Practical Unbiased Bayesian Pairwise Preference Learning and Inference on the Uniform Convex Polytopes

Abstract

In Bayesian approaches to utility learning from preferences, the objective is to infer a posterior belief distribution over an agents utility function based on previously observed agent preferences. From this, one can then estimate quantities such as the expected utility of a decision or the probability of an unobserved preference, which can then be used to make or suggest future decisions on behalf of the agent. In this paper, we build on Bayesian pairwise preference learning models under the assumptions of linearly additive multi-attribute utility functions and a bounded uniform utility prior. These assumptions lead to a posterior form that is a uniform distribution over a convex polytope for which we then demonstrate how to perform practical unbiased inference w.r.t. this posterior that is more scalable than existing unbiased methods and leads to improved prediction accuracy over state-of-the-art biased methods.

Introduction

Decision support systems (DSSs) require knowledge of an agent's preferences in order to make or suggest future decisions on their behalf. Clearly then, the most crucial task of a DSS is in learning to predict future agent preferences from previously observed agent preferences.

In recent years, an increasingly popular approach to preference learning relies on Bayesian updating of posterior beliefs in an underlying (multi-attribute) utility function conditioned on observed agent preferences (**?**; **?**; **?**; **?**; **?**; **?**); inference can then be performed by marginalizing over these posterior utility beliefs w.r.t. future preference queries. This *Bayesian preference learning* (BPL) is attractive for a multitude of reasons, among them (a) the generally acknowledged robustness of Bayesian learning methods to overfitting, (b) the fact that updates can be done online, and (c) the fact that Bayesian methods provide a useful measure of confidence in their predictions via a probability distribution over predicted outcomes.

Despite the attractiveness of BPL, we are not aware of a Bayesian *pairwise* preference learning (BPPL) approach that claims to do all learning and inference in an exact, closed-form. Arguably the two most common priors for BPPL are either Gaussian or Uniform — in the former case, a Gaussian prior is not conjugate to commonly used pairwise preference likelihoods leading to the need for approximate Bayesian updating in order to maintain a tractable form for the posterior (**?**; **?**; **?**; **?**), in the latter case, certain Uniform distributions are indeed conjugate to commonly used pairwise preference likelihoods, *but* inference with the posterior convex polytope requires difficult integrals that in practice are approximated via sampling or other methods (**?**; **?**).

In this paper, we argue that the exact integrals in the case of BPPL with commonly used linearly additive multi-attribute utility functions and a uniform utility prior are not as computationally difficult as they may appear. Indeed, armed with symbolic inference for piecewise functions and efficient data structures derived from algebraic decision diagram (ADD) for representing and performing operations on these functions, we show that such inference can be computed efficiently for reasonably sized problems in practice. This is the first time (to the best of the authors' knowledge) that an explicit algorithm has been provided for doing BPPL and all associated inference in exact, closed-form.

Bayesian Pairwise Preference Learning (BPPL)

Here we follow much of the BPPL setup of (**?**), except for the case of a Uniform rather than Gaussian prior distribution. In *multi-attribute utility theory* (MAUT) (**?**), utilities are modeled over a D-dimensional *attribute vector* $\mathbf{x}$ where each *attribute choice* x_d ($1 \leq d \leq D$) is either binary ($x_d \in \{0 \text{ (false)}, 1 \text{ (true)}\}$) or real-valued ($x_d \in \mathbb{R}$). An *item* is described by its attribute choice assignments $\mathbf{x} = (x_1, \ldots, x_D)$. For example, let our item space consist of possible cars we might buy, let binary attribute choice x_1 represent whether the car is a *2-door* (if false, we assume it is a *4-door*) and let x_2 represent the amount of *engine displacement in liters*. Perhaps two of our available items are a 2.4L two-door Accord with attribute vector $\mathbf{x}^a = (1, 2.4)$ and a 3.5L four-door Buick with attribute vector $\mathbf{x}^b = (0, 3.5)$.

