Relevant subtask learning by constrained mixture models

Jaakko Peltonen[1,*] Yusuf Yaslan[2] Samuel Kaski[1]

[1] Helsinki University of Technology,
Department of Information and Computer Science,
P. O. Box 5400, FI-02015 TKK, Finland

[2] Istanbul Technical University,
Department of Computer Engineering,
34469 Maslak Istanbul, Turkey

Abstract

We introduce relevant subtask learning, a new learning problem which is a variant of multi-task learning. The goal is to build a classifier for a task-of-interest for which we have too few training samples. We additionally have "supplementary data" collected from other tasks, but it is uncertain which of these other samples are relevant, that is, which samples are classified in the same way as in the task-of-interest. The research problem is how to use the "supplementary data" from the other tasks to improve the classifier in the task-of-interest. We show how to solve the problem, and demonstrate the solution with logistic regression classifiers.

*Corresponding author. Tel. +358-9-47024429, Fax. +358-9-47023277, email: jaakko.peltonen@tkk.fi

The key idea is to model all tasks as mixtures of relevant and irrelevant samples, and model the irrelevant part with a sufficiently flexible model such that it does not distort the model of relevant data. We give two learning algorithms for the method—a simple maximum likelihood optimization algorithm and a more advanced variational Bayes inference algorithm; in both cases we show that the method works better than a comparable multi-task learning model and naive methods.

Keywords: Multi-task learning, relevant subtask learning

1 Introduction

It is all too common in classification tasks that there is too little training data to estimate sufficiently powerful models. This problem is ubiquitous in bioinformatics; there it is called the "large p, small n problem" where p is the dimensionality and n is the number of data. The problem appears also in image classification from few examples, finding of relevant texts, etc. Possible solutions could be to restrict the classifier complexity by prior knowledge, or to gather more data. However, prior knowledge may be insufficient or may not exist, measuring new data may be too expensive, and there may not exist more samples of *representative* data. Most classifiers assume that learning data are representative, that is, they come from the same distribution as test data.

Often, *partially representative* data is available; for example, in bioinformatics there are databases full of data measured for different tasks, conditions or contexts; for texts there is the web. They can be seen as training data from a (partly) different distribution as the test data. Assuming we have several sets, each potentially having some portion of relevant data, our research problem is, *can we use the partially relevant data sets to build a better classifier for the test data?* We discuss this learning problem and present a probabilistic mixture modeling solution for it; this paper extends our earlier conference

publication [17].

This learning problem is a special *multi-task learning* problem. In multi-task learning [13], where learning a classifier for one data set is called a task, models have mainly been symmetrical, and transfer to new tasks is done by using the posterior from other tasks as a prior (e.g. [22, 27]). By contrast, our problem is fundamentally asymmetric and more structured: test data fits one task, the "*task-of-interest*," and the other tasks may contain *subtasks* relevant for the task-of-interest, but no other task needs to be wholly relevant. Our models focus on the task-of-interest and should be better than standard models in it, yet have the same order of complexity as earlier multi-task models.

Previous work. The problem is partly related to several other learning problems: transfer learning, multi-task learning, and semisupervised learning.

A common multi-task learning approach is to build a hierarchical (Bayesian) model of all tasks, with constrained priors favoring similar parameters across tasks (see, e.g., [34] for a description of several typical assumptions), or formulate the same principle through non-probabilistic regularization. Tasks may be learned together [2, 4, 7, 10, 19, 24, 28, 33] or a new task can use a prior learned from previous tasks [22, 27]. Both approaches model all tasks symmetrically. Support vector machines (SVMs) have been used in symmetric hierarchical modeling as well (e.g. [15]). In contrast, we study an asymmetric situation with a specific task-of-interest for which only some tasks, or parts thereof, are relevant.

