Decision-theoretic Sparsification for Gaussian Process Preference Learning

M. Ehsan Abbasnejad[1], Edwin V. Bonilla[2], and Scott Sanner[3]

[1] Australian National University (ANU) and National ICT Australia (NICTA)[⋆]
`eabbasnejad@anu.edu.au`
[2] National ICT Australia (NICTA) and Australian National University (ANU)
`ebonilla@nicta.com.au`
[3] National ICT Australia (NICTA) and Australian National University (ANU)
`ssanner@nicta.com.au`

Abstract. We propose a decision-theoretic sparsification method for Gaussian process preference learning. This method overcomes the loss-insensitive nature of popular sparsification approaches such as the Informative Vector Machine (IVM). Instead of selecting a subset of users and items as inducing points based on uncertainty-reduction principles, our sparsification approach is underpinned by decision theory and directly incorporates the loss function inherent to the underlying preference learning problem. We show that by selecting different specifications of the loss function, the IVM's differential entropy criterion, a value of information criterion, and an upper confidence bound (UCB) criterion used in the bandit setting can all be recovered from our decision-theoretic framework. We refer to our method as the *Valuable Vector Machine* (VVM) as it selects the most useful items during sparsification to minimize the corresponding loss. We evaluate our approach on one synthetic and two real-world preference datasets, including one generated via Amazon Mechanical Turk and another collected from Facebook. Experiments show that variants of the VVM significantly outperform the IVM on all datasets under similar computational constraints.

1 Introduction

Preference learning has become an important subfield in machine learning transcending multiple disciplines such as economics, operations research and social sciences. A wide range of applications in areas such as recommender systems, autonomous agents, human-computer interaction and e-commerce has motivated machine learning researchers to investigate flexible and effective ways to construct predictive preference models from preference observations. This is a challenging problem since complex relations between users and their preferred products (items) must be uncovered. Furthermore, flexible and principled ways to

[⋆] NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

handle uncertainty over the users' preferences are required in order to balance what the system knows. To address these challenges, non-parametric Bayesian approaches based on Gaussian processes (GPs) have shown to be effective in real applications [1,2,3,4]. However, one of the major limitations of preference learning approaches based on GPs is their poor scalability when dealing with a large number of users and items.

Scalability issues in GPs are not exclusive to preference learning and they are common in other settings such as regression and classification. It is customary in these settings to adopt *sparsification* approaches, where a subset of training examples is selected as inducing points, considerably reducing the time complexity of posterior approximation and prediction [5,6,7]. A popular approach to GP sparsification is the Informative Vector Machine [8], where inducing points are selected according to an information-theoretic criterion. A key characteristic of the IVM is that it can be embedded naturally in sequential algorithms such as Assumed Density Filtering [9] or Expectation Propagation [10]. These algorithms provide efficient computation of the quantities of interest (i.e., posterior variances) to be used by the IVM's sparsification criterion.

Nevertheless, the IVM's purely entropic sparsification criterion fails at addressing the varying loss functions that may be of interest to the final decision-theoretic task — especially those tasks that naturally arise in preference learning. For example, we might be interested in (a) optimizing the utility of the best recommendation, (b) giving a ranking of all items (or a subset), or (c) correctly classifying all pairwise preferences. In each case we seek to optimize a loss for a different decision-theoretic task and when we need to approximate in a Bayesian setting, it is important that our approximation is loss-calibrated [11]. We note that the uncertainty reduction principle inherent to the IVM approximation is not loss-calibrated for all tasks (a)–(c).

In this paper, we continue to bridge the gap between decision theory and approximate Bayesian inference [11] in a direction that leverages the efficiency of the IVM approach for GP sparsification, while overcoming its loss-insensitive approximation. We show that the IVM's differential entropy criterion, a value of information criterion, and an upper confidence bound [12] criterion can *all* be recovered in our framework by specifying the appropriate loss.

An additional important aspect of the preference learning problem that distinguishes it from standard machine learning settings is that the complexity of making predictions does not directly depend upon the number of observations (i.e. preference relations), but rather the number of users and items. Our method takes this into consideration and adopts an item-driven sparsification strategy that retains the items that best encode the users' preferences. Our experiments show that this is an effective way of reducing the complexity of inference in preference learning with GPs while addressing the objective function of interest directly. We refer to our generic method as the *Valuable Vector Machine* (VVM) since it incorporates the loss function directly into its sparsification mechanism.

The rest of this paper is organized as follows: we outline the use of GPs for multi-user preference learning and prediction in Section 2 followed by our our proposed VVM sparsification framework in Section 3 and empirical evaluation in Section 4. We differentiate our approach from related work in Section 5 and conclude in Section 6.

2 GPs for Preference Learning

In this section, we define a general approximation framework for Bayesian preference learning via Gaussian Processes that we will adapt to the loss-calibrated setting in the next section.

Let $U = \{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n\}$ be a set of n users and let $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ be a set of m items and denote the set of observed preferences of each user $\mathbf{u} \in U$ with $\mathcal{D}^{\mathbf{u}} = \{\mathbf{x}_i \succ \mathbf{x}_j\}$ where $1 \leq i \leq m$ and $1 \leq j \leq m$. Given the preferences $\mathcal{D}^{\mathbf{u}}$ for $\mathbf{u}$, satisfaction of the von Neumann-Morgenstern axioms [13] justify the existence of utilities $f_i^{\mathbf{u}} \in \mathbb{R}$ for each item $\mathbf{x}_i \in X$ s.t. $\mathbf{x}_i \succ \mathbf{x}_j \in D^{\mathbf{u}}$ iff $f_i^{\mathbf{u}} > f_j^{\mathbf{u}}$. In order to model the distribution over these utilities, we build upon the model proposed by [2]. We denote a latent utility vector $\mathbf{f}$ for *all* users and items with $\mathbf{f} = [f_1^{\mathbf{u}_1}, f_2^{\mathbf{u}_1}, \ldots, f_m^{\mathbf{u}_n}]^T$. Then, we can define the likelihood over all the preferences given the latent functions as:

$$p(\mathcal{D}|\mathbf{f}) = \prod_{\mathbf{u} \in U} \prod_{\mathbf{x}_i \succ \mathbf{x}_j \in \mathcal{D}^{\mathbf{u}}} p(\mathbf{x}_i \succ \mathbf{x}_j | f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) \tag{1}$$

$$\text{with} \qquad p(\mathbf{x}_i \succ \mathbf{x}_j | f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) = \Phi\left(\frac{f_i^{\mathbf{u}} - f_j^{\mathbf{u}}}{\alpha}\right), \tag{2}$$

