Expectation Propagation for Bayesian Multi-task Feature Selection

Daniel Hernández-Lobato[1], José Miguel Hernández-Lobato[2], Thibault Helleputte[1], and Pierre Dupont[1]

[1] Machine Learning Group, ICTEAM institute,
Université catholique de Louvain,
Place Sainte Barbe 2, B-1348 Louvain-la-Neuve, Belgium,
{daniel.hernandez-lobato, thibault.helleputte,
pierre.dupont}@uclouvain.be,
[2] Computer Science Department, Universidad Autónoma de Madrid,
C/ Francisco Tomás y Valiente, 11 28049 Madrid, Spain,
josemiguel.hernandez@uam.es

Abstract. In this paper we propose a Bayesian model for multi-task feature selection. This model is based on a generalized *spike and slab* sparse prior distribution that enforces the selection of a common subset of features across several tasks. Since exact Bayesian inference in this model is intractable, approximate inference is performed through expectation propagation (EP). EP approximates the posterior distribution of the model using a parametric probability distribution. This posterior approximation is particularly useful to identify relevant features for prediction. We focus on problems for which the number of features d is significantly larger than the number of instances for each task. We propose an efficient parametrization of the EP algorithm that offers a computational complexity linear in d. Experiments on several multi-task datasets show that the proposed model outperforms baseline approaches for single-task learning or data pooling across all tasks, as well as two state-of-the-art multi-task learning approaches. Additional experiments confirm the stability of the proposed feature selection with respect to various sub-samplings of the training data.

Key words: Multi-task learning, feature selection, expectation propagation, approximate Bayesian inference.

1 Introduction

The automatic induction of a predictor for a dependent variable y given a feature vector $\mathbf{x}$ can be a difficult task when the number of training instances is very small and the number of explanatory variables is large. Examples of learning applications with these characteristics include, among others, the classification of microarray data [1] or the analysis of high-dimensional images [2]. Under these circumstances, an underlying linear model is often considered, possibly in an expanded feature space. A potential way of improving the robustness of these

models is to assume that only a small subset of the original features are relevant for prediction [3]. That is, the underlying linear model is assumed to be sparse with many coefficients being equal to zero. The identification of the relevant features is typically implemented by optimizing an objective function penalized by a sparsity enforcing regularizer. Such a regularizer drives to zero some of the coefficients during the optimization process. A common choice is the ℓ_1 norm of the vector of model coefficients [4]. Within a Bayesian framework, sparsity can be favored by considering sparse prior distributions for the coefficients of the linear model. Examples of such priors include the Student's t distribution, the Laplace [2] and the *spike and slab* [5]. Among these, the *spike and slab* is the only prior that can assign a non-zero probability to solutions with many coefficients being equal to zero. Under this prior it is furthermore possible to specify intuitively the fraction of coefficients that are *a priori* different from zero. The *spike and slab* prior also provides a selective shrinkage of the model coefficients [6]: for high sparsity levels, most of the coefficients are close to zero while a few of them have significantly larger values. By contrast, other priors shrink towards zero *all* the coefficients when the sparsity level is increased. Since exact Bayesian inference is typically intractable under sparse priors, approximate algorithms have to be used in practice.

We address here prediction problems for which the number of instances is typically small. In such cases, it may be beneficial for the induction to rely on several distinct but related tasks. Microarray datasets offer examples of such tasks for which the common objective is typically to discriminate between normal and tumor samples, while tissue and RNA extraction protocols may differ across tasks. Specifically, the multi-task approach proposed by Obozinski *et al.* assumes that the distinct tasks share a reduced set of common relevant features [7]. They propose to solve an optimization problem that minimizes a logistic loss function combined with a term that penalizes the ℓ_1 norm of the vector of ℓ_2 norms of the feature-specific coefficient vectors across the different tasks. Such a mixed norm regularization drives to zero the same coefficients of the task-specific vectors during the optimization process. This favors the selection of a common set of features to describe each task. The amount of sparsity is determined in this model by a hyper-parameter λ which has to be tuned by cross validation. In particular, [7] gives an efficient path-following algorithm to find the potential values of λ.

In a different work, Evgeniou and Pontil consider that the hyperplanes of the distinct tasks are the linear combination of a common hyperplane and a task-specific hyperplane [8]. They specifically propose to minimize a hinge loss function that is penalized by two terms. The first term is proportional to the hyper-parameter λ_1 and penalizes the squared values of the ℓ_2 norms of the task-specific hyperplanes. The second term is proportional to the hyper-parameter λ_2 and penalizes the squared value of the ℓ_2 norm of the common hyperplane. These two parameters are tuned by cross-validation and their ratio determines the contribution of the common hyperplane to each task. If λ_1/λ_2 is high, the task-specific hyperplanes are penalized more and all the hyperplanes of the different

tasks tend to be equal to the common hyperplane. By contrast, when λ_1/λ_2 is low, the common hyperplane is penalized more and the models for the different tasks tend to be equal to task-specific hyperplanes.

In the present work, we propose a Bayesian model for multi-task feature selection based on a generalization of the *spike and slab* prior distribution. This generalized prior, detailed in Sect. 2, enforces the selection of a common subset of features to describe each different task. Exact Bayesian inference in this model is infeasible and approximate techniques have to be used in practice. We consider here the expectation propagation (EP) algorithm [9], which is briefly reviewed in Sect. 3. EP approximates the posterior distribution of the model using a parametric distribution that belongs to the exponential family. We detail in Sect. 4 a specific posterior approximation for our multi-task Bayesian model. We also introduce an efficient parametrization that guarantees that EP has a time complexity that can be made linear in the number of features d, under the assumption that d is significantly larger than the number of instances for each task. EP also approximates the posterior probability of using each feature for prediction. These probabilities are particularly useful to identify relevant features. Finally, experiments reported in Sect. 5 are conducted on a collection of multi-task learning problems. They show that our Bayesian model is competitive with respect to baseline approaches of single-task learning and data pooling across all tasks, and with respect to the multi-task learning methods [7, 8] mentioned above. Additional experiments detail the stability of the various feature selection methods under different sub-samplings of the training data.

2 Bayesian Multi-task Feature Selection

Following Obozinski *et al.* [7], we assume that the different tasks share a small subset of common relevant features. This need not be strictly satisfied in practice and a few specific features may be relevant only for some tasks given that the number of these specific features is small. To identify the relevant features we define a Bayesian model relying on a set of linear functions that discriminate between two class labels. A set of $k = 1, \ldots, K$ learning tasks are assumed to be available, each one consisting of n_k d-dimensional input samples $\mathbf{X}_k = \{\mathbf{x}_{k1}, \ldots, \mathbf{x}_{kn_k}\}$ and the corresponding class labels $\mathbf{y}_k = \{y_{k1}, \ldots, y_{kn_k}\}$, where $y_{ki} \in \{-1, 1\}$, $\forall i, k$. We further assume that $n_k \ll d$, $\forall k$. Given $\mathbf{X}_k$ and $\mathbf{y}_k$, let us consider the following labeling rule:

$$y_{ki} = \begin{cases} 1 & \text{if } \mathbf{w}_k^T \mathbf{x}_{ki} + \epsilon_{ki} \geq 0 \\ -1 & \text{otherwise,} \end{cases} \tag{1}$$

