that d denotes the data space dimension) and plot the log determinant of the Gram matrix of the examples selected in each window, which can be used as a measure of example diversity (higher values correspond to higher levels of example diversity) [220]. In Figure D.7a, MaxVar, APM-LR-V, APM-LR, and Random have the lowest average maximin distances, corresponding to lower levels of isolated unlabeled examples. Similarly, these methods generally have large initial Gram matrix log determinants, as depicted in Figure D.7b.

The ablation of individual terms in APM-LR and direct measurement of exploitation and exploration of each active learning method suggests that when tested on *vehicle-cars*, exploration-based methods outperform methods that do not explicitly optimize for diverse selection. While this extended analysis is limited to a single dataset, it provides evidence that



(a) (b)

Figure D.7: Exploration metrics for *vehicle-cars*: (a) maximum distance from an unlabeled example to its closest labeled example. Smaller values indicate lower levels of unlabeled data isolation, and correspond to higher levels of exploration. (b) Log determinant of Gram matrix, where larger values correspond to higher levels of exploration.

the exploration term in APM-LR can lead to higher levels of performance on a real-world dataset, where methods that do not directly account for exploration might fail.

REFERENCES

[1] G. Canal, Y. Diaz-Mercado, M. Egerstedt, and C. Rozell, "A low-complexity brain-computer interface for high-complexity robot swarm control," *Submitted*, 2021.

[2] G. Canal, S. Manivasagam, S. Liang, and C. Rozell, "Interactive object segmentation with noisy binary inputs," in *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, © 2018 IEEE, 2018, pp. 405–409.

[3] G. Canal, S. Fenu, and C. Rozell, "Active ordinal querying for tuplewise similarity learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 3332–3340, Apr. 2020.

[4] G. Canal, A. Massimino, M. Davenport, and C. Rozell, "Active embedding search via noisy paired comparisons," in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, Sep. 2019, pp. 902–911.

[5] G. H. Canal, M. R. O'Shaughnessy, C. J. Rozell, and M. A. Davenport, "Joint estimation of trajectory and dynamics from paired comparisons," in *2019 IEEE 8th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, © 2019 IEEE, 2019, pp. 121–125.

[6] G. Canal, M. Bloch, and C. Rozell, "Feedback coding for active learning," in *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, A. Banerjee and K. Fukumizu, Eds., ser. Proceedings of Machine Learning Research, vol. 130, PMLR, 13–15 Apr 2021, pp. 1468–1476.

[7] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[8] H. Pham, Z. Dai, Q. Xie, M.-T. Luong, and Q. V. Le, *Meta pseudo labels*, 2021. arXiv: 2003.10580 [cs.LG].

[9] Y. Liu, "Active learning with support vector machine applied to gene expression data for cancer classification," *Journal of Chemical Information and Computer Sciences*, vol. 44, no. 6, pp. 1936–1941, 2004, PMID: 15554662. eprint: https://doi.org/10.1021/ci049810a.

[10] M. K. Warmuth, J. Liao, G. Rätsch, M. Mathieson, S. Putta, and C. Lemmen, "Active learning with support vector machines in the drug discovery process," *Journal of Chemical Information and Computer Sciences*, vol. 43, no. 2, pp. 667–673, 2003, PMID: 12653536. eprint: https://doi.org/10.1021/ci025620t.

[11] D. Reker and G. Schneider, "Active-learning strategies in computer-assisted drug discovery," *Drug Discovery Today*, vol. 20, no. 4, pp. 458–465, 2015.

[12] T. Wang, J.-Y. Zhu, A. Torralba, and A. A. Efros, *Dataset distillation*, 2020. arXiv: 1811.10959 [cs.LG].

[13] "Interactive learning," Foundations of Machine Learning Program, Simons Institute for the Theory of Computing, 2017.

[14] X. Zhu, A. Singla, S. Zilles, and A. N. Rafferty, *An overview of machine teaching*, 2018. arXiv: 1801.05927 [cs.LG].

[15] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[16] G. A. Miller, "The magical number seven, plus or minus two: Some limits on our capacity for processing information.," *Psychological review*, vol. 63, no. 2, p. 81, 1956.

[17] A. Xu and M. Raginsky, "Information-theoretic analysis of generalization capability of learning algorithms," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., 2017, pp. 2524–2533.

[18] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29, Curran Associates, Inc., 2016.

[19] Y. Chen, S. H. Hassani, A. Karbasi, and A. Krause, "Sequential information maximization: When is greedy near-optimal?" In *Proceedings of The 28th Conference on Learning Theory*, P. Grünwald, E. Hazan, and S. Kale, Eds., ser. Proceedings of Machine Learning Research, vol. 40, Paris, France: PMLR, Mar. 2015, pp. 338–363.

[20] D. J. C. MacKay, "Information-Based Objective Functions for Active Data Selection," *Neural Computation*, vol. 4, no. 4, pp. 590–604, Jul. 1992. eprint: https://direct.mit.edu/neco/article-pdf/4/4/590/812354/neco.1992.4.4.590.pdf.

[21] W. Gao, S. Kannan, S. Oh, and P. Viswanath, "Estimating mutual information for discrete-continuous mixtures," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc., 2017.

[22] O. Shayevitz and M. Feder, "Optimal feedback communication via posterior matching," *IEEE Transactions on Information Theory*, vol. 57, no. 3, pp. 1186–1222, 2011.

[23] C. Omar, A. Akce, M. Johnson, T. Bretl, R. Ma, E. Maclin, M. McCormick, and T. P. Coleman, "A feedback information-theoretic approach to the design of brain–computer interfaces," *International Journal of Human–Computer Interaction*, vol. 27, no. 1, pp. 5–23, 2010. eprint: https://doi.org/10.1080/10447318.2011.535749.

[24] L. Lovász and S. Vempala, "The geometry of logconcave functions and sampling algorithms," *Random Structures & Algorithms*, vol. 30, no. 3, pp. 307–358, 2007. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rsa.20135.