where $\Phi(x) = \int_{-\infty}^{x} \mathcal{N}(y)dy$ and $\mathcal{N}(y)$ is a zero-mean Gaussian distribution with unit variance. In this model, $p(\mathbf{f})$ is the prior over the latent utilities $\mathbf{f}$ and is defined via a GP with zero-mean function and a covariance function that factorizes over users and items [2]. Therefore:

$$p(\mathbf{f}) = \mathcal{N}(\mathbf{f}; \mathbf{0}, \mathbf{K}), \quad \mathbf{K} = \mathbf{K}_u \otimes \mathbf{K}_x, \tag{3}$$

where $\mathbf{K}$ is the kernel matrix composed of the Kronecker product of the kernel matrix over the users $\mathbf{K}_u$ and the kernel matrix over the items $\mathbf{K}_x$. One interesting feature of this model is the inherent transfer of preferences across users through the correlated prior, which will subsequently help the prediction on those users for which there are not many preferences recorded. Additionally, as we shall see later, having a fully factorized likelihood across users and items will facilitate the application of sequential approximate posterior inference methods such as *Expectation Propagation* (EP) [10].

The posterior of the latent functions $\mathbf{f}$ given all the preferences is:

$$p(\mathbf{f}|\mathcal{D}) = \frac{1}{Z} p(\mathbf{f}) p(\mathcal{D}|\mathbf{f}), \tag{4}$$

with Z being the normalizer. This posterior is analytically intractable due to the non-Gaussian nature of the likelihood. Therefore, we need to resort to approximations. Here we use EP which approximates the posterior $p(\mathbf{f}|\mathcal{D})$ by a tractable distribution $q(\mathbf{f})$. EP assumes that each likelihood term $p(\mathbf{x}_i \succ \mathbf{x}_j | f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$ can be approximated by a distribution $q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$ such that the approximated posterior $q(\mathbf{f})$ factorizes over $q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$. Then EP iteratively approximates each $q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$ in turn by dividing it out from the approximated posterior $q(\mathbf{f})$ (obtaining the cavity distribution), multiplying in the true likelihood $p(\mathbf{x}_i \succ \mathbf{x}_j | f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$, and projecting the result back to its factorized form by matching its moments to an updated $q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$. This overall procedure is motivated by the aim to minimize the KL−divergence between the true posterior $p(\mathbf{f}|\mathcal{D})$ and its approximation $q(\mathbf{f})$.

In the preference learning case we detailed earlier, we can approximate the posterior with a Gaussian:

$$q(\mathbf{f}) = \frac{1}{\tilde{Z}} p(\mathbf{f}) \prod_{\mathbf{u} \in U} \prod_{\{\{i,j\} | \mathbf{x}_i \succ \mathbf{x}_j \in \mathcal{D}^{\mathbf{u}}\}} q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) = \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{5}$$

We are interested in locally approximating each likelihood term in Equation 1 as:

$$\begin{aligned} p(\mathbf{x}_i \succ \mathbf{x}_j | f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) &\approx q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) \qquad (6)\\ &= \tilde{Z}_{i,j}^{\mathbf{u}} \mathcal{N}(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}; \tilde{\boldsymbol{\mu}}_{\mathbf{u},[i,j]}, \tilde{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}), \end{aligned}$$

where $\mathcal{N}(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}; \tilde{\boldsymbol{\mu}}_{\mathbf{u},[i,j]}, \tilde{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]})$ denotes the local two-dimensional Gaussian over $[f_i^{\mathbf{u}}, f_j^{\mathbf{u}}]^T$ with mean $\tilde{\boldsymbol{\mu}}_{\mathbf{u},[i,j]}$ and covariance $\tilde{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}$ corresponding to items i and j.

Hence we can approximate the posterior as:

$$q(\mathbf{f}) = \frac{1}{\tilde{Z}} p(\mathbf{f}) \prod_{\mathbf{u} \in U} \prod_{\{i,j\} \in \mathcal{D}} q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) = \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{7}$$

where

$$\boldsymbol{\mu}_{\mathbf{u},[i,j]} = \boldsymbol{\Sigma}_{\mathbf{u},[i,j]} \tilde{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}^{-1} \tilde{\boldsymbol{\mu}}_{\mathbf{u},[i,j]} \tag{8}$$

$$\boldsymbol{\Sigma}_{\mathbf{u},[i,j]}^{-1} = (\mathbf{K}_{\mathbf{u},[i,j]}^{-1} + \tilde{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}^{-1}). \tag{9}$$

This means that in order to determine the parameters of our approximate posterior, we need to compute estimates of the local parameters $\tilde{\boldsymbol{\mu}}$ and $\tilde{\boldsymbol{\Sigma}}$. To show these updates, we need to define additional distributions: (a) the *cavity* distribution which we will denote with the backslash symbol "$\backslash$" and (b) the *unnormalized marginal posterior*, which we will denote with the hat symbol " ˆ ".

Here we only show how to compute the parameters necessary to estimate the posterior[4]. We iterate through the following steps:

1. Update the cavity distribution: The cavity distribution $q_{\backslash}(f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$ results from multiplying the prior by all the local approximate likelihood terms except $q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$ and marginalizing all latent dimensions except $f_i^{\mathbf{u}}$ and $f_j^{\mathbf{u}}$. This is done in practice simply by removing the current approximate likelihood term from the approximate posterior. Hence we obtain:

[4] Similar updates for the single user case are given in [1].

$$q_{\backslash}(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) = \mathcal{N}(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}; \boldsymbol{\mu}_{\backslash \mathbf{u},[i,j]}, \boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}) \tag{10}$$

$$\boldsymbol{\mu}_{\backslash \mathbf{u},[i,j]} = \boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}(\boldsymbol{\Sigma}_{\mathbf{u},[i,j]}{}^{-1}\boldsymbol{\mu}_{\mathbf{u},[i,j]} - \tilde{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}^{-1}\tilde{\boldsymbol{\mu}}_{\mathbf{u},[i,j]}) \tag{11}$$

$$\boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]} = (\boldsymbol{\Sigma}_{\mathbf{u},[i,j]}^{-1} - \tilde{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}^{-1})^{-1}. \tag{12}$$