where ^{T} denotes vector transpose, $\mathbf{w}_k$ are the model coefficients for task k, and a noise term ϵ_{ki} is assumed to follow a standard Gaussian distribution. The likelihood for $\mathbf{w}_k$ is hence defined as:

$$\mathcal{P}(\mathbf{y}_k | \mathbf{w}_k, \mathbf{X}_k) = \prod_{i=1}^{n_k} \mathcal{P}(y_{ki} | \mathbf{w}_k, \mathbf{x}_{ki}) = \prod_{i=1}^{n_k} \Phi(y_{ki} \mathbf{w}_k^T \mathbf{x}_{ki}) \,, \tag{2}$$

where $\Phi(\cdot)$ denotes the cumulative probability function of a standard Gaussian distribution. For notational convenience, we will remove the explicit conditional dependence on $\mathbf{X}_k$ in the subsequent expressions. We consider a bias term by extending each vector $\mathbf{x}_{ki}$ with a constant component equal to one. Each input sample will also be assumed to have been multiplied by its corresponding label y_{ki} and the result will be simply denoted by $\mathbf{x}_{ki}$.

The prior distribution for the model coefficients $\mathbf{w}_k$ is a generalization of the *spike and slab* sparse prior distribution [5]. In particular, we assume that all components of the vectors $\mathbf{w}_k$ are independent. Binary latent variables γ_j, with $j = 1, \ldots, d+1$ are introduced to indicate whether the j-th feature is used for classification *in each* of the different tasks ($\gamma_j = 1$) or not ($\gamma_j = 0$). Given the vector $\boldsymbol{\gamma}$, the prior for each $\mathbf{w}_k$ is

$$\mathcal{P}(\mathbf{w}_k|\boldsymbol{\gamma}) = \prod_{j=1}^{d+1} \mathcal{N}(w_{kj}|0, \sigma_{1j}^2)^{\gamma_j} \mathcal{N}(w_{kj}|0, \sigma_{0j}^2)^{1-\gamma_j} \,, \tag{3}$$

where $\mathcal{N}(w_{kj}|0, \sigma_{1j}^2)$ denotes a Gaussian density with zero mean and variance equal to σ_{1j}^2. When a single task is considered for learning (3) reduces to the standard spike and slab prior. We set the *spikes* to be deltas centered at the origin, *i.e.* $\sigma_{0j}^2 \to 0$, $\forall j$, to enforce sparsity among the components of each vector $\mathbf{w}_k$. The variances of the *slabs*, σ_{1j}^2, are set equal to one for $j = 1, \ldots, d$. The variance of the *slab* corresponding to the bias term, $\sigma_{1(d+1)}^2$, is set to a significantly larger value (*e.g* 10). Since we further assume that each feature can be used *a priori* independently for classification, the prior on $\boldsymbol{\gamma}$ is simply defined as a multivariate Bernoulli distribution:

$$\mathcal{P}(\boldsymbol{\gamma}) = \prod_{j=1}^{d+1} p_{0j}^{\gamma_j} (1 - p_{0j})^{1-\gamma_j} \,, \tag{4}$$

where p_{0j} specifies the prior probability for $\gamma_j = 1$. The prior probability corresponding to the bias component $p_{0(d+1)}$ is set equal to one in this model.

According to the above definitions, and given $\mathbf{X}_k$ and $\mathbf{y}_k$ for $k = 1, \ldots, K$, we can use the Bayes' theorem to make inference about each $\mathbf{w}_k$ and $\boldsymbol{\gamma}$. Let $\mathbf{W} = \{\mathbf{w}_1, \ldots, \mathbf{w}_K\}$ and $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_K\}$ be two matrices summarizing the model coefficients and the class labels of the different tasks, respectively. The posterior for $\mathbf{W}$ and $\boldsymbol{\gamma}$ is

$$\mathcal{P}(\mathbf{W}, \boldsymbol{\gamma}|\mathbf{Y}) = \frac{\mathcal{P}(\mathbf{Y}|\mathbf{W})\mathcal{P}(\mathbf{W}|\boldsymbol{\gamma})\mathcal{P}(\boldsymbol{\gamma})}{\mathcal{P}(\mathbf{Y})} \,, \tag{5}$$

where $\mathcal{P}(\mathbf{Y}|\mathbf{W}) = \prod_{k=1}^{K} \mathcal{P}(\mathbf{y}_k|\mathbf{w}_k)$, $\mathcal{P}(\mathbf{W}|\boldsymbol{\gamma}) = \prod_{k=1}^{K} \mathcal{P}(\mathbf{w}_k|\boldsymbol{\gamma})$ and $\mathcal{P}(\mathbf{Y})$ is just a normalization constant, known as the model evidence, which can be used to perform model selection under a Bayesian framework [10].

In this model the class label $y_k^{\text{new}} \in \{-1, 1\}$ of a new unlabeled instance $\mathbf{x}_k^{\text{new}}$ corresponding to the task k is computed using the predictive distribution:

$$\mathcal{P}(y_k^{\text{new}}|\mathbf{x}_k^{\text{new}}, \mathbf{Y}) = \int \sum_{\boldsymbol{\gamma}} \mathcal{P}(y_k^{\text{new}}|\mathbf{x}_k^{\text{new}}, \mathbf{w}_k)\mathcal{P}(\mathbf{W}, \boldsymbol{\gamma}|\mathbf{Y})\, d\mathbf{W}\,. \tag{6}$$

This probabilistic output is useful to quantify the uncertainty in the prediction.

Finally, those features with the highest contribution in *all* the classification tasks can be identified using the posterior distribution for $\boldsymbol{\gamma}$:

$$\mathcal{P}(\boldsymbol{\gamma}|\mathbf{Y}) = \int \mathcal{P}(\mathbf{W}, \boldsymbol{\gamma}|\mathbf{Y})\, d\mathbf{W}\,. \tag{7}$$

Unfortunately, the exact computation of (5), (6) and (7) is too expensive for typical learning problems and have to be approximated. We rely here on expectation propagation [9], a fast algorithm for approximate Bayesian inference. This algorithm is described in the next section.

3 Expectation Propagation

In the Bayesian model for multi-task feature selection described in Sect. 2, the joint probability of $\mathbf{W}$, $\boldsymbol{\gamma}$ and $\mathbf{Y}$, *i.e.* the numerator in the right hand side of (5), can be written as a product of several terms t_i

$$\mathcal{P}(\mathbf{W}, \boldsymbol{\gamma}, \mathbf{Y}) = \mathcal{P}(\mathbf{Y}|\mathbf{W})\mathcal{P}(\mathbf{W}|\boldsymbol{\gamma})\mathcal{P}(\boldsymbol{\gamma}) = \prod_i t_i(\mathbf{W}, \boldsymbol{\gamma})\,, \tag{8}$$