Our goal in BPPL will be to learn an *attribute weight vector* $\mathbf{w} = \mathbb{R}^D$ that describes the utility of *each* attribute choice in *each* attribute dimension. As commonly done in MAUT (**?**; **?**), we assume that the overall utility $u(\mathbf{x}|\mathbf{w})$ of item $\mathbf{x}$ w.r.t. attribute weight vector $\mathbf{w}$ decomposes *ad-*

ditively over the attribute choices of $\mathbf{x}$, i.e.,

$$u(\mathbf{x}|\mathbf{w}) = \sum_{d=1}^{D} \mathbf{w}_d \mathbf{x}_d. \quad (1)$$

We let u^* represent the user's true utility w.r.t. their true (but hidden) $\mathbf{w}^*$. Since $\mathbf{w}^*$ is unknown to the decision support system, we take a Bayesian perspective on learning $\mathbf{w}$ (**?**) and thus maintain a probability distribution $P(\mathbf{w})$ representing our beliefs over $\mathbf{w}^*$. For this, we must define our prior utility beliefs, likelihood (preference query response model), the form of the posterior for our Bayesian update, and the form of any inferences we wish to make with the posterior.

Utility Prior

For our prior beliefs $P(\mathbf{w})$, we start with a hyper-rectangular multivariate uniform distribution over $\mathbf{w}$ represented as follows

$$P(\mathbf{w}) = \prod_{d=1}^{D} p(w_d) = \prod_{d=1}^{D} \mathcal{U}(w_d; -C, C). \quad (2)$$

where $C \in \mathbb{R}$ ($C \neq 0$) is some constant [1] and the uniform distribution $\mathcal{U}$ is simply defined using an indicator function[2]:

$$\mathcal{U}(w_d; L, U) = \frac{1}{U - L} \mathbb{I}[(w_d \geq L) \wedge (w_d \leq U)].$$

By using respective lower and upper bounds of $-C$ and C for each uniform dimension of the prior, we enforce that $\mathbb{E}_{\mathbf{w} \sim P(\mathbf{w})}[\mathbf{w}] = \mathbf{0}$ indicating agnostic prior beliefs about the utility of each attribute.

As it will be important observation for future sections, we remark that such a hyperrectangular uniform prior as defined here can also be conveniently viewed as the intersection (product) of half-spaces given by the indicator definition of $\mathcal{U}$.

Query & User Response Model

In this paper, we focus on pairwise comparison queries known to require low cognitive load for users to answer (**?**). We use $Q_{ab} = \{a \succ b, a \prec b, a \sim b\}$ to represent a pairwise comparison query indicating the user's preferences of item $\mathbf{x}^a$ vs. item $\mathbf{x}^b$ (henceforth just a and b). Depending on the user's attribute weight vector $\mathbf{w}$ and the corresponding item utilities, $u(a|\mathbf{w})$ and $u(b|\mathbf{w})$, the user's response $q_{ab} \in Q_{ab}$ indicates the following:

- $a \succ b$: the user prefers a to b,
- $a \prec b$: the user prefers a to b,
- $a \sim b$: the user is indifferent between a and b.

[1] In practice, C could be different for each dimension, if justified.

[2] We use $\mathbb{I}[\cdot]$ as an indicator function taking the value 1 when its argument is true and 0 otherwise.

We represent the user query model in (5) with an indicator function over the pairwise utility differences as follows

$$\begin{aligned} P(Q_{ab} = a \succ b|\mathbf{w}) &= \mathbb{I}[u(a|\mathbf{w}) - u(b|\mathbf{w}) > \epsilon] \\ P(Q_{ab} = a \prec b|\mathbf{w}) &= \mathbb{I}[u(b|\mathbf{w}) - u(a|\mathbf{w}) > \epsilon] \\ P(Q_{ab} = a \sim b|\mathbf{w}) &= \mathbb{I}[|u(a|\mathbf{w}) - u(b|\mathbf{w})| \leq \epsilon], \end{aligned} \quad (3)$$

where we can modulate the range of utility differences for which the user is indifferent by adjusting ϵ. Note that by definition, $\sum_{q_{ab}} P(q_{ab}|\mathbf{w}) = 1$.