2. Update the unnormalized marginal posterior: This results from finding the unnormalized Gaussian that best approximates the product of the cavity distribution and the exact likelihood:

$$\hat{q}(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) \approx p(\mathbf{x}_i \succ \mathbf{x}_j | f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) q_{\backslash}(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) \tag{13}$$

$$\hat{q}(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) = \hat{Z}^{-1}\mathcal{N}(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}; \hat{\boldsymbol{\mu}}_{\mathbf{u},[i,j]}, \hat{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}) \quad \text{with} \tag{14}$$

$$\hat{Z} = \Phi(r_{i,j}) \tag{15}$$

$$\hat{\boldsymbol{\mu}}_{\mathbf{u},[i,j]} = \boldsymbol{\mu}_{\backslash \mathbf{u},[i,j]} + \boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}\mathbf{w}_{\mathbf{u},[i,j]} \tag{16}$$

$$\hat{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]} = \boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]} - \boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}(\mathbf{w}_{\mathbf{u},[i,j]}\mathbf{w}_{\mathbf{u},[i,j]}^{\top}\hat{r}_{i,j}\mathbf{w}_{\mathbf{u},[i,j]}\mathbf{1_1}^{\top})\boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}, \tag{17}$$

where

$$\mathbf{w}_{\mathbf{u},[i,j]} = \frac{\mathcal{N}(r_{i,j})}{\Phi(r_{i,j})(\alpha^2 + \mathrm{tr}(\boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}\mathbf{1_2}))}\mathbf{1_1},$$

$$r_{i,j} = \frac{\boldsymbol{\mu}_{\backslash \mathbf{u},[i,j]}\mathbf{1_1}}{\alpha^2 + \mathrm{tr}(\boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}\mathbf{1_2})}, \quad \hat{r}_{i,j} = \frac{r_{i,j}}{\alpha^2 + \mathrm{tr}(\boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}\mathbf{1_2})}$$

$$\text{and} \quad \mathbf{1_1} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \qquad \mathbf{1_2} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{18}$$

3. Update the local factor approximation: by performing moment matching, we can calculate the corresponding parameters in $q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$ as:

$$\tilde{\boldsymbol{\mu}}_{\mathbf{u},[i,j]} = \tilde{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}(\hat{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}^{-1}\hat{\boldsymbol{\mu}}_{\mathbf{u},[i,j]} - \boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}^{-1}\boldsymbol{\mu}_{\backslash \mathbf{u},[i,j]})$$

$$\tilde{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]} = (\hat{\boldsymbol{\Sigma}}_{\mathbf{u},[i,j]}^{-1} - \boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}^{-1})^{-1}. \tag{19}$$

At each iteration once we have local factor parameters $\tilde{\boldsymbol{\mu}}$ and $\tilde{\boldsymbol{\Sigma}}$, we can compute the parameters of the full posterior approximation using 7. We iterate through all the factors and update the local approximations sequentially.

2.1 Prediction

Given a pair of items $\mathbf{x}_1^*, \mathbf{x}_2^*$ for a particular user, we will be able to determine the predictive distribution over the latent utility functions as:

$$p(f_1^*, f_2^* | \mathcal{D}) = \int_{-\infty}^{\infty} p(f_1^*, f_2^* | \mathbf{f}) p(\mathbf{f}|\mathcal{D}) d\mathbf{f} = \mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{C}^*) \tag{20}$$

$$\text{with} \quad \boldsymbol{\mu}^* = \mathbf{K}^*(\mathbf{K} + \tilde{\boldsymbol{\Sigma}})^{-1}\boldsymbol{\mu} \tag{21}$$

$$\boldsymbol{C}^* = \boldsymbol{\Sigma}^* - \mathbf{K}^{*\top}(\mathbf{K} + \tilde{\boldsymbol{\Sigma}})^{-1}\mathbf{K}^*, \tag{22}$$

where $\boldsymbol{\Sigma}^*$ is the 2×2 kernel matrix built from the item pair $\mathbf{x}_1^*$ and $\mathbf{x}_2^*$; $\mathbf{K}^* = \mathbf{K}_u^* \otimes \mathbf{K}_x^*$ that represents the kernel matrix of the test user and items with all the users and items in the training set; $\mathbf{K}_u^*$ is the $1 \times n$ kernel matrix of the queried user with other users; and $\mathbf{K}_x^*$ is the $2 \times m$ kernel matrix of the queried pair of items with other items. Subsequently, their preference for a user is determined by integrating out the latent utility functions:

$$\begin{aligned} p(\mathbf{x}_1^* \succ \mathbf{x}_2^* | \mathcal{D}) &= \int \int p(\mathbf{x}_1^* \succ \mathbf{x}_2^* | f_1^*, f_2^*, \mathcal{D}) p(f_1^*, f_2^* | \mathcal{D}) df_1^* df_2^* \\ &= \Phi \left(\frac{\boldsymbol{\mu}_1^* - \boldsymbol{\mu}_2^*}{\alpha^2 + \boldsymbol{C}_{1,1}^* + \boldsymbol{C}_{2,2}^* - 2\boldsymbol{C}_{1,2}^*} \right). \end{aligned} \tag{23}$$

We see that the mean and covariance of the predictive distribution require the inversion of a (possibly) very large matrix. This matrix is, in general, of dimensions $nm \times nm$. Even though the inverse matrix can be reused for multiple query points, this is intractable for any real application. Hence, we will focus on how to sparsify this matrix by selecting a subset of *inducing items*. Note the main difference with other machine learning settings where there is a one-to-one correspondence between the number of observations and the dimensionality of the corresponding matrix. In our case, the observations (preference relations) affect the dimensionality of this matrix only indirectly and we are more concerned with the number of users and items. More importantly, as we shall see in the following section, we will make use of decision theory for sparsification. Our method, which we will refer to as the Valuable Vector Machine (VVM), selects the most useful items during sparsification so as to minimize the loss inherent to the preference learning problem. Therefore, the prediction time which is cubic in the number of items is improved in VVM over the case where all items are used.

Since our focus is on improving prediction time and this scales cubically with the number of items we need to obtain a risk-sensitive posterior approximation. EP is well-suited for this case because it considers all data efficiently and locally.

2.2 Optimizing the Kernel Hyper-parameters

One of the inherent advantages of GPs over other non-Bayesian kernel methods is its capability of optimizing the hyper-parameters. This can be easily done by maximizing the marginal likelihood in a gradient descent algorithm. The marginal likelihood can be obtained from the normalizer $\tilde{Z}$ in Equation 5 as:

$$\tilde{Z} = \int p(\mathbf{f}) \prod_{\mathbf{u} \in U} \prod_{\{i,j\} \in \mathcal{D}} q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}}) d\mathbf{f} \tag{24}$$

where both $p(\mathbf{f})$ and $q(f_i^{\mathbf{u}}, f_j^{\mathbf{u}})$ are Gaussian distributions and their product produces an unnormalized Gaussian distribution. Therefore, the log likelihood is:

$$\log(\tilde{Z}) = -\frac{1}{2} \tilde{\boldsymbol{\mu}}^\top (\mathbf{K} + \tilde{\boldsymbol{\Sigma}})^{-1} \tilde{\boldsymbol{\mu}} - \frac{1}{2} \log \det(\mathbf{K} + \tilde{\boldsymbol{\Sigma}}) - \frac{n}{2} \log 2\pi \tag{25}$$

Table 1. Loss and corresponding risk to minimize for VVM variants. Let $q(\mathbf{f}) := q_{S''}(\mathbf{f})$ and $a \in \{\mathbf{x}_i^{\mathbf{u}}\}$.