[25] A. Saumard and J. A. Wellner, "Log-concavity and strong log-concavity: A review," *Statistics surveys*, vol. 8, pp. 45–114, 2014, PMC4847755[pmcid].

[26] J. R. Pierce, *An introduction to information theory: symbols, signals and noise*. Courier Corporation, 2012.

[27] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. USA: Wiley-Interscience, 2006, ISBN: 0471241954.

[28] C. Shannon, "The zero error capacity of a noisy channel," *IRE Transactions on Information Theory*, vol. 2, no. 3, pp. 8–19, 1956.

[29] M. Horstein, "Sequential transmission using noiseless feedback," *IEEE Transactions on Information Theory*, vol. 9, no. 3, pp. 136–143, 1963.

[30] J. Schalkwijk and T. Kailath, "A coding scheme for additive noise channels with feedback–i: No bandwidth constraint," *IEEE Transactions on Information Theory*, vol. 12, no. 2, pp. 172–182, 1966.

[31] R. Ma and T. P. Coleman, "Generalizing the posterior matching scheme to higher dimensions via optimal transportation," in *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2011, pp. 96–102.

[32] T. P. Coleman, "A stochastic control viewpoint on 'posterior matching'-style feedback communication schemes," in *2009 IEEE International Symposium on Information Theory*, 2009, pp. 1520–1524.

[33] B. Jedynak, P. I. Frazier, and R. Sznitman, "Twenty questions with noise: Bayes optimal policies for entropy loss," *Journal of Applied Probability*, vol. 49, no. 1, pp. 114–136, 2012.

[34] T. Tsiligkaridis, B. M. Sadler, and A. O. Hero, "Collaborative 20 questions for target localization," *IEEE Transactions on Information Theory*, vol. 60, no. 4, pp. 2233–2252, 2014.

[35] R. Castro and R. Nowak, "Active learning and sampling," in *Foundations and Applications of Sensor Management*, A. O. Hero, D. A. Castañón, D. Cochran, and K. Kastella, Eds. Boston, MA: Springer US, 2008, pp. 177–200, ISBN: 978-0-387-49819-5.

[36] R. Waeber, P. I. Frazier, and S. G. Henderson, "Bisection search with noisy responses," *SIAM Journal on Control and Optimization*, vol. 51, no. 3, pp. 2261–2279, 2013. eprint: https://doi.org/10.1137/120861898.

[37] R. D. Nowak, "The geometry of generalized binary search," *IEEE Transactions on Information Theory*, vol. 57, no. 12, pp. 7893–7906, 2011.

[38] M. V. Burnashev and K. S. Zigangirov, "An interval estimation problem for controlled observations," *Problems of Information Transmission*, vol. 10, no. 3, pp. 223–231, 1974.

[39] B. Settles, "Active learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6, no. 1, pp. 1–114, 2012. eprint: https://doi.org/10.2200/S00429ED1V01Y201207AIM018.

[40] G. A. Hollinger, B. Englot, F. S. Hover, U. Mitra, and G. S. Sukhatme, "Active planning for underwater inspection and the benefit of adaptivity," *The International Journal of Robotics Research*, vol. 32, no. 1, pp. 3–18, 2013. eprint: https://doi.org/10.1177/0278364912467485.

[41] D. V. Lindley, "On a measure of the information provided by an experiment," *The Annals of Mathematical Statistics*, vol. 27, no. 4, pp. 986–1005, 1956.

[42] H. Chernoff, "Sequential design of experiments," *The Annals of Mathematical Statistics*, vol. 30, no. 3, pp. 755–770, 1959.

[43] K. Chaloner and I. Verdinelli, "Bayesian experimental design: A review," *Statistical Science*, vol. 10, no. 3, pp. 273–304, 1995.

[44] R. Pinsler, J. Gordon, E. Nalisnick, and J. M. Hernández-Lobato, "Bayesian batch active learning as sparse subset approximation," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019.

[45] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach," in *International Conference on Learning Representations*, 2018.

[46] S. Sinha, S. Ebrahimi, and T. Darrell, "Variational adversarial active learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019.

[47] W. H. Beluch, T. Genewein, A. Nürnberger, and J. M. Köhler, "The power of ensembles for active learning in image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018.

[48] Y. Gal, R. Islam, and Z. Ghahramani, "Deep Bayesian active learning with image data," D. Precup and Y. W. Teh, Eds., ser. Proceedings of Machine Learning Research, vol. 70, International Convention Centre, Sydney, Australia: PMLR, Jun. 2017, pp. 1183–1192.

[49] A. Kirsch, J. van Amersfoort, and Y. Gal, "Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019.

[50] J. Tantiongloc, D. A. Mesa, R. Ma, S. Kim, C. H. Alzate, J. J. Camacho, V. Manian, and T. P. Coleman, "An information and control framework for optimizing user-compliant human–computer interfaces," *Proceedings of the IEEE*, vol. 105, no. 2, pp. 273–285, 2017.

[51] A. Akce, M. Johnson, O. Dantsker, and T. Bretl, "A brain–machine interface to navigate a mobile robot in a planar workspace: Enabling humans to fly simulated aircraft with eeg," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 21, no. 2, pp. 306–318, 2013.

[52] A. Akce, M. Johnson, and T. Bretl, "Remote teleoperation of an unmanned aircraft with a brain-machine interface: Theory and preliminary results," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 5322–5327.

[53] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain–computer interfaces for communication and control," *Clinical Neurophysiology*, vol. 113, no. 6, pp. 767–791, 2002.

[54] S. R. Soekadar, M. Witkowski, C. Gómez, E. Opisso, J. Medina, M. Cortese, M. Cempini, M. C. Carrozza, L. G. Cohen, N. Birbaumer, and N. Vitiello, "Hybrid eeg/eog-based brain/neural hand exoskeleton restores fully independent daily living activities after quadriplegia," *Science Robotics*, vol. 1, no. 1, 2016. eprint: https://robotics.sciencemag.org/content/1/1/eaag3296.full.pdf.