where the first $n = n_1 + \ldots + n_K$ terms correspond to $\mathcal{P}(\mathbf{Y}|\mathbf{W})$, the next $K(d+1)$ terms correspond to $\mathcal{P}(\mathbf{W}|\boldsymbol{\gamma})$ and the last term corresponds to $\mathcal{P}(\boldsymbol{\gamma})$. Expectation propagation (EP) approximates each term t_i in (8) by a corresponding simpler term $\tilde{t}_i$. These approximate terms are restricted to have the form of a parametric probability distribution that belongs to the exponential family. They however do not need to integrate to one. Once normalized with respect to $\mathbf{W}$ and $\boldsymbol{\gamma}$, (8) becomes the posterior distribution for the model parameters. Similarly, when normalized with respect to $\mathbf{W}$ and $\boldsymbol{\gamma}$, the product of the approximate terms $\tilde{t}_i$ becomes the posterior approximation:

$$\mathcal{Q}(\mathbf{W}, \boldsymbol{\gamma}) = \frac{1}{Z} \prod_i \tilde{t}_i(\mathbf{W}, \boldsymbol{\gamma}) \approx \mathcal{P}(\mathbf{W}, \boldsymbol{\gamma}|\mathbf{Y})\,, \tag{9}$$

where the normalization constant Z approximates $\mathcal{P}(\mathbf{Y})$, the model evidence. Because of the closure property of the exponential family, $\mathcal{Q}$ has the same parametric form as the approximate terms $\tilde{t}_i$. In practice, the form of $\mathcal{Q}$ is selected first and the approximate terms $\tilde{t}_i$ are constrained by this form. EP iteratively updates each approximate term $\tilde{t}_i$, until the convergence of the posterior approximation $\mathcal{Q}$, in such a way that $\tilde{t}_i \prod_{j \neq i} \tilde{t}_j$ is as close as possible to $t_i \prod_{j \neq i} \tilde{t}_j$. Closeness is defined here in terms of the Kullback-Leibler (KL) divergence. This

procedure guarantees that each approximate term $\tilde{t}_i$ is similar to the corresponding exact term t_i in regions of high posterior probability, as defined by the product of the other approximate terms [9]. The different steps of the EP algorithm are:

1. Initialize all $\tilde{t}_i$ and $\mathcal{Q}$ to be uniform.
2. Repeat until $\mathcal{Q}$ converges:
   (a) Select a $\tilde{t}_i$ to refine and compute $\mathcal{Q}^{\setminus i} \propto \frac{\mathcal{Q}}{\tilde{t}_i}$.
   (b) Update $\tilde{t}_i$ so that $\mathrm{KL}(t_i \mathcal{Q}^{\setminus i} || \tilde{t}_i \mathcal{Q}^{\setminus i})$ is minimized.
   (c) Compute an updated posterior approximation $\mathcal{Q}^{\text{new}} \propto \tilde{t}_i \mathcal{Q}^{\setminus i}$.

Whenever needed for model selection, an approximate model evidence can also be computed by integrating the product of all $\tilde{t}_i$'s.

When $\mathcal{Q}$ is assumed to be Gaussian, the first step of the EP algorithm is implemented by setting the mean and the variance of all the approximate terms and the posterior approximation $\mathcal{Q}$ equal to zero and $+\infty$ respectively. In step 2-(a), $\mathcal{Q}^{\setminus i}$ has the same form as $\mathcal{Q}$ because of the closure property of the exponential family. The optimization problem of step 2-(b) is convex and it can be efficiently solved by matching the sufficient statistics between $t_i \mathcal{Q}^{\setminus i}$ and $\tilde{t}_i \mathcal{Q}^{\setminus i}$ [10]. The EP algorithm is not guaranteed to converge although extensive empirical evidence shows that most of the times it converges to a fixed point solution [9]. Oscillations without convergence can be prevented by using *damped* updates [11]. Finally, the EP algorithm has shown an excellent performance in terms of its computational cost versus its approximation accuracy when compared to other approximate inference methods such as the Laplace approximation, variational inference or Markov chain Monte Carlo sampling [9].

4 The Posterior Approximation

We propose to approximate the posterior (5) by the following parametric distribution belonging to the exponential family:

$$\mathcal{Q}(\mathbf{W}, \boldsymbol{\gamma}) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{w}_k | \mathbf{m}_k, \mathbf{V}_k) \prod_{j=1}^{d+1} p_j^{\gamma_j} (1 - p_j)^{1-\gamma_j} \,, \tag{10}$$

where $\mathcal{N}(\cdot | \mathbf{m}_k, \mathbf{V}_k)$ denotes a multivariate Gaussian distribution with mean vector $\mathbf{m}_k$ and covariance matrix $\mathbf{V}_k$. In (10), $\mathbf{m}_k$, $\mathbf{V}_k$ and $\mathbf{p} = (p_1, \ldots, p_{d+1})^T$ are free parameters that determine the posterior approximation. We denote t_{ki} the exact term of the true posterior associated to the likelihood of the i-th training instance of the k-th learning task:

$$t_{ki}(\mathbf{w}_k) = \Phi(\mathbf{w}_k^T \mathbf{x}_{ki}) \,, \tag{11}$$

and $\tilde{t}_{ki}$ its associated approximate term. The definition of $\mathcal{Q}$ given in (10) constrains the form of $\tilde{t}_{ki}$ to be:

$$\tilde{t}_{ki}(\mathbf{w}_k) = \tilde{s}_{ki} \exp\left\{ -\frac{1}{2\tilde{v}_{ki}} \left(\mathbf{w}_k^T \mathbf{x}_{ki} - \tilde{m}_{ki} \right)^2 \right\} \,, \tag{12}$$

where $\tilde{s}_{ki}$, $\tilde{v}_{ki}$ and $\tilde{m}_{ki}$ are free parameters. Both the exact term t_{ki} and its approximation $\tilde{t}_{ki}$ do not depend on $\boldsymbol{\gamma}$. This is known as the locality property of EP [12]. Additionally, $\tilde{t}_{ki}$ can be written as a univariate Gaussian distribution since each likelihood term only constrains the corresponding vector $\mathbf{w}_k$ through the direction of $\mathbf{x}_{ki}$ [9]. Similarly, we denote t_{kj} the exact term corresponding to the prior for the j-th component of $\mathbf{w}_k$:

$$t_{kj}(\mathbf{w}_k, \boldsymbol{\gamma}) = \mathcal{N}(w_{kj}|0, \sigma_{1j}^2)^{\gamma_j} \mathcal{N}(w_{kj}|0, \sigma_{0j}^2)^{1-\gamma_j} , \tag{13}$$

and $\tilde{t}_{kj}$ its approximation. From (10), it follows that

$$\tilde{t}_{kj}(\mathbf{w}_k, \boldsymbol{\gamma}) = \tilde{s}_{kj} \tilde{p}_{kj}^{\gamma_j} (1 - \tilde{p}_{kj})^{1-\gamma_j} \exp\left\{ -\frac{1}{2\tilde{\nu}_{kj}} (w_{kj} - \tilde{\mu}_{kj})^2 \right\} , \tag{14}$$

where $\tilde{s}_{kj}$, $\tilde{p}_{kj}$, $\tilde{\nu}_{ki}$ and $\tilde{\mu}_{ki}$ are free parameters. Again, $\tilde{t}_{kj}$ only depends on one component of $\mathbf{w}_k$ and $\boldsymbol{\gamma}$ because of the locality property of EP. Finally, the exact term corresponding to the prior for $\boldsymbol{\gamma}$ is given by (4). This term can be estimated exactly, *i.e.* $\tilde{t}(\boldsymbol{\gamma}) = t(\boldsymbol{\gamma}) = \mathcal{P}(\boldsymbol{\gamma})$. Since $\tilde{t}(\boldsymbol{\gamma})$ has no free parameters, it does not require updating.