Algorithm	IVM (item)	VVM-VOI	VVM-UCB
Loss Type	Log loss	Regret	Risk-seeking
$\mathcal{L}(\mathbf{f}, \mathbf{a}, \mathbf{u})$	$-\log(q(f_i^{\mathbf{u}}))$	$-\mathbb{I}[f_i^{\mathbf{u}} > f^{\mathbf{u},*}](f_i^{\mathbf{u}} - f^{\mathbf{u},*})$	$-\exp(\beta f_i^{\mathbf{u}}), \beta > 0$
$Risk_{\mathcal{L}}(\mathbf{S}', \mathbf{x_i}, \mathbf{u})$	$H(q(f_i^{\mathbf{u}}))$	$\sigma(\mathbf{u}, \mathbf{x}_i)\,[c\Phi(c) + \mathcal{N}(c)]$	$1 + \beta\mu(\mathbf{u}, \mathbf{x}_i) + \frac{\beta^2}{2}\bar{\sigma}^2(\mathbf{u}, \mathbf{x}_i)$
Selection Time	$\mathcal{O}(1)$	$\mathcal{O}(n \log n)$	$\mathcal{O}(1)$

The derivative of the marginal likelihood with respect to the kernel hyper-parameters can be used in a gradient descent algorithm to optimize the kernel.

3 Decision-theoretic Sparsification

To recap, our multi-user preference learning objective is to approximate a posterior $q(\mathbf{f}) = \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ over latent utilities $\mathbf{f} = [f_1^{\mathbf{u}_1}, f_2^{\mathbf{u}_1}, \ldots, f_m^{\mathbf{u}_n}]^T$ for users $\mathbf{u} \in U$ and items $\mathbf{x}_i \in X$. In the previous section, we showed how to learn $q(\mathbf{f})$ from preference data by EP; in this section due to computational considerations, we wish to sparsify this Gaussian posterior in a loss-calibrated manner. We note that in the special case of GP-based preference learning, there are at least two different ways one might approach sparsification: observation-driven sparsification and item-driven sparsification.

3.1 Sparsification

Observation-driven Sparsification In this approach, we incrementally select a subset of observations (in this case preferences) in order to approximate the posterior $q(\mathbf{f})$. More formally, recall that $\mathcal{D}^{\mathbf{u}} = \{\mathbf{x}_i \succ \mathbf{x}_j\}$ and let $\mathcal{D}' \subseteq \mathcal{D}$ be a subset of selected preferences. Observation-driven sparsification simply chooses the data subset $\mathcal{D}'$ according to some criterion to obtain a posterior approximation $q_{\mathcal{D}'}(\mathbf{f}) \approx p(\mathbf{f}|\mathcal{D}')$ (e.g., via EP as outlined in the last section).

As a concrete example, the original *Informative Vector Machine* [8] initializes $\mathcal{D}'$ to a small random subset and then incrementally builds $\mathcal{D}' := \mathcal{D}' \cup \{d^*\}$ for the d^* that maximizes information gain

$$d^* = \operatorname*{arg\,max}_{d \in \mathcal{D} \setminus \mathcal{D}'} H(q_{\mathcal{D}' \cup \{d\}}(\mathbf{f})) - H(q_{\mathcal{D}'}(\mathbf{f})), \tag{26}$$

where $q_{\mathcal{D}'}(\mathbf{f}) \approx p(\mathbf{f}|\mathcal{D}')$ and $q_{\mathcal{D}' \cup \{d\}}(\mathbf{f}) \approx p(\mathbf{f}|\mathcal{D}' \cup \{d\})$. This repeats until the desired level of observation sparsity has been reached. Since $\mathcal{D}'$ is fixed at each iteration and thus $H(q_{\mathcal{D}'}(\mathbf{f}))$ is a constant, each incremental selection in the IVM is equivalent to choosing the d^* that maximizes entropy, i.e., $d^* = \arg\max_{d \in \mathcal{D} \setminus \mathcal{D}'} H(q_{\mathcal{D}' \cup \{d\}}(\mathbf{f}))$.

Item-driven Sparsification: Valuable Vector Machine Inclusion of a preference observation entails a 2-dimensional update to our GP posterior; however, since preferences may overlap, there is not a direct relationship between the number of included preferences and the dimensionality of the posterior and hence the computational complexity of prediction described in section 2.1. A more direct way to control the sparsity level of our Gaussian posterior is to simply retain the *items* for users (equivalently dimensions $f_i^{\mathbf{u}}$ of our Gaussian posterior) that minimizes some criterion.

This item-driven approach underlies the *Valuable Vector Machine* (VVM) that we propose in this paper for different decision-theoretic settings. First we introduce some notation. Let $S' \subseteq S = \{(\mathbf{x}_i, \mathbf{u})\}$ be a selected subset of user-item pairs corresponding to latent utility dimensions $f_i^{\mathbf{u}}$ of $\mathbf{f}$ with cardinality $|S'|$. Let $q(\mathbf{f}_{[S']}) = \mathcal{N}(\mathbf{f}_{[S']}; \boldsymbol{\mu}_{[S']}, \boldsymbol{\Sigma}_{[S',S']})$ where $\mathbf{f}_{[S']}$ and $\boldsymbol{\mu}_{[S']}$ respectively represent the subvectors of $\mathbf{f}$ and $\boldsymbol{\mu}$ for selected dimensions S' (i.e. selected users and items in set S') and $\boldsymbol{\Sigma}_{[S',S']}$ the corresponding submatrix of $\boldsymbol{\Sigma}$. Motivated by the observation-driven IVM, after running EP, let us incrementally select dimensions $s^* \in S$ of our Gaussian posterior to retain so that initializing $S' = \emptyset$, at each iteration we update $S' := S' \cup \{s^*\}$ to obtain an improved posterior $q_{S'}(\mathbf{f})$ until some dimensionality limit has been reached.