[55] J. L. Collinger, B. Wodlinger, J. E. Downey, W. Wang, E. C. Tyler-Kabara, D. J. Weber, A. J. McMorland, M. Velliste, M. L. Boninger, and A. B. Schwartz, "High-

performance neuroprosthetic control by an individual with tetraplegia," *The Lancet*, vol. 381, no. 9866, pp. 557–564, 2013.

[56] L. R. Hochberg, D. Bacher, B. Jarosiewicz, N. Y. Masse, J. D. Simeral, J. Vogel, S. Haddadin, J. Liu, S. S. Cash, P. van der Smagt, and J. P. Donoghue, "Reach and grasp by people with tetraplegia using a neurally controlled robotic arm," *Nature*, vol. 485, no. 7398, pp. 372–375, May 2012.

[57] L. R. Hochberg, M. D. Serruya, G. M. Friehs, J. A. Mukand, M. Saleh, A. H. Caplan, A. Branner, D. Chen, R. D. Penn, and J. P. Donoghue, "Neuronal ensemble control of prosthetic devices by a human with tetraplegia," *Nature*, vol. 442, no. 7099, pp. 164–171, Jul. 2006.

[58] B. Wodlinger, J. E. Downey, E. C. Tyler-Kabara, A. B. Schwartz, M. L. Boninger, and J. L. Collinger, "Ten-dimensional anthropomorphic arm control in a human brain-machine interface: Difficulties, solutions, and limitations," *Journal of Neural Engineering*, vol. 12, no. 1, p. 016 011, Dec. 2014.

[59] R. Leeb, D. Friedman, G. R. Muller-Putz, R. Scherer, M. Slater, and G. Pfurtscheller, "Self-paced (asynchronous) bci control of a wheelchair in virtual environments: A case study with a tetraplegic," *Computational intelligence and neuroscience*, vol. 2007, pp. 79 642–8, 2007.

[60] F. Galán, M. Nuttin, E. Lew, P. Ferrez, G. Vanacker, J. Philips, and J. d. R. Millán, "A brain-actuated wheelchair: Asynchronous and non-invasive brain–computer interfaces for continuous control of robots," *Clinical neurophysiology*, vol. 119, no. 9, pp. 2159–2169, 2008.

[61] I. Iturrate, J. M. Antelis, A. Kubler, and J. Minguez, "A noninvasive brain-actuated wheelchair based on a p300 neurophysiological protocol and automated navigation," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 614–627, 2009.

[62] B. Rebsamen, E. Burdet, C. Guan, H. Zhang, C. L. Teo, Q. Zeng, C. Laugier, and M. H. Ang, "Controlling a wheelchair indoors using thought," *IEEE Intelligent Systems*, vol. 22, no. 2, pp. 18–24, 2007.

[63] D. Huang, K. Qian, D. Fei, W. Jia, X. Chen, and O. Bai, "Electroencephalography (eeg)-based brain–computer interface (bci): A 2-d virtual wheelchair control based on event-related desynchronization/synchronization and state control," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 20, no. 3, pp. 379–388, 2012.

[64] Y. Li, J. Pan, F. Wang, and Z. Yu, "A hybrid bci system combining p300 and ssvep and its application to wheelchair control," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 11, pp. 3156–3166, 2013.

[65] J. Long, Y. Li, H. Wang, T. Yu, J. Pan, and F. Li, "A hybrid brain computer interface to control the direction and speed of a simulated or real wheelchair," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 20, no. 5, pp. 720–729, 2012.

[66] J. Li, J. Liang, Q. Zhao, J. Li, K. Hong, and L. Zhang, "Design of assistive wheelchair system directly steered by human thoughts," *International journal of neural systems*, vol. 23 3, p. 1 350 013, 2013.

[67] T. Carlson and J. del R. Millan, "Brain-controlled wheelchairs: A robotic architecture," *IEEE Robotics Automation Magazine*, vol. 20, no. 1, pp. 65–73, 2013.

[68] R. Zhang, Y. Li, Y. Yan, H. Zhang, S. Wu, T. Yu, and Z. Gu, "Control of a wheelchair in an indoor environment based on a brain–computer interface and automated navigation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 1, pp. 128–139, 2016.

[69] S. Mueller, W. Cardoso, T. Freire, and M. Sarcinelli-Filho, "Brain-computer interface based on visual evoked potentials to command autonomous robotic wheelchair," *J. Med. Biol. Eng*, vol. 30, Jan. 2010.

[70] B. Rebsamen, C. Guan, H. Zhang, C. Wang, C. Teo, M. H. Ang, and E. Burdet, "A brain controlled wheelchair to navigate in familiar environments," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 18, no. 6, pp. 590–598, 2010.

[71] B. Blankertz, G. Dornhege, M. Krauledat, K.-R. Müller, and G. Curio, "The non-invasive berlin brain–computer interface: Fast acquisition of effective performance in untrained subjects," *NeuroImage*, vol. 37, no. 2, pp. 539–550, 2007.

[72] J. R. Wolpaw, D. J. McFarland, and E. Bizzi, "Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 101, no. 51, pp. 17 849–17 854, 2004.

[73] B. Xia, D. An, C. Chen, H. Xie, and J. Li, "A mental switch-based asynchronous brain-computer interface for 2d cursor control," in *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2013, pp. 3101–3104.

[74] D. J. McFarland, W. A. Sarnacki, and J. R. Wolpaw, "Electroencephalographic (eeg) control of three-dimensional movement," *Journal of neural engineering*, vol. 7, no. 3, pp. 036 007–036 007, 2010.

[75] J. Long, Y. Li, T. Yu, and Z. Gu, "Target selection with hybrid feature for bci-based 2-d cursor control," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 1, pp. 132–140, 2012.

[76] L. J. Trejo, R. Rosipal, and B. Matthews, "Brain-computer interfaces for 1-d and 2-d cursor control: Designs using volitional control of the eeg spectrum or steady-state visual evoked potentials," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 14, no. 2, pp. 225–229, 2006.