From the definition of $\mathcal{Q}$ as the normalized product of all the approximate terms, the parameters of the posterior approximation are:

$$\begin{aligned} \mathbf{V}_k &= \left(\mathbf{X}_k \boldsymbol{\Lambda}_k \mathbf{X}_k^T + \boldsymbol{\Delta}_k\right)^{-1} , & \mathbf{m}_k &= \mathbf{V}_k \left(\mathbf{X}_k \boldsymbol{\eta}_k + \boldsymbol{\Delta}_k \tilde{\boldsymbol{\mu}}_k\right) , \\ p_j &= \frac{\prod_{k=1}^K \tilde{p}_{kj} p_{0j}}{\prod_{k=1}^K \tilde{p}_{kj} p_{0j} + \prod_{k=1}^K (1 - \tilde{p}_{kj})(1 - p_{0j})} , & & \text{for } j = 1, \ldots, d+1 , \end{aligned} \tag{15}$$

where we have defined $\boldsymbol{\Lambda}_k = \mathrm{diag}(\tilde{v}_{k1}^{-1}, \ldots, \tilde{v}_{kn_k}^{-1})$, $\boldsymbol{\Delta}_k = \mathrm{diag}(\tilde{\nu}_{k1}^{-1}, \ldots, \tilde{\nu}_{k(d+1)}^{-1})$, $\boldsymbol{\eta}_k = (\tilde{m}_{k1}/\tilde{v}_{k1}, \ldots, \tilde{m}_{kn_k}/\tilde{v}_{kn_k})^T$, $\tilde{\boldsymbol{\mu}}_k = (\tilde{\mu}_{k1}, \ldots, \tilde{\mu}_{k(d+1)})^T$ and $\mathrm{diag}(\cdot)$ denotes a diagonal matrix.

4.1 Efficient EP Update Scheme

The EP algorithm iteratively updates the posterior approximation $\mathcal{Q}$, which includes the computation of K covariance matrices of size $(d+1) \times (d+1)$. Thus, a straightforward implementation would have a time complexity in $\mathcal{O}(Kd^2)$ for each EP iteration. Since, in our context $d \gg n_k \ \forall k$, we introduce an alternative parametrization of the posterior approximation which provides a more efficient updating scheme. This parametrization is similar to the one that arises in kernel classifiers when the EP algorithm is written in terms of inner products [9]. Specifically, instead of explicitly storing the parameters $\mathbf{m}_k$ and $\mathbf{V}_k$ of $\mathcal{Q}$, we define and store

$$\mathbf{A}_k = \mathbf{X}_k^T \mathbf{V}_k \mathbf{X}_k , \qquad \mathbf{B}_k = \mathbf{X}_k^T \mathbf{V}_k \boldsymbol{\Delta}_k , \qquad \mathbf{h}_k = \mathbf{X}_k^T \mathbf{m}_k , \tag{16}$$

where $\mathbf{A}_k \in \mathbb{R}^{n_k, n_k}$, $\mathbf{B}_k \in \mathbb{R}^{n_k, d}$, and $\mathbf{h} \in \mathbb{R}^{n_k}$. These new parameters are updated after each EP iteration.

The first step of the EP algorithm would typically set the approximate terms to be uniform. Some iterations of the EP algorithm may however be saved if the approximate terms are initialized in such a way that the Gaussian part of $\mathcal{Q}$ has the same mean and variance as the exact prior for $\mathbf{W}$:

$$\begin{aligned} &\tilde{v}_{ki} = +\infty\,, \qquad \tilde{m}_{ki} = 0\,, \quad h_{ki} = 0\,, \qquad p_j = p_{0j}\,, \\ &\tilde{\nu}_{kj} = \sigma_{1j}^2 p_{0j}\,, \quad \tilde{\mu}_{kj} = 0\,, \quad \mathbf{B}_k = \mathbf{X}_k^T\,, \quad \mathbf{A}_k = \mathbf{X}_k^T \mathbf{D} \mathbf{X}_k\,, \quad \forall k, i, j\,, \end{aligned} \tag{17}$$

where $\mathbf{D} = \text{diag}\left(\sigma_{11}^2 p_{01}, \ldots, \sigma_{1(d+1)}^2 p_{0(d+1)}\right)$ and $\mathcal{Q}$ has been initialized to the normalized product of all approximate terms. The constants $\tilde{s}_{ki}$ and $\tilde{s}_{kj}$ of the approximate terms can be set to any arbitrary positive value.

Let $\mathcal{Q}^{\backslash ki}$ denote the posterior approximation that results from removing from $\mathcal{Q}$ the approximate likelihood term $\tilde{t}_{ki}$. Furthermore, let $\mathbf{m}_k^{\backslash ki}$ and $\mathbf{V}_k^{\backslash ki}$ denote the parameters of the resulting Gaussian approximation to the posterior of $\mathbf{w}_k$. If we define

$$h_{ki}^{\backslash ki} = \mathbf{x}_{ki}^T \mathbf{m}_k^{\backslash ki}\,, \qquad\qquad \lambda_{ki}^{\backslash ki} = \mathbf{x}_{ki}^T \mathbf{V}_k^{\backslash ki} \mathbf{x}_{ki}\,, \tag{18}$$

the step 2-(a) of the EP algorithm for these approximate terms becomes:

$$h_{ki}^{\backslash ki} = h_{ki} + \frac{(\mathbf{A}_k)_{ii}}{\tilde{v}_{ki} - (\mathbf{A}_k)_{ii}} \left(h_{ki} - \tilde{m}_{ki}\right)\,, \quad \lambda_{ki}^{\backslash ki} = (\mathbf{A}_k)_{ii} + \frac{(\mathbf{A}_k)_{ii}^2}{\tilde{v}_{ki} - (\mathbf{A}_k)_{ii}}\,. \tag{19}$$

Part of an updated posterior distribution $\mathcal{Q}^{\text{new}}$ may be computed:

$$h_{ki}^{\text{new}} = \mathbf{x}_{ki}^T \mathbf{m}_k^{\text{new}} = h_{ki}^{\backslash ki} + \alpha_{ki} \lambda_{ki}^{\backslash ki}\,, \tag{20}$$

where

$$\alpha_{ki} = \frac{\mathcal{N}(u_{ki}|0,1)}{Z_{ki}} \frac{1}{\sqrt{\lambda_{ki}^{\backslash ki} + 1}}\,, \qquad u_{ki} = \frac{h_{ki}^{\backslash ki}}{\sqrt{\lambda_{ki}^{\backslash ki} + 1}}\,, \qquad Z_{ki} = \Phi(u_{ki})\,. \tag{21}$$