In decision theory, our objective is to select an action $a^* \in A$ from a possible space of actions A so as to minimize the expectation of some loss $\mathcal{L}(a)$ w.r.t. uncertainty (in this case utility uncertainty over $\mathbf{f}$), i.e. $a^* = \arg\min_a \mathbb{E}_{\mathbf{f}}\mathcal{L}(\mathbf{f}, a)$. Our specific task at each iteration of the VVM is to propose an item-user dimension $\mathbf{x}_i^{\mathbf{u}}$ for inclusion in the posterior — hence the action space $A = \{\mathbf{x}_i\}$ — and to select the item s^* that minimizes expected loss (risk)

$$\begin{aligned} s^* &= \mathop{\arg\min}_{(\mathbf{x}_i,\mathbf{u})\in S\setminus S'} \mathit{Risk}_{\mathcal{L}}(S', \mathbf{x}_i, \mathbf{u}); \\ &\text{where } \mathit{Risk}_{\mathcal{L}}(S', \mathbf{x}_i, \mathbf{u}) := \mathbb{E}_{\mathbf{f}\sim q_{S''}}\left[\mathcal{L}(\mathbf{f}, \mathbf{x}_i, \mathbf{u})\right], \end{aligned} \tag{27}$$

and $S'' = S' \cup \{\mathbf{x}_i^{\mathbf{u}}\}$. In the following, we will detail choices of loss functions and their respective $\mathit{Risk}_{\mathcal{L}}(S', \mathbf{x}_i, \mathbf{u})$ yielding the VVM variants as summarized in Table 1 and its corresponding method in Algorithm 1.

In each iteration, VVM selects the action (i.e. item) that minimizes the expected loss for each user until desired predefined dimensionality is reached. Our experiments with a variable number of items per user led to worse performance since it often overemphasizes item selection for the noisiest users. Hence, we found that a constant number of items enforces fairness of GP modeling effort per user.

It should also be noted that the greedy selection here is fairly general and in special cases such as submodular losses, one can prove further convergence guarantees [14].

3.2 Loss Functions and Risk

Log loss and IVM Log-loss is appropriate when we want to maximize the log posterior over all preferences. Here we see that we can actually recover an

Algorithm 1 Valuable Vector Machine

```
input: $X, U, \mathcal{D}, r$ // $r$ is the percentage of items selected for each user
while not converged do
    for $\mathbf{x}_i \succ \mathbf{x}_j \in \mathcal{D}$ do
        1. Update the cavity distribution $\boldsymbol{\mu}_{\backslash \mathbf{u},[i,j]}$, $\boldsymbol{\Sigma}_{\backslash \mathbf{u},[i,j]}$ from Equation 11 and 12.
        2. Update the unnormalized marginal posterior $\hat{\boldsymbol{\mu}}$,$\hat{\boldsymbol{\Sigma}}$ from Equation 16 and 17.
        3. Update the local factor approximation $\tilde{\boldsymbol{\mu}}$, $\tilde{\boldsymbol{\Sigma}}$ from Equation 19.
        5. Update $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ from Equation 7.
    end for
end while
for each $\mathbf{u} \in U$ // Selection of best items for each user do
    $S'^{\mathbf{u}} = \{\}$ //The user $\mathbf{u}$'s subset
    while $|S'^{\mathbf{u}}| < r$ do
        $\mathbf{x}_i^* = \arg\min_{\mathbf{x}_i \in X, \mathbf{x}_i \notin S'^{\mathbf{u}}} Risk_{\mathcal{L}}(S', \mathbf{x}_i, \mathbf{u})$// Table 1
        $S'^{\mathbf{u}} = S'^{\mathbf{u}} \cup \{\mathbf{x}_i^*\}$
    end while
    $S = S \cup S'^{\mathbf{u}}$
end for
return $\boldsymbol{\mu}_{[S]}, \boldsymbol{\Sigma}_{[S,S]}$ // Subset of posterior parameters
```

item-based variant of the IVM when using log-loss. Specifically, letting $q(f_i^{\mathbf{u}})$ refer to the marginal of $q(\mathbf{f})$ over $f_i^{\mathbf{u}}$ then

$$\begin{aligned} Risk_{\mathcal{L}}(S', \mathbf{x}_i, \mathbf{u}) &= \int_{-\infty}^{\infty} -q_{S''}(f_i^{\mathbf{u}})[\log q_{S''}(f_i^{\mathbf{u}})] df_i^{\mathbf{u}} \\ &= H(q_{S''}(f_i^{\mathbf{u}})), \end{aligned} \tag{28}$$

which corresponds to the entropic criterion used by the IVM. Recall that the second entropy term in the standard IVM information gain calculation is constant and can be omitted as noted for (26).

Valuable Vector Machine – Value Of Information (VVM-VOI) In the case that our end objective is to predict or recommend the best item $\mathbf{x}_i$ for user $\mathbf{u}$, the loss we might consider minimizing is the *regret*, $\mathbb{I}[f_i^{\mathbf{u}} - f^{\mathbf{u},*} > 0](f_i^{\mathbf{u}} - f^{\mathbf{u},*})$ where we could define $f^{\mathbf{u},*} = \arg\max_i f_i^{\mathbf{u}}$; in words, we want to minimize how much utility we lose for recommending a suboptimal item. In expectation, we might simply fix $f^{\mathbf{u},*} = \max_i \mathbb{E}_{q_{S''}}[f_i^{\mathbf{u}}]$ (the best current item in expectation) where expected loss minimization leads us to the following risk:

$$\begin{aligned} Risk_{\mathcal{L}}(S', \mathbf{x}_i, \mathbf{u}) &= \int_{-\infty}^{\infty} \mathbb{I}[f_i^{\mathbf{u}} > f^{\mathbf{u},*}](f_i^{\mathbf{u}} - f^{\mathbf{u},*}) q_{S''}(f_i^{\mathbf{u}}) df_i^{\mathbf{u}} \\ &= \underbrace{\sigma(\mathbf{u}, \mathbf{x}) \left[c\Phi(c) + \mathcal{N}(c)\right]}_{VOI} \end{aligned} \tag{29}$$

where $q_{S''}(f_i^{\mathbf{u}}) = \mathcal{N}(f_i^{\mathbf{u}}; \mu(\mathbf{u}, \mathbf{x}), \sigma^2(\mathbf{u}, \mathbf{x}))$ and $c = \frac{\mu(\mathbf{u},\mathbf{x}) - \hat{f^{\mathbf{u}}}}{\sigma(\mathbf{u},\mathbf{x})}$. This is precisely the statement of *Value of Information* (VOI) [15] under a Gaussian assumption of uncertainty — quite simply, the more probability mass an item utility has in its tail above the best item in expectation, the higher its chance of being the best item — hence the higher VOI associated with selecting this item in S''.