[77] *Imagining a new interface: Hands-free communication without saying a word*, https://tech.fb.com/imagining-a-new-interface-hands-free-communication-without-saying-a-word/, 2020.

[78] E. Musk, "An integrated brain-machine interface platform with thousands of channels," *J Med Internet Res*, vol. 21, no. 10, e16194, Oct. 2019.

[79] *Nonsurgical neural interfaces could significantly expand use of neurotechnology*, https://www.darpa.mil/news-events/2018-03-16, 2018.

[80] E. J. Pratt, M. Ledbetter, R. Jiménez-Martínez, B. Shapiro, A. Solon, G. Z. Iwata, S. Garber, J. Gormley, D. Decker, D. Delgadillo, A. T. Dellis, J. Phillips, G. Sundar, J. Leung, J. Coyne, M. McKinley, G. Lopez, S. Homan, L. Marsh, M. Zhang, V. Maurice, B. Siepser, T. Giovannoli, B. Leverett, G. Lerner, S. Seidman, V. DeLuna, K. Wright-Freeman, J. Kates-Harbeck, T. Lasser, H. Mohseni, T. Sharp, A. Zorzos, A. H. Lara, A. Kouhzadi, A. Ojeda, P. Chopra, Z. Bednarke, M. Henninger, and J. K. Alford, "Kernel Flux: a whole-head 432-magnetometer optically-pumped magnetoencephalography (OP-MEG) system for brain activity imaging during natural human experiences," in *Optical and Quantum Sensing and Precision Metrology*, S. M. Shahriar and J. Scheuer, Eds., International Society for Optics and Photonics, vol. 11700, SPIE, 2021, pp. 162–179.

[81] R. Scherer, G. Muller, C. Neuper, B. Graimann, and G. Pfurtscheller, "An asynchronously controlled eeg-based virtual keyboard: Improvement of the spelling rate," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 6, pp. 979–984, 2004.

[82] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kübler, J. Perelmouter, E. Taub, and H. Flor, "A spelling device for the paralysed," *Nature*, vol. 398, no. 6725, pp. 297–298, Mar. 1999.

[83] A. Kübler, A. Furdea, S. Halder, E. M. Hammer, F. Nijboer, and B. Kotchoubey, "A brain–computer interface controlled auditory event-related potential (p300) spelling system for locked-in patients," *Annals of the New York Academy of Sciences*, vol. 1157, no. 1, pp. 90–100, 2009. eprint: https://nyaspubs.onlinelibrary.wiley.com/doi/pdf/10.1111/j.1749-6632.2008.04122.x.

[84] M. Cheng, X. Gao, S. Gao, and D. Xu, "Design and implementation of a brain-computer interface with high transfer rates," *IEEE Transactions on Biomedical Engineering*, vol. 49, no. 10, pp. 1181–1186, 2002.

[85] J. Meng, S. Zhang, A. Bekyo, J. Olsoe, B. Baxter, and B. He, "Noninvasive electroencephalogram based control of a robotic arm for reach and grasp tasks," *Scientific Reports*, vol. 6, no. 1, p. 38 565, Dec. 2016.

[86] B. J. Edelman, J. Meng, D. Suma, C. Zurn, E. Nagarajan, B. S. Baxter, C. C. Cline, and B. He, "Noninvasive neuroimaging enhances continuous neural tracking for robotic device control," *Science Robotics*, vol. 4, no. 31, 2019. eprint: https://robotics.sciencemag.org/content/4/31/eaaw6844.full.pdf.

[87] K. LaFleur, K. Cassady, A. Doud, K. Shades, E. Rogin, and B. He, "Quadcopter control in three-dimensional space using a noninvasive motor imagery-based brain–computer interface," *Journal of Neural Engineering*, vol. 10, no. 4, p. 046 003, Jun. 2013.

[88] G. K. Karavas, D. T. Larsson, and P. Artemiadis, "A hybrid bmi for control of robotic swarms: Preliminary results," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5065–5075.

[89] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010, vol. 33.

[90] Y. Diaz-Mercado, S. G. Lee, and M. Egerstedt, "Distributed dynamic density coverage for human-swarm interactions," in *2015 American Control Conference (ACC)*, 2015, pp. 353–358.

[91] X. Xu and Y. Diaz-Mercado, "Multi-agent control using coverage over time-varying domains," in *2020 American Control Conference (ACC)*, IEEE, 2020, pp. 2030–2035.

[92] G. Pfurtscheller and C. Neuper, "Motor imagery activates primary sensorimotor area in humans," *Neuroscience Letters*, vol. 239, no. 2, pp. 65–68, 1997.

[93] B. Blankertz, R. Tomioka, S. Lemm, M. Kawanabe, and K.-r. Muller, "Optimizing spatial filters for robust eeg single-trial analysis," *IEEE Signal Processing Magazine*, vol. 25, no. 1, pp. 41–56, 2008.

[94] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, and B. Arnaldi, "A review of classification algorithms for EEG-based brain–computer interfaces," *Journal of Neural Engineering*, vol. 4, no. 2, R1–R13, Jan. 2007.

[95] L. D. Brown, T. T. Cai, and A. DasGupta, "Interval estimation for a binomial proportion," *Statistical Science*, vol. 16, no. 2, pp. 101–117, 2001.

[96] C. Omar, M. Johnson, T. W. Bretl, and T. P. Coleman, "Querying the user properly for high-performance brain-machine interfaces: Recursive estimation, control, and feedback information-theoretic perspectives," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2008, pp. 5216–5219.

[97] F. Dubost, L. Peter, C. Rupprecht, B. G. Becker, and N. Navab, "Hands-free segmentation of medical volumes via binary inputs," in *Deep Learning and Data Labeling for Medical Applications*, G. Carneiro, D. Mateus, L. Peter, A. Bradley, J. M. R. S. Tavares, V. Belagiannis, J. P. Papa, J. C. Nascimento, M. Loog, Z. Lu, J. S. Cardoso, and J. Cornebise, Eds., Cham: Springer International Publishing, 2016, pp. 259–268, ISBN: 978-3-319-46976-8.