The updated approximate term $\tilde{t}_{ki}$ follows from

$$\begin{aligned} &\tilde{v}_{ki} = \lambda_{ki}^{\backslash ki} \left(\frac{1}{\alpha_{ki}(h_{ki}^{\text{new}} + \alpha_{ki})} - 1 \right)\,, \qquad\qquad \tilde{m}_{ki} = h_{ki}^{\text{new}} + \alpha_{ki} \tilde{v}_{ki}\,, \\ &\tilde{s}_{ki} = Z_{ki} \sqrt{1 + \lambda_{ki}^{\backslash ki} / \tilde{v}_{ki}} \exp\left\{ \frac{\alpha_{ki}}{2} \frac{\lambda_{ki}^{\backslash ki} + 1}{h_{ki}^{\text{new}} + \alpha_{ki}} \right\}\,. \end{aligned} \tag{22}$$

Finally, $\mathcal{Q}^{\text{new}}$ results from updating the matrices $\mathbf{A}_k$ and $\mathbf{B}_k$ and the vector $\mathbf{h}_k$ using the Woodbury formula:

$$\begin{aligned} &\mathbf{A}_k^{\text{new}} = \mathbf{A}_k - \frac{(\mathbf{A}_k)_{\cdot,i} (\mathbf{A}_k)_{i,\cdot}}{c_k^{-1} + (\mathbf{A}_k)_{ii}}\,, \qquad\qquad \mathbf{B}_k^{\text{new}} = \mathbf{B}_k - \frac{(\mathbf{A}_k)_{\cdot,i} (\mathbf{B}_k)_{i,\cdot}}{c_k^{-1} + (\mathbf{A}_k)_{ii}}\,, \\ &\mathbf{h}_k^{\text{new}} = \mathbf{A}_k^{\text{new}} \boldsymbol{\eta}_k + \mathbf{B}_k^{\text{new}} \tilde{\boldsymbol{\mu}}_k\,, \end{aligned} \tag{23}$$

where $c_k = 1/\tilde{v}_{ki}^{\text{new}} - 1/\tilde{v}_{ki}^{\text{old}}$. The computational complexity of these updates is in $\mathcal{O}(n_k d)$.

The approximate terms $\tilde{t}_{kj}$ corresponding to the prior for $\mathbf{W}$ are updated in parallel for each task k, as in [13]. For all $j = 1, \ldots, d+1$, $\tilde{t}_{kj}$ is removed from $\mathcal{Q}$ and the posterior approximation $\mathcal{Q}^{\backslash kj}$ is computed . Given each $\mathcal{Q}^{\backslash kj}$, the corresponding approximate terms are updated simultaneously. The updates for such a parallel scheme are simpler as they only require the marginals of the approximate posterior for $\mathbf{w}_k$. Once the parallel updates have been performed, the posterior approximation for $\mathbf{w}_k$ needs to be recomputed as the normalized product of the approximate terms.

Given $\mathcal{Q}$, the cost of computing the marginals of the posterior approximation for $\mathbf{w}_k$ is in $\mathcal{O}(n_k^2 d)$ when $d \gg n_k$. Let $\mathbf{v}_k = (v_{k1}, \ldots, v_{k(d+1)})^T$ be a vector summarizing the variance of each marginal and $\mathbf{m}_k$ a vector summarizing the corresponding means:

$$\mathbf{v}_k = \tilde{\boldsymbol{\nu}}_k - \tilde{\boldsymbol{\nu}}_k \circ ((\mathbf{X}_k \mathbf{L}_k \circ \mathbf{X}_k)\,\mathbf{1}) \circ \tilde{\boldsymbol{\nu}}_k\,, \quad \mathbf{m}_k = \boldsymbol{\zeta} - \tilde{\boldsymbol{\nu}}_k \circ \left(\mathbf{X}_k \mathbf{L}_k \mathbf{X}_k^T \boldsymbol{\zeta}\right)\,, \tag{24}$$

where $\circ$ indicates the Hadamard element-wise product, $\tilde{\boldsymbol{\nu}}_k = (\tilde{\nu}_{k1}, \ldots, \tilde{\nu}_{k(d+1)})^T$, $\mathbf{1}$ is a vector of ones $\mathbf{L}_k = \boldsymbol{\Lambda}_k - \boldsymbol{\Lambda}_k \mathbf{A}_k \boldsymbol{\Lambda}_k$ and $\boldsymbol{\zeta} = \tilde{\boldsymbol{\nu}}_k \circ (\mathbf{X}_k \boldsymbol{\eta}_k + \boldsymbol{\Delta}_k \tilde{\boldsymbol{\mu}}_k)$. $\mathcal{Q}^{\backslash kj}$ is computed simultaneously for each approximate term $\tilde{t}_{kj}$ from (24). Let $v_{kj}^{\backslash kj}$ and $m_{kj}^{\backslash kj}$ denote the variance and the mean of the posterior distribution of w_{kj} under each $\mathcal{Q}^{\backslash kj}$. Let $p_j^{\backslash kj}$ denote the parameter that determines the posterior probability of $\gamma_j = 1$ in $\mathcal{Q}^{\backslash kj}$. For each $\mathcal{Q}^{\backslash kj}$, these parameters are defined as:

$$\begin{aligned} p_j^{\backslash kj} &= \frac{p_j/\tilde{p}_{kj}}{p_j/\tilde{p}_{kj} + (1-p_j)/(1-\tilde{p}_{kj})}\,, \quad \forall j\,, \quad v_{kj}^{\backslash kj} = \left(v_{kj}^{-1} - \tilde{\nu}_{kj}^{-1}\right)^{-1}\,, \quad \forall j\,, \\ m_{kj}^{\backslash kj} &= v_{kj}^{\backslash kj}\left(v_{kj}^{-1} m_{kj} - \tilde{\nu}_{kj}^{-1}\tilde{\mu}_{kj}\right)^{-1}\,, \quad \forall j\,. \end{aligned} \tag{25}$$

The corresponding approximate terms $\tilde{t}_{kj}$ are

$$\begin{aligned} \tilde{\nu}_{kj} &= c_3^{-1} - v_{ki}^{\backslash ki}\,, \quad \forall j\,, \quad \tilde{s}_{kj} = (\mathcal{G}_1 + \mathcal{G}_0)\sqrt{1 + v_{ki}^{\backslash ki}/\tilde{\nu}_{kj}}\exp\left\{\frac{1}{2}\frac{c_1^2}{c_3}\right\}\,, \quad \forall j\,, \\ \tilde{p}_{kj} &= \frac{\mathcal{G}_1}{\mathcal{G}_1 + \mathcal{G}_0}\,, \quad \forall j\,, \qquad \tilde{\mu}_{kj} = m_{kj}^{\backslash kj} - c_1\left(\tilde{\nu}_{kj} + v_{ki}^{\backslash ki}\right)\,, \quad \forall j\,, \end{aligned} \tag{26}$$

where

$$\begin{aligned} c_1 &= \rho_{kj} a_1 + (1-\rho_{kj}) a_0\,, & c_2 &= \rho_{kj}\left[a_1^2 - \frac{a_1}{m_{kj}^{\backslash kj}}\right] + (1-\rho_{kj})\left[a_0^2 - \frac{a_0}{m_{kj}^{\backslash kj}}\right]\,, \\ c_3 &= c_1^2 - c_2\,, & Z_{kj} &= p_j^{\backslash kj}\mathcal{G}_1 + (1-p_j^{\backslash kj})\mathcal{G}_0\,, \\ \mathcal{G}_0 &= \mathcal{N}(0|m_{kj}^{\backslash kj}, v_{kj}^{\backslash kj} + \sigma_{0j}^2)\,, & \mathcal{G}_1 &= \mathcal{N}(0|m_{kj}^{\backslash kj}, v_{kj}^{\backslash kj} + \sigma_{1j}^2)\,, \\ a_0 &= m_{kj}^{\backslash kj}/\left(v_{kj}^{\backslash kj} + \sigma_{0j}^2\right)\,, & a_1 &= m_{kj}^{\backslash kj}/\left(v_{kj}^{\backslash kj} + \sigma_{1j}^2\right)\,, \\ \rho_{kj} &= p_j^{\backslash kj}\mathcal{G}_1/Z_{kj}\,. \end{aligned} \tag{27}$$