In some multi-task solutions all tasks are not relevant for all others. In [32], tasks are assumed to come in clusters, and tasks in the same cluster are generated with the same parameters; a similar setup with added parameter noise is used in [34] as one possible scenario. Tasks are also clustered or gated in [3, 5, 11], and based on SVMs in [15]. In [16] a different approach is used, where relevance parameters are learned telling how much the learning of each particular task should be affected by data of other tasks. However, in all these

approaches all tasks are equally important with respect to the clustering or gating or learning of relevance parameters, and there is no specific task-of-interest.

In [21], multi-task learning is combined with semisupervised learning, that is, learning is done from several data sets each of which has some labeled and some unlabeled samples. The semisupervised learning is done in each task, so that for each point, the class prediction is the average prediction of a basic model over a random-walk neighborhood. The multi-task learning is done roughly similarly to [32], by placing a prior on parameters of the basic class predictor models; the prior assumes that each task is either similar to a previous task or comes from a base distribution of new tasks. In this approach, the multi-task learning again considers all tasks equally important with respect to learning the parameters, and there is no specific task-of-interest. It would be interesting to combine semisupervised learning as in [21] also with approaches that assume a single task-of-interest; in this paper we do not yet incorporate semisupervised learning, but rather focus on a novel approach for multi-task learning with a specific task-or-interest.

Coupling of learning tasks is also done in e.g. [12] where instances of several related predicates are extracted from a text corpus, in a semisupervised fashion where an initial model is used to label some unlabeled examples, the model is retrained with the newly labeled examples too, and the iteration goes on. In [12] explicit relationships between predicates are known (for example, two predicates may be known to be mutually exclusive) whereas in our multi-task setting relationships between tasks are not known in advance. Also, the setting of [12] again has no specific task-of-interest.

We point out that our multi-task setting, where we have a specific task-of-interest, can be seen from a semisupervised learning perspective: whether a data sample is relevant to the task-of-interest could be seen as an additional binary-valued label k besides the class label c. Data samples from the task-of-

interest are known to be relevant but for samples from the supplementary tasks the label k is unknown. Estimating the relevant parts of the supplementary tasks can thus be seen as semisupervised learning with respect to k; we will accomplish this by a mixture modeling approach described in the next section.

In some interesting partly heuristic approaches a single task-of-interest is assumed. In [31] a global parameter controls the weight of auxiliary samples in nonparametric classification, or supplementary data are used as support vectors or constraints in SVMs. In [20] extra variables are used to artificially improve the log-likelihood of undesirable samples of auxiliary data, and a constraint on the use of the extra variables forces the model to seek useful auxiliary samples. In [29] combination weights of multiple kernels are learned for SVMs, by a meta classifier that compares the data set from the task-of-interest to other data sets; however, the other data sets are used only for selection of the kernel, not for learning the final classifier for the task-of-interest.

In [25] a cross-validation procedure is used to select which other tasks are useful for learning the task-of-interest; the cross-validation procedure optimizes performance measured on the task-of-interest. However, for each potential selection of other tasks, learning the task-of-interest together with the selected tasks is done with multi-target decision trees where the task-of-interest and the selected other tasks are all equally important.

The most closely related work to our own is the recent method in [8, 9] where there is a single task-of-interest, and a weighting (density ratio) is learned for supplementary data samples. Our approach is different in that we find relevant supplementary samples by modeling whether they can be classified with the same classifier, rather than only using the differences in the spatial distribution of tasks by modeling density ratios between them; another difference is that the model in [8, 9] pools all supplementary data, whereas our model learns simultaneously but separately from each task (supplementary data set).

Our setting is also related to *learning from imbalanced data sets* where some

classes have much fewer samples than others and are thus in danger of being neglected (modeled poorly) by classifiers. Similarly, in our setting the task-of-interest may have much fewer samples than the other tasks put together, and hence some multi-task learning methods might end up neglecting the task-of-interest in favor of other tasks. In this sense, our setting can be seen as *learning from imbalanced tasks*. (Additionally, within each task there may be imbalance of the classes; handling such imbalance could be done by existing approaches, which is complementary to multi-task learning which works across tasks.)