[98] M. Sadeghi, G. Tien, G. Hamarneh, and M. S. A. M.D., "Hands-free interactive image segmentation using eyegaze," in *Medical Imaging 2009: Computer-Aided Diagnosis*, N. Karssemeijer and M. L. Giger, Eds., International Society for Optics and Photonics, vol. 7260, SPIE, 2009, pp. 441–450.

[99] G. Pfurtscheller, C. Neuper, C. Guger, W. Harkam, H. Ramoser, A. Schlogl, B. Obermaier, and M. Pregenzer, "Current trends in graz brain-computer interface (bci) research," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 2, pp. 216–219, 2000.

[100] J. MacCalla and A. Howard, "A plush switch for accessing tablet-based applications for children with mild to severe motor limitations," in *Rehabilitation Eng. and Technology Soc. of North America (RESNA) Annu. Conf.*, 2014.

[101] H. Park, M. Kiani, H.-M. Lee, J. Kim, J. Block, B. Gosselin, and M. Ghovanloo, "A wireless magnetoresistive sensing system for an intraoral tongue-computer interface," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 6, no. 6, pp. 571–585, 2012.

[102] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary region segmentation of objects in n-d images," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 1, 2001, 105–112 vol.1.

[103] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 9, pp. 1124–1137, 2004.

[104] D. Freedman and T. Zhang, "Interactive graph cut based segmentation with shape priors," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, 755–762 vol. 1.

[105] J. Ning, L. Zhang, D. Zhang, and C. Wu, "Interactive image segmentation by maximal similarity based region merging," *Pattern Recognition*, vol. 43, no. 2, pp. 445–456, 2010, Interactive Imaging and Vision.

[106] C. Rother, V. Kolmogorov, and A. Blake, ""grabcut": Interactive foreground extraction using iterated graph cuts," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 309–314, Aug. 2004.

[107] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr, "Interactive image segmentation using an adaptive gmmrf model," in *Computer Vision - ECCV 2004*, T. Pajdla and J. Matas, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 428–441, ISBN: 978-3-540-24670-1.

[108] V. Lempitsky, P. Kohli, C. Rother, and T. Sharp, "Image segmentation with a bounding box prior," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 277–284.

[109] L. Marchesotti, C. Cifarelli, and G. Csurka, "A framework for visual saliency detection with applications to image thumbnailing," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 2232–2239.

[110] H. Jiang, J. Wang, Z. Yuan, T. Liu, N. Zheng, and S. Li, "Automatic salient object segmentation based on context and shape prior.," in *BMVC*, vol. 6, 2011, p. 9.

[111] C. Rupprecht, L. Peter, and N. Navab, "Image segmentation in twenty questions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015.

[112] D.-J. Chen, H.-T. Chen, and L.-W. Chang, "Interactive 1-bit feedback segmentation using transductive inference," *Machine Vision and Applications*, vol. 29, no. 4, pp. 617–631, May 2018.

[113] ——, "Interactive segmentation from 1-bit feedback," in *Computer Vision – ACCV 2016*, S.-H. Lai, V. Lepetit, K. Nishino, and Y. Sato, Eds., Cham: Springer International Publishing, 2017, pp. 261–274, ISBN: 978-3-319-54181-5.

[114] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945. eprint: https://esajournals.onlinelibrary.wiley.com/doi/pdf/10.2307/1932409.

[115] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Cham: Springer International Publishing, 2014, pp. 740–755, ISBN: 978-3-319-10602-1.

[116] K. G. Jamieson and R. Nowak, "Active ranking using pairwise comparisons," in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, Eds., vol. 24, Curran Associates, Inc., 2011.

[117] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[118] V. Ferrari, T. Tuytelaars, and L. Van Gool, "Simultaneous object recognition and segmentation by image exploration," in *Computer Vision - ECCV 2004*, T. Pajdla and J. Matas, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 40–54, ISBN: 978-3-540-24670-1.

[119] L. Yang, R. Jin, L. Mummert, R. Sukthankar, A. Goode, B. Zheng, S. C. Hoi, and M. Satyanarayanan, "A boosting framework for visuality-preserving distance metric learning and its application to medical image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 30–44, 2010.

[120] D. Parikh and K. Grauman, "Relative attributes," in *2011 International Conference on Computer Vision*, 2011, pp. 503–510.

[121] O. Tamuz, C. Liu, S. Belongie, O. Shamir, and A. Kalai, "Adaptively learning the crowd kernel," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, L. Getoor and T. Scheffer, Eds., ser. ICML '11, Bellevue, Washington, USA: ACM, Jun. 2011, pp. 673–680, ISBN: 978-1-4503-0619-5.

[122] L. van der Maaten and K. Weinberger, "Stochastic triplet embedding," in *2012 IEEE International Workshop on Machine Learning for Signal Processing*, 2012, pp. 1–6.

[123] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *Similarity-Based Pattern Recognition*, A. Feragen, M. Pelillo, and M. Loog, Eds., Cham: Springer International Publishing, 2015, pp. 84–92, ISBN: 978-3-319-24261-3.

[124] B. Fernando, E. Gavves, D. Muselet, and T. Tuytelaars, "Learning to rank based on subsequences," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015.

[125] L. Liang and K. Grauman, "Beyond comparing image pairs: Setwise active learning for relative attributes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2014.

[126] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li, "Learning to rank: From pairwise approach to listwise approach," in *Proceedings of the 24th International Conference*

on Machine Learning, ser. ICML '07, Corvalis, Oregon, USA: Association for Computing Machinery, 2007, pp. 129–136, ISBN: 9781595937933.