The parallel updates for the corresponding posterior distribution $\mathcal{Q}^{\text{new}}$ are

$$m_{kj}^{\text{new}} = m_{kj}^{\backslash kj} - c_1 v_{kj}^{\backslash kj}\,, \quad \forall j\,, \qquad v_{kj}^{\text{new}} = v_{kj}^{\backslash kj}\left(1 - c_3 v_{kj}^{\backslash kj}\right)\,, \quad \forall j\,,$$
$$p_{kj}^{\text{new}} = p_j^{\backslash kj} + \frac{\mathcal{G}_1 - \mathcal{G}_0}{Z_{kj}} p_j^{\backslash kj}(1 - p_j^{\backslash kj})\,, \quad \forall j\,. \tag{28}$$

Finally, $\mathcal{Q}^{\text{new}}$ results from updating the matrices $\mathbf{A}_k$ and $\mathbf{B}_k$:

$$\mathbf{A}_k^{\text{new}} = \mathbf{M}_k - \mathbf{M}_k\left(\mathbf{M}_k + \boldsymbol{\Lambda}_k^{-1}\right)^{-1}\mathbf{M}_k\,, \tag{29}$$
$$\mathbf{B}_k^{\text{new}} = \mathbf{X}_k^T - \mathbf{M}_k\left(\mathbf{M}_k + \boldsymbol{\Lambda}_k^{-1}\right)^{-1}\mathbf{X}_k^T\,, \tag{30}$$

where $\mathbf{M}_k = \mathbf{X}_k^T \boldsymbol{\Delta}_k^{-1}\mathbf{X}_k \in \mathbb{R}^{n_k, n_k}$. The computational complexity of these updates is in $\mathcal{O}(n_k^2 d)$ since $d \gg n_k$. The vector $\mathbf{h}_k$ is updated as in (23).

4.2 Predictive Distribution and Feature Selection

The predictive distribution (6) can be approximated using $\mathcal{Q}$ as an estimate of the exact posterior:

$$\mathcal{P}(y_k^{\text{new}}|\mathbf{x}_k^{\text{new}}, \mathbf{Y}) \approx \Phi\left(\frac{\mathbf{m}_k^T \mathbf{x}_k^{\text{new}}}{\sqrt{(\mathbf{x}_k^{\text{new}})^T \mathbf{V}_k \mathbf{x}_k^{\text{new}} + 1}}\right)\,, \tag{31}$$

where $\mathbf{m}_k$ can be obtained as in (24) and

$$(\mathbf{x}_k^{\text{new}})^T \mathbf{V}_k \mathbf{x}_k^{\text{new}} = (\mathbf{x}_k^{\text{new}})^T \boldsymbol{\Delta}_k^{-1}\mathbf{x}_k^{\text{new}} - (\mathbf{x}_k^{\text{new}})^T \boldsymbol{\Delta}_k^{-1}\mathbf{X}_k \mathbf{L}_k \mathbf{X}_k^T \boldsymbol{\Delta}_k^{-1}\mathbf{x}_k^{\text{new}}\,.$$

Since $\mathbf{m}_k$ is already computed once the model is estimated, the cost of evaluating (31) is in $\mathcal{O}(n_k d)$.

The EP approximation of (7) is used to identify the most relevant features:

$$\mathcal{P}(\boldsymbol{\gamma}|\mathbf{Y}) \approx \prod_{j=1}^{d+1} p_j^{\gamma_j}(1 - p_j)^{1-\gamma_j}\,, \tag{32}$$

where p_j estimates the posterior probability of using attribute j for prediction in *all* the tasks.

Assuming a constant number of EP iterations until convergence, a reasonable assumption in practice, the time complexity of the EP algorithm is in $\mathcal{O}(\sum_{k=1}^K n_k^2 d)$. This complexity, linear in d, is good since $d \gg n_k, \forall k$. Finally, our actual EP implementation also relies on *damped* updates [11], as they seem to improve the overall convergence.

5 Experiments

We detail here several multi-task experiments to assess the performance of the Bayesian multi-task feature selection (BMFS) introduced in Sect. 2. Comparative

results are reported with single-task learning (one classifier estimated independently on each task) or data pooling across all tasks (one global classifier). In both cases, those individual classifiers are estimated through the same EP procedure as in BMFS but with the original single-task spike and slab prior [5]. We also present the performances of linear models regularized with a mixed norm [7] (the ℓ_1/ℓ_2 method) and the regularized multi-task learning (RMLT) method [8][3].

5.1 Arabic Digits

A first batch of experiments is carried out using the dataset of Arabic handwritten digits MADBase [14]. This dataset contains $70,000$ images of size 28×28 pixels of each Arabic digit written by 700 different writers. There are 10 images of each digit (0 to 9) written by the same writer. Fig. 1 displays some examples of the images contained in this dataset. We consider binary classification tasks to discriminate each digit from the digit 0 for a particular writer, which is arguably a difficult problem [14]. For each digit i versus digit 0 with $i \neq 0$, we extract the 800 available images corresponding to 40 different writers (writers 601 to 640). We thus consider 40 tasks (one per writer) with 20 samples per task. The data for each task is randomly split 100 times into training and test sets, with 17 and 3 instances respectively, and the average prediction accuracy is reported.

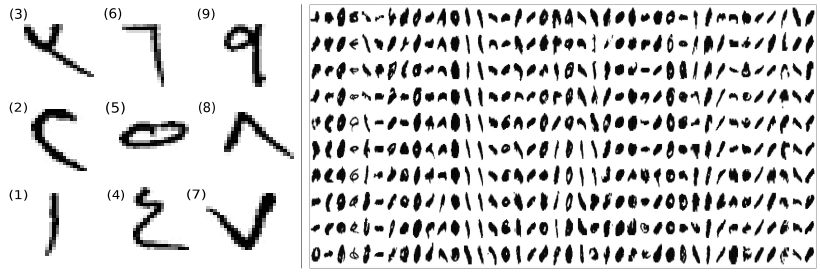

Fig. 1. Arabic digits from 1 to 9 (left). The Arabic digit 0 written ten times by forty different persons, with a strongly writer-dependent style (right).

In BMFS, single-task learning and pooling we set $p_{0j} = 5\%$ for $j = 1, \ldots, d$ to model our prior belief that an accurate classification may only depend on a few pixels (approximately 40). In the ℓ_1/ℓ_2 method, λ is chosen using the algorithm described in [7] by minimizing the cross-validation error[4]. Similarly,

[3] As for BMFS, we consider a bias term in those multi-task methods by extending each input sample with a constant component equal to one. Additionally, in the ℓ_1/ℓ_2 method it is straightforward to not regularize the corresponding coefficient.