Note that more generally, multi-task learning is one of several complementary approaches for dealing with settings where there are too few samples compared to the number of features and the complexity of the underlying distributions. In particular, dimensionality reduction is often applied to reduce the number of features. Multi-task learning allows learning from more data than the data set from the task of interest alone; thus successful multi-task models may need less dimensionality reduction than single-task models.

2 Relevant subtask learning

In this section we first introduce the principle of relevant subtask learning, then we present a simple logistic regression-based model that builds on this principle, and two learning algorithms (maximum likelihood learning and variational Bayes learning) for the model.

2.1 The principle

Consider a set of classification tasks indexed by $S = 1, ..., M$. Each task S has a small training data set $D_S = \{\mathbf{x}_i, c_i\}_{i=1}^{N_S}$ where the $\mathbf{x}_i \in \mathbb{R}^d$ are d-dimensional input features, the c_i are class labels, and N_S is the number of samples for that task. For simplicity, in this paper we assume all tasks to be two-class classification tasks (c_i is +1 or -1) with the same dimensionality d, but we assume the

process that generates the classes to be different in each task. One task, with index U, is the *task-of-interest*: we want to perform well on this task, because we know that future test data will come from the same distribution as the training data of this task. The other tasks are *supplementary* tasks: in each supplementary task, some unknown portion (0-100%) of the samples are assumed to come from the same distribution as the task-of-interest; the rest come from another distribution which is potentially different for each supplementary task.

We wish to learn to predict classes well for data coming from the task-of-interest. Because the data from the task-of-interest is not abundant enough to learn the task well, we wish to use the other tasks (and their data sets) for help. We are not interested in the other tasks except as a source of information for the task-of-interest. Note that the samples are not paired between tasks; the only connections between the tasks are possible similarities in their underlying distributions, which we do not know beforehand.

The relevant subtask learning problem is to build a classifier, more specifically a model for the class density $p(c|\mathbf{x}, U)$ in task U; the motivation for restricting our interest to task U is that the test data is known to come from this distribution. In addition to data $D_U = \{(c_i, \mathbf{x}_i)\}_{i=1}^{N_U}$ of task U, data D_S from other tasks S are available. The assumption is that some samples of each D_S may come from the distribution $p(c|\mathbf{x}, U)$ but the rest do not.

As usual, the analyst chooses a model family for the task-of-interest, by prior knowledge, or resorting to a nonparametric or semiparametric model. Particular models are denoted by $p(c|\mathbf{x}, U; \mathbf{w}_U)$, where the parameter values $\mathbf{w}_U$ identify the model. The interesting question is how to model the relationships between the task-of-interest and the other tasks, which we discuss next.

For each supplementary task S we assume part of the samples to come from the shared distribution $p(c|\mathbf{x}, U; \mathbf{w}_U)$, part from a different one. Only the former are relevant for modeling the task-of-interest. The analyst must

specify a model for the non-relevant samples as well; typically a nonparametric or semiparametric model would be used to avoid the need to collect specific prior infomation about all tasks. Denote the model for the non-relevant samples of supplementary task S by $p_{\text{nonrelevant}}(c|\mathbf{x}, S; \mathbf{w}_S)$. Since the task S is a mix of relevant and nonrelevant data, we model it by a mixture model:

$$p(c|\mathbf{x}, S; \boldsymbol{\theta}) = (1 - \pi_S)p(c|\mathbf{x}, U; \mathbf{w}_U) + \pi_S p_{\text{nonrelevant}}(c|\mathbf{x}, S; \mathbf{w}_S) \ , \qquad (1)$$

where $\pi_S \in [0, 1]$ is a parameter modeling the mixture proportion of irrelevant samples in task S and $\boldsymbol{\theta}$ denotes all parameters of all tasks. Note that this model reduces to $p(c|\mathbf{x}, U; \mathbf{w}_U)$ for the task-of-interest (where $\pi_S = 0$).