If needed, we could easily model more noise in the elicitation process simply by allowing non-zero probability for weight vectors that disagree with the query response. For example, here we model a 10% chance that a user's query response disagrees with their utility valuation of items a and b by more than ϵ:

$$\begin{aligned} P(Q_{ab} = a \succ b|\mathbf{w}) &= 0.9 \cdot \mathbb{I}[u(a|\mathbf{w}) - u(b|\mathbf{w}) > \epsilon] \\ &\quad + 0.1 \cdot \mathbb{I}[u(a|\mathbf{w}) - u(b|\mathbf{w})] \leq \epsilon. \end{aligned} \quad (4)$$

Whatever exact form of the above query response model is used, we note that due to the linearity of $u(\cdot|\mathbf{w})$ in $\mathbf{w}$, all of the above query response likelihood models can be represented by (a sum of) linearly separated half-spaces owing to the the linear constraints in the likelihood indicator functions.

Bayesian Utility Updating

Given a prior utility belief $P(\mathbf{w}|R^n)$ w.r.t. a (possibly empty) set of $n \geq 0$ query responses $R^n = \{q_{kl}\}$ and a new query response q_{ab}, we perform the following Bayesian update to obtain a posterior belief $P(\mathbf{w}|R^{n+1})$ where $R^{n+1} = R^n \cup \{q_{ab}\}$:

$$\begin{aligned} P(\mathbf{w}|R^{n+1}) &\propto P(q_{ab}|\mathbf{w}, R^n) P(\mathbf{w}|R^n) \\ &\propto P(q_{ab}|\mathbf{w}) P(\mathbf{w}|R^n). \end{aligned} \quad (5)$$

Assuming that our query likelihood $P(q_{ab}|\mathbf{w})$ is modeled as described in (3) [or (4)], we note that since the prior is a product of linear half-spaces and the likelihood also [a sum of] linear half-spaces and these are multiplied, the resulting posterior is a *uniform convex polytope*[3] [or after placing the result in sum of products form, a sum of uniform convex polytopes]. Hence, an unnormalized Bayesian update appears to be computationally straightforward — and in fact a very efficient $O(nD)$ complexity for Bayesian updating for the case of the likelihood model in (3).

In Figure 1, we show a $D = 2$ representation of a posterior for a likelihood in the form of (3)

Preference Inference

Now that we have posterior beliefs in our utility weighting $P(\mathbf{w}|R^n)$, we would like to perform at least two types of inference:

[3] Here an unnormalized uniform distribution on the convex polytope defined by the product of linear constraints in the indicator functions.

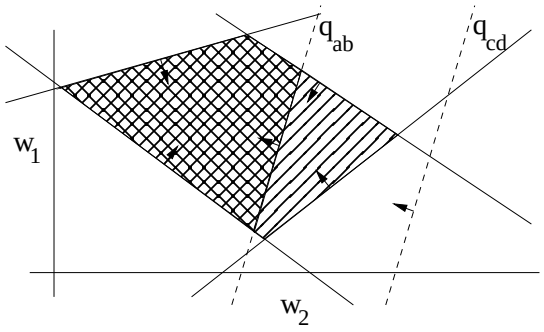

Figure 1: A representation of the uniform convex polytope representing the posterior distribution on $\mathbf{w}$ when using the likelihood model in (3). The union of the striped and cross-hatched area shows the non-zero posterior polytope before updating with query response q_{ab} and the cross-hatched area shows the non-zero posterior polytope after updating with query response q_{ab}. We note that query response q_{cd} would not lead to any change in either posterior before or after updating with q_{cd} and its constraint could be pruned from the constraint set.

- *Item with maximum expected utility:*

$$\arg\max_{\mathbf{x}} \ \mathbb{E}_{\mathbf{w}} \left[u(\mathbf{x}|\mathbf{w}) \middle| R^n \right] \tag{6}$$

$$= \arg\max_{\mathbf{x}} \int_{\mathbf{w}} P(\mathbf{w}|R^n) \left[\sum_{d=1}^{D} \mathbf{w}_d \mathbf{x}_d \right] d\mathbf{w} \tag{7}$$

- *Probability of preference outcomes:*

$$P(q_{ab}|R^n) \ = \int_{\mathbf{w}} P(q_{ab}|\mathbf{w}) P(\mathbf{w}|R^n) d\mathbf{w} s$$