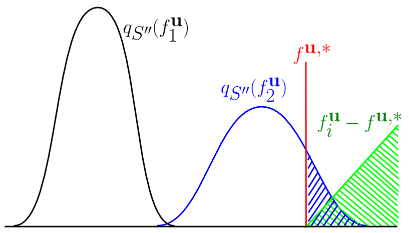

$q_{S''}(f_1^{\mathbf{u}})$
$f^{\mathbf{u},*}$
$q_{S''}(f_2^{\mathbf{u}})$
$f_i^{\mathbf{u}} - f^{\mathbf{u},*}$

Fig. 1. Illustration of Value of Information: it is the product of the shaded area under the normal curve when the utility is higher than the optimal value and the linear function of their difference. This value corresponds to the expectation of the difference of the utility of the item and the optimal under the shaded mass. As it is observed, $f_1^{\mathbf{u}}$ has negligible mass above $f^{\mathbf{u},*}$ point, therefore the item corresponding to $f_2^{\mathbf{u}}$ is selected.

Valuable Vector Machine – Upper Confidence Bound (VVM-UCB) It is well-known that a concave or convex valuation of underlying utility respectively encourages risk-averse or risk-seeking behavior w.r.t. utility uncertainty. Risk-seeking behavior from a convex utility function will encourage including "potentially optimal" items according to how uncertain we are regarding their utility function value. A natural convex utility transformation is $\exp(\beta f_{\mathbf{x}}^{\mathbf{u}})$, which leads to the following risk

$$\begin{aligned} Risk_{\mathcal{L}}(S', \mathbf{x}_i, \mathbf{u}) &= -\int_{-\infty}^{\infty} q_{S''}(f_i^{\mathbf{u}}) \exp(\beta f_i^{\mathbf{u}}) df_i^{\mathbf{u}} \\ &= -\int_{-\infty}^{\infty} q_{S''}(f_i^{\mathbf{u}}) \left(1 + \beta f_i^{\mathbf{u}} + \frac{\beta^2}{2} f_i^{\mathbf{u}2} + \dots \right) df_i^{\mathbf{u}} \\ &\approx 1 + \beta \cdot UCB(\mathbf{u}, \mathbf{x}_i) \\ \text{where } UCB(\mathbf{u}, \mathbf{x}_i) &= \mu(\mathbf{u}, \mathbf{x}_i) + \frac{\beta}{2} \bar{\sigma}^2(\mathbf{u}, \mathbf{x}_i). \end{aligned} \tag{30}$$

where $\bar{\sigma}^2(\mathbf{u}, \mathbf{x}_i) = \mathbb{E}_{q_{S''}}[f_i^{\mathbf{u}2}]$. Here, we first replaced $\exp(\beta f_i^{\mathbf{u}})$ with its Taylor expansion and approximated it by truncating third-order terms and above. When doing this, we see that the dimension $\mathbf{x}_i^u$ selected by the VVM will be the one with the greatest *Upper Confidence Bound* (UCB) [16] used in bandit problems, where larger $\beta > 0$ encourages more risk-seeking behavior.

4 Empirical Evaluation

In this section we evaluate the performance of our algorithms (VVM-VOI and VVM-UCB) compared to the IVM and the full GP, i.e. a GP-preference model that does not use sparsification, in terms of two losses: the 0/1 loss and recommendation loss. The *0/1 loss* is the percentage of incorrectly predicted preferences and the *recommendation loss* is the proportion of items that are incorrectly predicted as the best for recommendation. In other words, if the set of items that a user considers to be the best (as induced by her preferences) is denoted by T and the predicted set of best items is T^*, the recommendation loss is $\frac{|\bar{T}|}{m} \times 100$, where m is the number of items and $\bar{T} = \{\mathbf{x} \in T^* | \mathbf{x} \notin T\}$. We report the results as a function of the proportion of items selected for sparsification.

Our experimental rationale is to exhibit how different risk-sensitive sparsifications perform across two different important losses related to preference learning compared to IVM. As such, we use IVM in its original form that works with ADF since it was argued by [17] that it performs better than running full EP, which we observed as well. Hence we chose the IVM variant that offered best performance and compared it against VVM.

We consider three datasets: a synthetic dataset and two real-world datasets. The synthetic experiment assesses the effectiveness of our approach in a controlled setting and the real-world datasets include users' preferences over cars that we have collected using Amazon Mechanical Turk and a Facebook dataset that we have obtained via an in-house application that collects user preferences over web links.

In all these datasets we are given a set of users and items and their corresponding features along with each user's preferences over item pairs. For each user and item we augment their feature vectors with their ID index (this is a common practice in collaborative filtering) and transform their categorical features into binary variables. We split each user's set of preferences into 60% for training and 40% for testing.

We use the squared exponential covariance kernel with automatic relevance determination (ARD) (see [18], Page 106) for both users and items and optimize the hyper-parameters by maximizing the marginal likelihood under the EP approximation as detailed in section 2.2. Finally, we have set $\alpha = 3$ (see Equation (2)) and $\beta = 1$ (see Equation (30)) for all experiments on all datasets.

4.1 Datasets

Synthetic Dataset: In this experiment we created a synthetic dataset where the utility function value for each item is known beforehand and is subsequently used to generate users' preferences. A set of hypothetical users and items are created and identified by their IDs. For each user, items are randomly split into two sets to indicate the ones that are liked (with a constant utility value of 10) and disliked (with a constant utility value of 5). From these utility functions we generate full sets of preferences for 10 items and 50 users. Consequently, for each user, 5 items have higher utility value and are naturally preferred to the other half.

Facebook Data: This dataset has been created using a Facebook App that recommends web links to users every day. The users may give their feedback on the links indicating whether they liked/disliked them. At its peak usage, 111 users had elected to install the Facebook app developed for this project. We also collected user information consisting of ID, age and gender and the link features including count of link's total "likes", count of link's "shares" and count of total link comments. The Facebook App recommended three links per day to avoid position bias and information overload. The preference set is built such that the links that are liked are considered *preferred* to the ones disliked in the batch of three recommended each day. We used 20% of users with the highest number of preferences over 50 links commonly recommended to all users.

Car Preference Dataset using Amazon Mechanical Turk: We set up an experiment using Amazon Mechanical Turk[5] (AMT) to collect real pair-wise preferences over users. In this experiment users are presented with a choice of a car over another based on their attributes. The car attributes used are (1) body type: sedan, SUV, hatchback, (2) engine capacity: 2.5L, 3.5L, 4.5L, etc, (3) transmission: manual, automatic, (4) fuel consumption: hybrid, non-hybrid, and (5) engine/transmission layout: all-wheel-drive (AWD), forward-wheel-drive (FWD). The dataset has been collected so that 20 unique cars (items) are considered but users are required to answer only 20% of all 190 possible pair-wise preferences. We targeted US users mainly to have a localized preference dataset. For all 60 unique users that participated in this experiment, a set of attributes in terms of general questions (age range, education level, residential region and gender) has been collected as user features.