[127] K. G. Jamieson and R. D. Nowak, "Low-dimensional embedding using adaptively selected ordinal data," in *2011 49th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2011, pp. 1077–1084.

[128] E. Y. Liu, Z. Guo, X. Zhang, V. Jojic, and W. Wang, "Metric learning from relative comparisons by minimizing squared residual," in *2012 IEEE 12th International Conference on Data Mining*, 2012, pp. 978–983.

[129] N. Chater and G. D. Brown, "Scale-invariance as a unifying psychological principle," *Cognition*, vol. 69, no. 3, B17–B24, 1999.

[130] L. Jain, K. G. Jamieson, and R. Nowak, "Finite sample prediction and recovery bounds for ordinal embedding," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29, Curran Associates, Inc., 2016.

[131] H. Yu, "Svm selective sampling for ranking with application to data retrieval," in *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, ser. KDD '05, Chicago, Illinois, USA: Association for Computing Machinery, 2005, pp. 354–363, ISBN: 159593135X.

[132] B. Qian, X. Wang, F. Wang, H. Li, J. Ye, and I. Davidson, "Active learning from relative queries," in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, ser. IJCAI '13, Beijing, China: AAAI Press, 2013, pp. 1614–1620, ISBN: 9781577356332.

[133] C. Cao and H.-Z. Ai, "Facial similarity learning with humans in the loop," *Journal of Computer Science and Technology*, vol. 30, no. 3, pp. 499–510, May 2015.

[134] G. Patterson, G. Van Horn, S. Belongie, P. Perona, and J. Hays, "Tropel: Crowdsourcing detectors with minimal training," *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, vol. 3, no. 1, Sep. 2015.

[135] M. Wilber, I. Kwak, and S. Belongie, "Cost-effective hits for relative similarity comparisons," *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, vol. 2, no. 1, Sep. 2014.

[136] N. Houlsby, F. Huszar, Z. Ghahramani, and J. Hernández-lobato, "Collaborative gaussian processes for preference learning," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25, Curran Associates, Inc., 2012.

[137] H. Stern, "Models for distributions on permutations," *Journal of the American Statistical Association*, vol. 85, no. 410, pp. 558–564, 1990. eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1990.10476235.

[138] M. Lohaus, P. Hennig, and U. von Luxburg, *Uncertainty estimates for ordinal embeddings*, 2019. arXiv: 1906.11655 [cs.LG].

[139] R. A. Bradley and M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons," *Biometrika*, vol. 39, no. 3/4, pp. 324–345, 1952.

[140] J. Guiver and E. Snelson, "Bayesian inference for plackett-luce ranking models," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML '09, Montreal, Quebec, Canada: Association for Computing Machinery, 2009, pp. 377–384, ISBN: 9781605585161.

[141] M. Wilber, I. S. Kwak, D. Kriegman, and S. Belongie, "Learning concept embeddings with combined human-machine expertise," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015.

[142] M. G. Kendall, "A new measure of rank correlation," *Biometrika*, vol. 30, no. 1/2, pp. 81–93, 1938.

[143] P. Kingston and M. Egerstedt, "Comparing apples and oranges through partial orders: An empirical approach," in *2009 American Control Conference*, 2009, pp. 5434–5439.

[144] J. Baldridge and A. Palmer, "How well does active learning *actually* work? Time-based evaluation of cost-reduction strategies for language documentation.," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore: Association for Computational Linguistics, Aug. 2009, pp. 296–305.

[145] H. A. David, *The method of paired comparisons*. London, 1963, vol. 12.

[146] C. H. Coombs, "Psychological scaling without a unit of measurement.," *Psychological review*, vol. 57, no. 3, p. 145, 1950.

[147] Y. Guo, P. Tian, J. Kalpathy-Cramer, S. Ostmo, J. P. Campbell, M. F. Chiang, D. Erdogmus, J. G. Dy, and S. Ioannidis, "Experimental design under the bradley-terry model.," in *IJCAI*, 2018, pp. 2198–2204.

[148] K. Jamieson, S. Katariya, A. Deshpande, and R. Nowak, "Sparse Dueling Bandits," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, G. Lebanon and S. V. N. Vishwanathan, Eds., ser. Proceedings of

Machine Learning Research, vol. 38, San Diego, California, USA: PMLR, Sep. 2015, pp. 416–424.

[149] F. Wauthier, M. Jordan, and N. Jojic, "Efficient ranking from pairwise comparisons," in *Proceedings of the 30th International Conference on Machine Learning*, S. Dasgupta and D. McAllester, Eds., ser. Proceedings of Machine Learning Research, vol. 28, Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 109–117.

[150] Y. Chen and C. Suh, "Spectral mle: Top-k rank aggregation from pairwise comparisons," in *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach and D. Blei, Eds., ser. Proceedings of Machine Learning Research, vol. 37, Lille, France: PMLR, Jul. 2015, pp. 371–380.

[151] B. Eriksson, "Learning to top-k search using pairwise comparisons," in *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, C. M. Carvalho and P. Ravikumar, Eds., ser. Proceedings of Machine Learning Research, vol. 31, Scottsdale, Arizona, USA: PMLR, 29 Apr–01 May 2013, pp. 265–273.

[152] N. B. Shah and M. J. Wainwright, "Simple, robust and optimal ranking from pairwise comparisons.," *Journal of machine learning research*, vol. 18, pp. 199–1, 2017.

[153] N. Shah, S. Balakrishnan, J. Bradley, A. Parekh, K. Ramchandran, and M. Wainwright, "Estimation from Pairwise Comparisons: Sharp Minimax Bounds with Topology Dependence," in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, G. Lebanon and S. V. N. Vishwanathan, Eds., ser. Proceedings of Machine Learning Research, vol. 38, San Diego, California, USA: PMLR, Sep. 2015, pp. 856–865.

[154] S. Negahban, S. Oh, and D. Shah, "Iterative ranking from pair-wise comparisons," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., 2012, pp. 2474–2482.