If using likelihood form (3), we note that both inferences are volume integrals over a function ($u(\mathbf{x}|\mathbf{w})$ in the first case and constant in the latter case) subject to boundary constraints over a convex polytope defined w.r.t. $P(\mathbf{w}|R^n)$ (appearing in both inferences) and $P(q_{ab}|\mathbf{w})$ (only for the second inference). In the case of likelihood form (3), we simply get a sum of such integrals. Consequently, the question we have to ask to perform exact inference in this Uniform BPPL setting is the following: given a function defined over the region of a convex polytope like the one in Figure 1, how do we compute this integral in closed-form?

At first, this would seem quite difficult — the boundaries in any dimension in Figure 1 may be a piecewise function of the other dimensions — but on closer inspection, there is a systematic way to compute these integrals if given the right representation. We discuss this next.

Symbolic Variable Elimination in Discrete/Continuous Graphical Models

In this section, we introduce a *case* notation and operations for piecewise polynomial functions and demonstrate that all Uniform BPPL operations, including the required integrals can be computed exactly and in closed-form using a purely *symbolic* representation.

Case Representation and Operators

For this section, we will assume all functions are represented in *case* form (**?**) as follows:

$$f = \begin{cases} \phi_1 & f_1 \\ \vdots & \vdots \\ \phi_k & f_k \end{cases} \tag{8}$$

Here, the f_i may be *polynomial* functions of $\mathbf{w}$ (although initially the functions f_i will be constant or linear, we will see that integration will introduce higher order polynomials for f_i). The ϕ_i are logical formulae defined over $\mathbf{w}$ that can consist of arbitrary logical combinations of inequalities ($\geq, >, \leq, <$) over *linear* functions of $\mathbf{w}$. We assume that the set of conditions $\{\phi_1, \ldots, \phi_k\}$ disjointly and exhaustively partition $\mathbf{w}$ such that f is well-defined for all $\mathbf{w} \in \mathbb{R}^D$. It is easy to verify that such a representation can represent all of the functions discussed in Section .

Unary operations such as scalar multiplication $c \cdot f$ (for some constant $c \in \mathbb{R}$) or negation $-f$ on case statements f are straightforward; the unary operation is simply applied to each f_i ($1 \leq i \leq k$). Intuitively, to perform a *binary operation* on two case statements, we simply take the cross-product of the logical partitions of each case statement and perform the corresponding operation on the resulting paired partitions. Thus, we perform the "cross-sum" $\oplus$ of two (unnamed) cases as follows:

$$\begin{cases} \phi_1 : & f_1 \\ \phi_2 : & f_2 \end{cases} \oplus \begin{cases} \psi_1 : & g_1 \\ \psi_2 : & g_2 \end{cases} = \begin{cases} \phi_1 \wedge \psi_1 : & f_1 + g_1 \\ \phi_1 \wedge \psi_2 : & f_1 + g_2 \\ \phi_2 \wedge \psi_1 : & f_2 + g_1 \\ \phi_2 \wedge \psi_2 : & f_2 + g_2 \end{cases}$$

Likewise, we perform $\ominus$ and $\otimes$ by, respectively, subtracting or multiplying partition values (rather than adding them) to obtain the result. Some partitions resulting from the application of the $\oplus$, $\ominus$, and $\otimes$ operators may be inconsistent (infeasible); if we can detect this (e.g., via a linear constraint solver), we may simply discard such partitions as they are irrelevant to the function value.

For the Uniform BPPL inference defined in Section , we'll need to compute definite integrals — a fairly non-trivial operation that is discussed in its own section. But first we discuss maximization needed for working with integral bounds.