4.2 Results

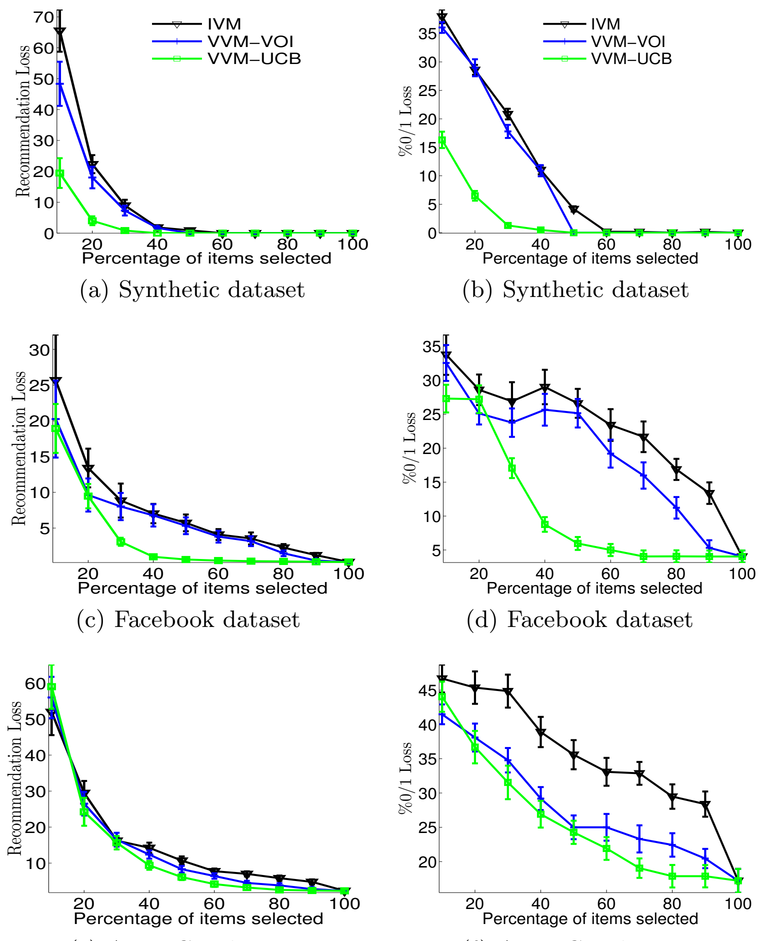
We evaluate our algorithms in a cross-validation setting using 60% of preferences for training and 40% for testing and repeated each experiment 40 times. Results are averaged over the number of test users. We analyze the performance of the algorithms as a function of the level of sparsification, as given by the percentage of items selected for inference. The larger the percentage of items selected, the smaller level of sparsification and the closer the algorithms are to the Full-GP method. The performances of the different algorithms on all datasets using the recommendation loss and the 0/1 loss are shown in Figure 2.

Figures 2(a) and 2(b) show the results on the synthetic dataset. Because of the clear distinction between the items that are preferred for each user, all algorithms perform very well when using at least 40% of the items. While IVM and VVM-VOI have very similar performance, VVM-UCB's performance is outstanding, requiring only a very small number of items to achieve perfect prediction.

As seen in Figures 2(c) and 2(d), on the Facebook dataset both VVM-VOI and VVM-UCB outperform (or have equal performance to) IVM when using at least 30% of the items.

It is interesting to note here that the risk-seeking behavior of VVM-UCB leads to a better approximation of the Full-GP which is particularly visible in the Facebook dataset where the number of items are larger. We conjecture that the excellent performance of VVM-UVB with this larger number of items is because it manages to quickly find and refine the set of highest value items, more effectively than even VVM-VOI. This simultaneously lowers recommendation

[5] http://mturk.com



(a) Synthetic dataset (b) Synthetic dataset

(c) Facebook dataset (d) Facebook dataset

(e) AMT Car dataset (f) AMT Car dataset

Fig. 2. Performance of the sparsification methods in terms of the *recommendation loss* (the proportion of items that are incorrectly predicted as the best item for recommendation) in the first column and the 0/1 *loss* (percentage of wrongly predicted preferences) in the second, as a function of the proportion of items selected for sparsification. The larger the number of items the lower the level of sparsification and the closer the algorithms are to the Full-GP method.

loss by finding a near-optimal item and 0/1 loss since the best items can then be identified with certainty in most pairwise comparisons.

Figures 2(e) and 2(f) show the results on the AMT Car dataset. In this dataset, where true preferences have been collected, VVM-VOI and VVM-UCB consistently outperform IVM. Similar to the Facebook results, we conjecture that VVM-VOI's and VVM-UCB's better performance than the IVM (most notably on 0/1 loss where all preferences matter) stems from the fact that they both select the potentially best items first and this helps identify the dominant item in all pairwise preferences. However it seems that identifying the single best item among the potentially best items is difficult in this particular dataset, requiring a large proportion of data to identify the best item with high accuracy.

It is interesting to mention that while VVM-VOI and VVM-UCB outperform IVM in most cases when using the recommendation loss, a similar trend is seen when using the 0/1 loss. Although this result may look unexpected, it is important to emphasize that neither the VVM or the IVM are designed to optimize the 0/1 loss. In fact, the risk-seeking nature of the VVM-UCB loss, as a consequence of the exponential transformation of the utility functions, may be better aligned with the 0/1 loss than the entropic criterion used by the IVM.

Another issue worth mentioning is the computational cost of running the different approximation algorithms. As a reference of the time spent by our algorithms compared to the Full-GP (where no sparsification is done), Figure 3 shows the prediction time for an indicative experiment. We see that – while IVM and VVM-UCB may enjoy very similar prediction time and similar structure in the posterior – sparsification improves prediction time significantly and that all approximation algorithms have roughly the same computational cost. Small variations as that observed when using 80% of the items can be explained by the different sparsity properties of the posterior covariance obtained when selecting a distinct subset of items.

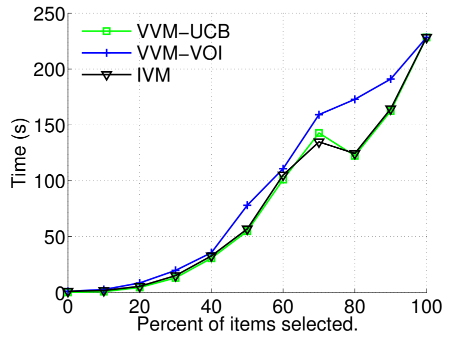


Fig. 3. Average prediction time for inference with 200 users and 10 items. The number calculated as the time consumed to make a series of predictions on the preferences of the test set.