[4] The parameters of this algorithm are $\epsilon = 0.02$, $\lambda_{\max} = \lambda_0/500$ and $\xi = \min(10^{-3}, 0.01\lambda)$, as suggested in [7].

Table 1. Prediction error in % (and standard deviation) of each method on each binary problem.

Problem	BMFS	ℓ_1/ℓ_2	RMTL	Single-Task	Pooling
0 vs 1	**0.5±0.4**	1.3±0.7	4.0±2.1	3.2±1.2	2.1±1.1
0 vs 2	**0.6±0.4**	1.0±0.7	4.7±1.5	6.6±1.7	3.5±2.1
0 vs 3	**1.6±0.7**	2.3±0.8	4.5±1.8	5.0±1.4	4.5±1.8
0 vs 4	**1.3±0.7**	**1.3±0.7**	4.9±1.7	5.0±1.4	2.1±0.9
0 vs 5	**1.0±0.6**	1.6±0.8	7.3±2.2	9.2±1.9	8.0±3.4
0 vs 6	**0.4±0.4**	1.1±0.6	3.5±1.8	3.2±1.2	3.0±6.1
0 vs 7	**0.5±0.4**	1.2±0.7	3.5±1.4	6.1±1.6	3.5±1.6
0 vs 8	**1.4±0.7**	2.2±1.0	4.2±2.2	6.0±1.7	4.5±1.9
0 vs 9	**1.2±0.7**	1.4±0.7	5.0±1.5	3.8±1.3	3.5±1.4
Average	**1.0±0.2**	1.5±0.2	4.6±0.5	5.3±0.4	3.8±0.9

in the RMTL method λ_1 and λ_2 are chosen with a grid search over ten values from 0.01 to 100 by cross-validation. The data is also normalized to have zero mean and unit standard deviation on the training set.

Table 1 displays the prediction error of each method for each binary problem of the form digit i vs digit 0 with $i \neq 0$ averaged over the 40 different tasks. The error of the best method for each binary problem is high-lighted in bold face. BMFS obtains the best prediction error in all the problems investigated, outperforming the ℓ_1/ℓ_2 and the RMTL methods, except for the problem 0 vs 4, where the ℓ_1/ℓ_2 method obtains similar results. The Bayesian approach also outperforms single-task learning or data pooling in all cases. A Friedman rank test (p-value $= 6.1 \cdot 10^{-81}$) and a Nemenyi post-hoc test (p-value $= 7.6 \cdot 10^{-5}$) confirm that there is statistical evidence supporting a performance difference in favor of BMFS when the average prediction error across the nine problems is considered.

Fig. 2 shows the estimate of the relative importance of each feature (pixel) as computed by EP in (32) for BMFS, single-task learning (we display the results for the first writer) and pooling. The figure displays the values of p_j for $j = 1, \ldots, d$ using a gray scale from white ($p_j = 0$) to black ($p_j = 1$). These results correspond to a fixed train / test partition of the binary classification problem 0 vs 8. The gray background color indicates a value for p_j equal to 5%, *i.e.* the prior value for $\gamma_j = 1$. The figure shows that BMFS allows us to be the most confident about the relative importance of each feature. With single-task learning all pixels are very close to the prior value, which is likely related to the reduced number of training instances for each task. Pooling improves over single-task learning but is still much less informative than BMFS. The smaller confidence obtained with pooling is likely related to the estimation of a single hyperplane to describe all the learning tasks. Similar observations can be made from the different binary problems considered and the different train/test partitions.

Fig. 2. Feature importance in gray scale as computed by EP for the Bayesian multi-task approach (left), single-task learning (middle) and pooling (right). The multi-task approach is the most confident method about the feature importance.

Table 2. Characteristics of the three prostate cancer microarray datasets.

Name	Normal	Tumor	Features	Platform	Ref.
Singh	50	52	12,626	HGU95Av2	[16]
Stuart	50	38	12,626	HGU95Av2	[17]
Welsh	9	25	12,625	HGU95A	[18]

5.2 Microarray Data

A second batch of experiments is carried out using three microarray prostate cancer datasets described in Table 2, where each dataset is identified by the first author of the corresponding reference. The Welsh dataset is made of gene expression values. The Singh and Stuart original data is made of laser intensity images from each microarray. The RMA preprocesing method [15] is used to produce gene expression values from these images. The microarray technology is common for Singh and Stuart but is slightly older for Welsh. We restrict our attention to the $12,600$ features they have in common. For each dataset there is one learning task to discriminate between normal and tumor samples. We randomly split 100 times the data for each task into $2/3$ (training) and $1/3$ (test). In BMFS, we set $p_{0j} = 50/d$, for $j = 1, \ldots, d$, to model our prior belief that only a few genes (50) may be relevant for classification . The data and the hyper-parameters of the other multi-task methods are respectively normalized and set as described in Sect. 5.1.

Table 3 reports balanced classification rates (BCR) for each method, averaged over data partitions. This evaluation metric is the arithmetic mean of the accuracy within each class (tumor, normal). It is preferred over standard accuracy to assess prediction performance from highly unbalanced classes [19]. It is also the arithmetic mean between specificity and sensitivity, commonly used in the medical domain. The table shows that BMFS obtains the best performance for Welsh, the ℓ_1/ℓ_2 model is optimal for Singh and the RMTL method is op-

Table 3. BCR in % (and standard deviation) of each method on each prostate cancer classification task.

Task	BMFS	ℓ_1/ℓ_2	RMTL	Single-Task	Pooling
Singh	90.3±4.5	**90.4±4.6**	89.6±5.1	89.4±4.5	89.4±5.0
Stuart	78.6±6.3	76.9±5.5	**80.0±5.7**	77.7±6.3	79.8±6.0
Welsh	**97.3±6.0**	94.0±10.9	93.4±6.8	95.6±8.4	93.4±6.8
Average	**88.7±2.9**	87.0±3.9	87.6±3.3	87.5±3.2	87.5±3.3

timal for Stuart. BMFS is overall the best method according to average results over the three tasks. A Friedman rank test (p-value $= 8.4 \cdot 10^{-5}$) and a Nemenyi post-hoc test (p-value $= 3.5 \cdot 10^{-3}$) confirm that these differences are statistically significant.

Finally, we compare the different methods in terms of their stability for identifying relevant features when the training data is slightly modified. The Kuncheva stability index [20] measures to which extent T sets of m selected features share common elements. Let $\mathcal{S}_i^m$ denote the set of the top m features identified by a method from the i-th train/test partition. The Kuncheva index over the several data partitions $\mathcal{A}_m = \{\mathcal{S}_i^m : i = 1, \ldots, T\}$ is defined as

$$\mathcal{I}(\mathcal{A}_m) = \frac{2}{T(T-1)} \sum_{i=1}^{T-1} \sum_{j=i+1}^{T} \frac{|\mathcal{S}_i^m \cap \mathcal{S}_j^m| - \frac{m^2}{d}}{m - \frac{m^2}{d}}, \tag{33}$$

where $T = 100$ is the number of training sets, d is the total number of features and m^2/d is the expected value of $|\mathcal{S}_i^m \cap \mathcal{S}_j^m|$ by chance. The index satisfies $-1 < \mathcal{I}(\mathcal{A}_m) \leq 1$ and the closer to one, the larger the number of common features in the different sets. A value of the index near zero indicates commonly selected features at a chance level.