Symbolic maximization is fairly straightforward to define:

$$\max\left(\begin{cases} \phi_1 : & f_1 \\ \phi_2 : & f_2 \end{cases}, \begin{cases} \psi_1 : & g_1 \\ \psi_2 : & g_2 \end{cases} \right) = \begin{cases} \phi_1 \wedge \psi_1 \wedge f_1 > g_1 : & f_1 \\ \phi_1 \wedge \psi_1 \wedge f_1 \leq g_1 : & g_1 \\ \phi_1 \wedge \psi_2 \wedge f_1 > g_2 : & f_1 \\ \phi_1 \wedge \psi_2 \wedge f_1 \leq g_2 : & g_2 \\ \phi_2 \wedge \psi_1 \wedge f_2 > g_1 : & f_2 \\ \phi_2 \wedge \psi_1 \wedge f_2 \leq g_1 : & g_1 \\ \phi_2 \wedge \psi_2 \wedge f_2 > g_2 : & f_2 \\ \phi_2 \wedge \psi_2 \wedge f_2 \leq g_2 : & g_2 \end{cases}$$

The key observation here is that case statements are closed under the max operation. While it may appear that this representation will lead to an unreasonable blowup in size, we note the XADD that we introduce later in Section will be

able to exploit the internal decision structure of this maximization to represent it much more compactly.

We have almost all of the operations needed to perform Uniform BPPL inference as defined in Section , except for a method to compute each of the integrals $\int_{w_1}, \ldots, \int_{w_D}$ w.r.t. the convex polytope constraints such as those shown in Figure 1. Finding the maximum Expected Utility that is required for preference learning can be done by performing the following steps:

- finding the posterior by symbolic multiplication of the prior and the likelihood
- performing symbolic integral as detailed in (**?**)
- finding the symbolic maximum as introduced earlier

Even though this procedure will produce the exact solution as the number of preferences increase, computing the integral becomes prohibitively hard. This is mainly because as the number of instances (that will introduce constraints as in Figure 1) increase, the resulting polytope becomes

- indefinite on XADD - traverses XADD to leaves - each path to leaves specifies a disjoint interval given by constant [LB, UB] (can eval indef integral at two points and subtract, this is the mass of the mixture of uniforms in that interval) - builds a number line with mixture of uniforms (computes normalizer Z) - draws u U(0,1) and proceeds to determine sampled value

Extended ADDs (XADDs) for Case Statements

In practice, it can be prohibitively expensive to maintain a case statement representation of a value function with explicit partitions. Motivated by algebraic decision diagrams (ADDs) (**?**), which maintain compact representations for finite discrete functions, we use an extended ADD (XADD) formalism introduced in (**?**) and demonstrated in Figure **??** (right).

In brief we note that an XADD is like an algebraic decision diagram (ADD) (**?**) except that (a) the decision nodes can have arbitrary inequalities, equalities, or disequalities (one per node) and (b) the leaf nodes can represent arbitrary functions. The decision nodes still have a fixed order from root to leaf and the standard ADD operations to build a canonical ADD (REDUCE) and to perform a binary operation on two ADDs (APPLY) still applies in the case of XADDs.

Of particular importance is that the XADD is a directed acyclic graph (DAG), and hence is often much more compact than a tree representation. For example, in Figure **??** (left) we provide the function represented by the XADD in Figure **??** (right) expanded from a DAG into it's full tree form. We note that not only is the XADD much more compact on account of its DAG, but that all unary and binary operations on XADDs can directly exploit this compact structure for efficiency. For more compactness and hence efficiency, we can apply a linear constraint solver to prune out unreachable branches of the convex polytope so that for example, dominated polytope constraints for preference q_{cd} in Figure 1 can simply be pruned from the XADD.

To this end, we can use the XADD to perform all case operations required for inference in Section and hence efficiently perform the Uniform BPPL queries from Section as we demonstrate next.

Empirical Results

In this section we demonstrate that performing the inference from Section is efficient in most cases for a reasonable number of attribute dimensions and pairwise query response observations.

Conclusions and Future Work

This work presented novel and efficient, exact Bayesian learning and inference methods for preference elicitation on the Uniform distribution on the convex polytope. In future work, we aim to investigate the following possible extensions:

- *Generalized additive independence* (**?**; **?**): can we introduce additive independence over joint utility factors rather than individual utility attributes while still achieving efficient computation? In principle, this seems relatively straightforward.
- *Nonlinear utility functions*: under what conditions can we use nonlinear utility functions and still compute the inferences from Section ?
- *Multi-user utility inference*: how can we extend the inference in this paper to leverage query responses from multiple users in a Bayesian *collaborative filtering* (**?**) framework?