[155] L. Qian, J. Gao, and H. V. Jagadish, "Learning user preferences by adaptive pairwise comparison," *Proc. VLDB Endow.*, vol. 8, no. 11, pp. 1322–1333, Jul. 2015.

[156] B. Eric, N. Freitas, and A. Ghosh, "Active preference learning with discrete choice data," in *Advances in Neural Information Processing Systems*, J. Platt, D. Koller, Y. Singer, and S. Roweis, Eds., vol. 20, Curran Associates, Inc., 2008.

[157] W. Chu and Z. Ghahramani, "Preference learning with gaussian processes," in *Proceedings of the 22nd International Conference on Machine Learning*, ser. ICML '05, Bonn, Germany: Association for Computing Machinery, 2005, pp. 137–144, ISBN: 1595931805.

[158] A. K. Massimino and M. A. Davenport, "As you like it: Localization via paired comparisons," *arXiv preprint arXiv:1802.10489*, 2018.

[159] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[160] S. Prasad, "Certain relations between mutual information and fidelity of statistical estimation," *arXiv preprint arXiv:1010.1508*, 2010.

[161] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[162] B. Carpenter, A. Gelman, M. D. Hoffman, D. Lee, B. Goodrich, M. Betancourt, M. Brubaker, J. Guo, P. Li, and A. Riddell, "Stan: A probabilistic programming language," *Journal of statistical software*, vol. 76, no. 1, 2017.

[163] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng, *Handbook of markov chain monte carlo*. CRC press, 2011.

[164] L. Wu, C.-J. Hsieh, and J. Sharpnack, "Large-scale collaborative ranking in near-linear time," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '17, Halifax, NS, Canada: Association for Computing Machinery, 2017, pp. 515–524, ISBN: 9781450348874.

[165] M. A. Davenport, "Lost without a compass: Nonmetric triangulation and landmark multidimensional scaling," in *2013 5th IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, 2013, pp. 13–16.

[166] M. R. O'Shaughnessy and M. A. Davenport, "Localizing users and items from paired comparisons," in *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2016, pp. 1–6.

[167] A. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," *IEE Proceedings F (Radar and Signal Processing)*, vol. 140, 107–113(6), 2 Apr. 1993.

[168] C. Frowd, V. Bruce, M. Pitchford, C. Gannon, M. Robinson, C. Tredoux, J. Park, A. Mcintyre, and P. J. B. Hancock, "Evolving the face of a criminal: How to search a face space more effectively," *Soft Computing*, vol. 15, no. 1, pp. 61–70, Jan. 2011.

[169] E. E. Ventura, J. N. Davis, and M. I. Goran, "Sugar content of popular sweetened beverages based on objective laboratory analysis: Focus on fructose content," *Obe-*

sity, vol. 19, no. 4, pp. 868–874, 2011. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1038/oby.2010.255.

[170] M. Naghshvar, T. Javidi, and K. Chaudhuri, "Bayesian active learning with non-persistent noise," *IEEE Transactions on Information Theory*, vol. 61, no. 7, pp. 4080–4098, 2015.

[171] N. Houlsby, F. Huszár, Z. Ghahramani, and M. Lengyel, *Bayesian active learning for classification and preference learning*, 2011. arXiv: 1112.5745 [stat.ML].

[172] E. Arias-Castro, E. J. Candes, and M. A. Davenport, "On the fundamental limits of adaptive sensing," *IEEE Transactions on Information Theory*, vol. 59, no. 1, pp. 472–481, 2013.

[173] R. M. Castro and R. D. Nowak, "Minimax bounds for active learning," *IEEE Transactions on Information Theory*, vol. 54, no. 5, pp. 2339–2353, 2008.

[174] Y. Yang and M. Loog, "A benchmark and comparison of active learning for logistic regression," *Pattern Recognition*, vol. 83, pp. 401–415, 2018.

[175] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of machine learning research*, vol. 2, no. Nov, pp. 45–66, 2001.

[176] C. Zhang, J. Shen, and P. Awasthi, *Efficient active learning of sparse halfspaces with arbitrary bounded noise*, 2020. arXiv: 2002.04840 [cs.LG].

[177] J. Singh, O. Dabeer, and U. Madhow, "On the limits of communication with low-precision analog-to-digital conversion at the receiver," *IEEE Transactions on Communications*, vol. 57, no. 12, pp. 3629–3639, 2009.

[178] M. Arjovsky, S. Chintala, and L. Bottou, *Wasserstein gan*, 2017. arXiv: 1701.07875 [stat.ML].

[179] C. Villani, *Optimal transport: old and new*. Springer Science & Business Media, 2008, vol. 338.

[180] C. M. Bishop, *Pattern recognition and machine learning*, ser. Information science and statistics. New York, NY: Springer, 2006, Softcover published in 2016.

[181] S. Dasgupta, "Two faces of active learning," *Theoretical Computer Science*, vol. 412, no. 19, pp. 1767–1781, 2011, Algorithmic Learning Theory (ALT 2009).

[182] A. Beygelzimer, S. Dasgupta, and J. Langford, "Importance weighted active learning," in *Proceedings of the 26th International Conference on Machine Learning*, L. Bottou and M. Littman, Eds., Montreal: Omnipress, Jun. 2009, pp. 49–56.

[183] S. Farquhar, Y. Gal, and T. Rainforth, "On statistical bias in active learning: How and when to fix it," in *International Conference on Learning Representations*, 2021.

[184] S. Dasgupta and D. Hsu, "Hierarchical sampling for active learning," A. McCallum and S. Roweis, Eds., pp. 208–215, 2008.

[185] S.-j. Huang, R. Jin, and Z.-H. Zhou, "Active learning by querying informative and representative examples," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds., Curran Associates, Inc., 2010, pp. 892–900.

[186] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jun. 2008.

[187] T. S. Jaakkola and M. I. Jordan, "Bayesian parameter estimation via variational methods," *Statistics and Computing*, vol. 10, no. 1, pp. 25–37, Jan. 2000.