Fig. 3 displays the stability of each method as a function of m. For BMFS, pooling and single-task learning, features are ranked according to the vector $\mathbf{p}$. For the ℓ_1/ℓ_2 method, features are ranked according to the order in which they are included in the active set of the path-following algorithm of [7]. For the RMTL approach features are ranked according to the sum of the corresponding squared coefficients of each hyperplane, *i.e.* $\sum_{k=1}^{K} w_{kj}^2$ for feature j. This value is used as an estimate of the relative feature importance. For the single-task method, we display the value of the stability index averaged over the three learning tasks. The figure shows that the ℓ_1/ℓ_2 method offers the most stable selection for small subsets of features, followed by BMFS, whose selection is more stable than the ones of single-task learning and data pooling.

6 Conclusion

We propose a novel Bayesian model for multi-task feature selection. This model is based on a generalized *spike and slab* sparse prior distribution that enforces

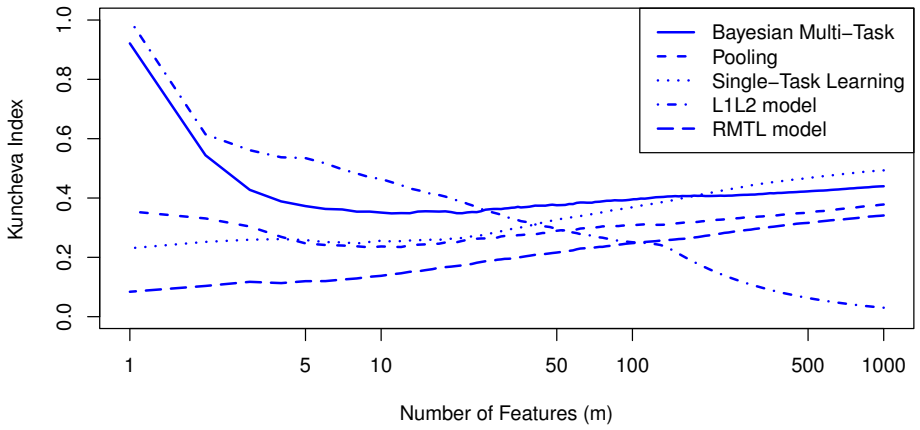

Fig. 3. Stability (Kuncheva index) of the feature ranking implemented by each method.

the selection of a common subset of features to describe each task. Since exact Bayesian inference is intractable for this model, approximate inference is performed through expectation propagation (EP). We propose an original parametrization of the EP procedure which offers a linear complexity in the number of features. Practical experiments on multi-task digit recognition and microarray data classification illustrate the benefits of the proposed approach, as compared to simple baselines and state-of-the-art multi-task approaches, in terms of predictive performance and stability of the feature selection.

Acknowledgment

T. Helleputte is funded by a FRIA grant (1.E091.07). Computations were run on the INGRID cluster of the Center for Intensive Computation and Mass Storage (Louvain). J. M. Hernández-Lobato is funded by the Spanish MCyT under project TIN2007-66862-C02.

References

1. Dudoit, S., Fridlyand, J.: Classification in microarray experiments. In: Statistical Analysis of Gene Expression Microarray Data. Chapman and Hall, CRC Press (2003) 93–158
2. Seeger, M., Nickisch, H., Pohmann, R., Schölkopf, B.: Optimization of k-space trajectories for compressed sensing by Bayesian experimental design. Magnetic Resonance in Medicine **63**(1) (2009) 116–126
3. Johnstone, I., Titterington, D.: Statistical challenges of high-dimensional data. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **367**(1906) (2009) 4237

4. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological) **58**(1) (1996) 267–288
5. George, E.I., McCulloch, R.E.: Approaches for Bayesian variable selection. Statistica Sinica **7**(2) (1997) 339–373
6. Ishwaran, H., Rao, J.: Spike and slab variable selection: frequentist and Bayesian strategies. The Annals of Statistics **33**(2) (2005) 730–773
7. Obozinski, G., Taskar, B., Jordan, M.: Joint covariate selection and joint subspace selection for multiple classification problems. Statistics and Computing (2009) 1–22
8. Evgeniou, T., Pontil, M.: Regularized multi–task learning. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2004) 109–117
9. Minka, T.: A Family of Algorithms for approximate Bayesian Inference. PhD thesis, Massachusetts Institute of Technology (2001)
10. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer (August 2006)
11. Minka, T., Lafferty, J.: Expectation-propagation for the generative aspect model. In Darwiche, A., Friedman, N., eds.: Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann (2002) 352–359
12. Seeger, M.: Notes on Minka's expectation propagation for Gaussian process classification. Technical report, University of Edinburgh (2002)
13. Gerven, M.V., Cseke, B., Oostenveld, R., Heskes, T.: Bayesian source localization with the multivariate Laplace prior. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A., eds.: Advances in Neural Information Processing Systems 22. (2009) 1901–1909
14. Abdleazeem, S., El-Sherif, E.: Arabic handwritten digit recognition. International Journal on Document Analysis and Recognition **11**(3) (2008) 127–141
15. Irizarry, R., Hobbs, B., Collin, F., Beazer-Barclay, Y., Antonellis, K., Scherf, U., Speed, T.: Exploration, normalization, and summaries of high density oligonucleotide array probe level data. Biostatistics **4**(2) (2003) 249
16. Singh, D., Febbo, P.G., Ross, K., Jackson, D.G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A.A., D'Amico, A.V., Richie, J.P., Lander, E.S., Loda, M., Kantoff, P.W., Golub, T.R., Sellers, W.R.: Gene expression correlates of clinical prostate cancer behavior. Cancer Cell **1** (2002) 203–209
17. Stuart, R., Wachsman, W., Berry, C., Wang-Rodriguez, J., Wasserman, L., Klacansky, I., Masys, D., Arden, K., Goodison, S., McClelland, M., et al.: In silico dissection of cell-type-associated patterns of gene expression in prostate cancer. Proceedings of the National Academy of Sciences **101**(2) (2004) 615
18. Welsh, J., Sapinoso, L., Su, A., Kern, S., Wang-Rodriguez, J., Moskaluk, C., Frierson Jr, H., Hampton, G.: Analysis of gene expression identifies candidate markers and pharmacological targets in prostate cancer. Cancer Research **61**(16) (2001) 5974
19. Helleputte, T., Dupont, P.: Feature selection by transfer learning with linear regularized models. In Buntine, W.L., Grobelnik, M., Mladenic, D., Shawe-Taylor, J., eds.: Proceedings of the 19th European Conference on Machine Learning. Volume 5781 of Lecture Notes in Computer Science., Springer (2009) 533–547
20. Kuncheva, L.I.: A stability index for feature selection. In: Proceedings of the 25th IASTED International Multi-Conference on Artificial Intelligence and Applications, Anaheim, CA, USA, ACTA Press (2007) 390–395