5 Related Work

Probabilistic models for utility functions in preference learning and elicitation have previously been proposed in the machine learning community (e.g. [19,20]). Extensions to non-parametric models have also been developed. In particular, [21] proposed a preference learning framework based on Gaussian processes and [22] used this framework for active learning with discrete choice data. Multi-user GP-based preference models have been given by [23] and [2]. Our method builds upon the model proposed by [2], where the Laplace method was used to approximate the posterior.

In standard machine learning settings, low-rank approximations to the Gram matrix are commonly used by practitioners and researchers (see Chapter 8 of [18] for an example). to deal with large datasets. A unifying framework in which most of these approximations can be formulated has been given by [24]. This framework also includes fully independent training conditional (FITC) that makes better use of all the data that can be combined with our approach to approximate the covariance at a higher cost. The work proposed in [7] considers sparsification approaches where the inducing points are latent variables and their values are optimized within a consistent probabilistic latent variable model. However, none of these algorithms addresses the sparsification problem from a decision-theoretic perspective.

Our approach is analogous to the IVM in that we borrow ideas from active learning in order to carry out sparsification during approximate inference. For example, upper confidence bounds (UCB) are used in [12] for GP optimization within the bandit setting. We note that, unlike this latter experimental design scenario, in our sparsification framework we see the data beforehand and decide to include it in our approximation afterwards.

An information theoretic algorithm that resembles IVM is also proposed in [25] where the log loss is considered. However this method, like the IVM, does not include the task loss and it is used in an active learning setting. Although casting preference learning as a classification problem and using FITC's idea to obtain a lower dimensional covariance allows us to devise an efficient algorithm for multi-user preference learning [26].

The most relevant work to ours has been recently proposed in [11] where the use of loss functions in Bayesian methods is considered by formulating an EM algorithm that alternates between variational inference and risk minimization. We take the idea of bridging the gap between decision theory and approximate Bayesian inference [11] in a direction that leverages the efficiency of the IVM approach for GP sparsification, while overcoming its loss-insensitive approximation.

6 Conclusion

We proposed a decision-theoretic sparsification method for Gaussian process preference learning. We referred to our method as the valuable vector machine (VVM) to emphasize the importance of considering a loss-sensitive sparsification approach. We show that the IVM's differential entropy criterion, a value of information criterion, and an upper confidence bound (UCB) criterion can *all* be recovered in a generalized decision-theoretic framework by specifying the appropriate loss. Overall, our approach contributes to the goal of bridging the gap between decision theory and approximate Bayesian inference in the context of loss-sensitive sparsification approaches for efficient Gaussian Process preference learning.

References

1. Chu, W., Ghahramani, Z.: Extensions of gaussian processes for ranking: semi-supervised and active learning. In: Workshop on Learning to Rank at NIPS. (2005)

2. Bonilla, E.V., Guo, S., Sanner, S.: Gaussian process preference elicitation. In: NIPS. (2010) 153–160
3. Platt, J.C., Burges, C.J.C., Swenson, S., Weare, C., Zheng, A.: Learning a Gaussian Process Prior for Automatically Generating Music Playlists. In: NIPS. (2002)
4. Xu, Z., Kersting, K., Joachims, T.: Fast active exploration for link-based preference learning using Gaussian processes. In: ECML PKDD. (2010)
5. Smola, A.J., Bartlett, P.: Sparse greedy Gaussian process regression. In: NIPS. (2001)
6. Keerthi, S.S., Chu, W.: A Matching Pursuit Approach to Sparse Gaussian Process Regression. In: NIPS. (2005)
7. Snelson, E., Ghahramani, Z.: Sparse Gaussian processes using pseudo-inputs. In: NIPS. (2006)
8. Lawrence, N., Seeger, M., Herbrich, R.: Fast sparse Gaussian process methods: The informative vector machine. In: NIPS. (2003)
9. Minka, T.P.: A family of algorithms for approximate bayesian inference. PhD thesis, Massachusetts Institute of Technology (2001)
10. Minka, T.P.: Expectation propagation for approximate Bayesian inference. In: UAI. Volume 17. (2001) 362–369
11. Lacoste-Julien, S., Huszar, F., Ghahramani, Z.: Approximate inference for the loss-calibrated Bayesian. In: AISTATS. (2011) 416–424
12. Srinivas, N., Krause, A., Kakade, S.M., Seeger, M.: Gaussian process optimization in the bandit setting: No regret and experimental design. In: NIPS. (2009)
13. Neumann, J.V., Morgenstern, O.: Theory of Games and Economic Behavior. Princeton University Press (1944)
14. Krause, A., Golovin, D.: Submodular Function Maximization. In: Tractability: Practical Approaches to Hard Problems. Cambridge University Press (2012)
15. Howard, R.: Information value theory. Systems Science and Cybernetics, IEEE Transactions on **2**(1) (aug. 1966) 22 –26
16. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Mach. Learn. **47**(2-3) (May 2002) 235–256
17. Seeger, M.W.: Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations. PhD thesis, University of Edinburgh (2003)
18. Rasmussen, C.E., Williams, C.: Gaussian Processes for Machine Learning. (2006)
19. Chajewska, U., Koller, D.: Utilities as random variables: Density estimation and structure discovery. In: UAI, Morgan Kaufmann Publishers Inc. (2000) 63–71
20. Guo, S., Sanner, S.: Real-time multiattribute Bayesian preference elicitation with pairwise comparison queries. In: AISTATS. (2010)
21. Chu, W., Ghahramani, Z.: Preference learning with Gaussian processes. In: ICML, New York, NY, USA, ACM (2005) 137–144
22. Eric, B., Freitas, N.D., Ghosh, A.: Active preference learning with discrete choice data. In: NIPS. (2008) 409–416
23. Birlutiu, A., Groot, P., Heskes, T.: Multi-task preference learning with an application to hearing aid personalization. In: Neurocomputing. Volume 73. (2010)
24. Quiñonero-Candela, J., Rasmussen, C.: A unifying view of sparse approximate Gaussian process regression. In: JMLR. (2005) 1939–1959
25. Houlsby, N., Huszr, F., Ghahramani, Z., Lengyel, M.: Bayesian Active Learning for Classification and Preference Learning. In: arXiv:1112.5745. (2011)
26. Houlsby, N., Hernndez-Lobato, J.M., Huszar, F., Ghahramani, Z.: Collaborative Gaussian Processes for Preference Learning. In: NIPS. (2012)