[188] D. Dua and C. Graff, *UCI machine learning repository*, 2017.

[189] R. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Transactions on Information Theory*, vol. 18, no. 4, pp. 460–473, 1972.

[190] S. Arimoto, "An algorithm for computing the capacity of arbitrary discrete memoryless channels," *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 14–20, 1972.

[191] G. Peyré and M. Cuturi, "Computational optimal transport: With applications to data science," *Foundations and Trends® in Machine Learning*, vol. 11, no. 5-6, pp. 355–607, 2019.

[192] M. Gastpar, B. Rimoldi, and M. Vetterli, "To code, or not to code: Lossy source-channel communication revisited," *IEEE Transactions on Information Theory*, vol. 49, no. 5, pp. 1147–1158, 2003.

[193] A. Ben-Yishai and O. Shayevitz, "Interactive schemes for the awgn channel with noisy feedback," *IEEE Transactions on Information Theory*, vol. 63, no. 4, pp. 2409–2427, 2017.

[194] J. Wu and A. Anastasopoulos, "Zero-rate achievability of posterior matching schemes for channels with memory," in *2016 IEEE International Symposium on Information Theory (ISIT)*, 2016, pp. 2384–2388.

[195] T. Koch and A. Lapidoth, "At low snr, asymmetric quantizers are better," *IEEE Transactions on Information Theory*, vol. 59, no. 9, pp. 5421–5445, 2013.

[196] R. Mathar and M. Dörpinghaus, "Threshold optimization for capacity-achieving discrete input one-bit output quantization," in *2013 IEEE International Symposium on Information Theory*, 2013, pp. 1999–2003.

[197] K. E. Train, *Discrete choice methods with simulation*. Cambridge university press, 2009.

[198] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," in *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., vol. 26, Curran Associates, Inc., 2013.

[199] J. Lee, M. Dabagia, E. Dyer, and C. Rozell, "Hierarchical optimal transport for multimodal distribution alignment," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019.

[200] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "The robotarium: A remotely accessible swarm robotics research testbed," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 1699–1706.

[201] D. Pickem, M. Lee, and M. Egerstedt, "The gritsbot in its natural habitat - a multi-robot testbed," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 4062–4067.

[202] S. Wilson, P. Glotfelter, L. Wang, S. Mayya, G. Notomista, M. Mote, and M. Egerstedt, "The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems," *IEEE Control Systems Magazine*, vol. 40, no. 1, pp. 26–44, 2020.

[203] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.

[204] R. Olfati-Saber, "Near-identity diffeomorphisms and exponential/spl epsi/-tracking and/spl epsi/-stabilization of first-order nonholonomic se (2) vehicles," in *Proceed-*

ings of the 2002 american control conference (ieee cat. no. ch37301), IEEE, vol. 6, 2002, pp. 4690–4695.

[205] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller, "Optimal spatial filtering of single trial eeg during imagined hand movement," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 4, pp. 441–446, 2000.

[206] J. Müller-Gerking, G. Pfurtscheller, and H. Flyvbjerg, "Designing optimal spatial filters for single-trial eeg classification in a movement task," *Clinical Neurophysiology*, vol. 110, no. 5, pp. 787–798, 1999.

[207] C. Guger, H. Ramoser, and G. Pfurtscheller, "Real-time eeg analysis with subject-specific spatial patterns for a brain-computer interface (bci)," *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 4, pp. 447–456, 2000.

[208] D. J. McFarland, L. M. McCane, S. V. David, and J. R. Wolpaw, "Spatial filter selection for eeg-based communication," *Electroencephalography and Clinical Neurophysiology*, vol. 103, no. 3, pp. 386–394, 1997.

[209] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[210] C. Neuper, A. Schlögl, and G. Pfurtscheller, "Enhancement of left-right sensorimotor eeg differences during feedback-regulated motor imagery," *Journal of Clinical Neurophysiology*, vol. 16, no. 4, 1999.

[211] Y. Aflalo and R. Kimmel, "Spectral multidimensional scaling," *Proceedings of the National Academy of Sciences*, vol. 110, no. 45, pp. 18 052–18 057, 2013. eprint: https://www.pnas.org/content/110/45/18052.full.pdf.

[212] A. Marsiglietti and V. Kostina, "A lower bound on the differential entropy of log-concave random vectors with applications," *Entropy*, vol. 20, no. 3, 2018.

[213] S. G. Bobkov and M. M. Madiman, "On the problem of reversibility of the entropy power inequality," in *Limit Theorems in Probability, Statistics and Number Theory*, P. Eichelsbacher, G. Elsner, H. Kösters, M. Löwe, F. Merkl, and S. Rolles, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 61–74, ISBN: 978-3-642-36068-8.

[214] R. Durrett, *Probability: theory and examples*. Cambridge university press, 2010.

[215] A. R. Klivans, P. M. Long, and A. K. Tang, "Baum's algorithm learns intersections of halfspaces with respect to log-concave distributions," in *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, I. Dinur, K. Jansen, J. Naor, and J. Rolim, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 588–600, ISBN: 978-3-642-03685-9.

[216] Q. Mérigot, "A multiscale approach to optimal transport," *Computer Graphics Forum*, vol. 30, no. 5, pp. 1583–1592, 2011. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2011.02032.x.

[217] A. Winkelbauer, *Moments and absolute moments of the normal distribution*, 2014. arXiv: 1209.4340 [math.ST].

[218] J. Siebert, "Vehicle recognition using rule based methods," Turing Institute, Glasgow, Project Report, 1987.

[219] Y. Ma, R. Nowak, P. Rigollet, X. Zhang, and X. Zhu, *Teacher improves learning by selecting a training subset*, 2018. arXiv: 1802.08946 [stat.ML].

[220] J. T. Ash, C. Zhang, A. Krishnamurthy, J. Langford, and A. Agarwal, "Deep batch active learning by diverse, uncertain gradient lower bounds," in *International Conference on Learning Representations*, 2020.