The solution is to use (1) to model the data. The idea behind the functional form is that a flexible enough model for $p_{\text{nonrelevant}}$ "explains away" irrelevant data in the supplementary tasks, and hence $p(c|\mathbf{x}, U; \mathbf{w}_U)$ learns only on the relevant data. In all supplementary tasks, we force one of the subtasks to have the same model $p(c|\mathbf{x}, U; \mathbf{w}_U)$ as the task of interest; by enforcing this, we force the model to find from the supplementary tasks the shared part that is useful for the task of interest. The tradeoff is that to improve performance on the task-of-interest, we spend much computational time to model data of the supplementary tasks too. This is sensible when the bottleneck is the amount of data in the task-of-interest. In this paper we simply call this method *Relevant Subtask Learning* (RSL), but in general "relevant subtask learning" should refer to the learning task rather than any specific method for solving it.

2.2 Logistic regression based model

We derive our solution details for a simple parametric model; it can easily be generalized to more general parametric or semiparametric models. The task-of-interest U is modeled with logistic regression, $p(c|\mathbf{x}, U; \mathbf{w}_U) = (1 + \exp(-c\mathbf{w}_U^T\mathbf{x}))^{-1}$. According to standard practice, the bias has been in-

cluded in the weights $\mathbf{w}_U$, yielding standard logistic regression when a constant element is appended to the input vectors $\mathbf{x}$.

We model the non-relevant data in the other tasks with logistic regression models as well. Each supplementary task S has a different regression model, having its own parameters: $p_{\text{nonrelevant}}(c|\mathbf{x}, S; \mathbf{w}_S) = (1 + \exp(-c\mathbf{w}_S^T\mathbf{x}))^{-1}$, where $\mathbf{w}_S$ is the weight vector. Hence the supplementary tasks are each generated from a mixture of two logistic regression models (with mixture weight π_S):

$$p(c|\mathbf{x}, S; \boldsymbol{\theta}) = (1 - \pi_S)/(1 + \exp(-c\mathbf{w}_U^T\mathbf{x})) + \pi_S/(1 + \exp(-c\mathbf{w}_S^T\mathbf{x})) \qquad (2)$$

where $\boldsymbol{\theta}$ denotes all parameters of all tasks.

2.3 Learning algorithms for RSL

In this paper we present two different methods to learn the above RSL model from data: a simple maximum likelihood algorithm producing a point estimate for the model parameters, and a variational Bayes algorithm.

Maximum likelihood learning. Since the task is to model the distribution of classes given data, the objective function for maximum likelihood learning is the conditional log-likelihood $L_{\text{ml-RSL}} = \sum_S \sum_{i \in D_S} \log p(c_i|\mathbf{x}_i, S; \boldsymbol{\theta})$ where S goes over all tasks including the task-of-interest, and $p(c_i|\mathbf{x}_i, S; \boldsymbol{\theta})$ is given in (2). To optimize the model, we use standard conjugate gradient to maximize $L_{\text{ml-RSL}}$ with respect to the parameters $\mathbf{w}_U$, the $\mathbf{w}_S$, and the π_S. The computational cost per iteration is linear with respect to both the dimensionality and the number of samples. We will call the maximum likelihood algorithm ml-RSL.

The obvious advantage of maximum likelihood learning is that it is very simple; the equally obvious disadvantage is that it does not take uncertainty of parameters into account. Hence, maximum likelihood learning in general is prone to overfitting, especially for high-dimensional data. Variational Bayesian learning should perform better.

Variational Bayes learning. Bayesian inference would correctly take the uncertainty into account and yield the entire posterior distribution of parameters rather than a point estimate, but for models like RSL such inference is intractable. The idea of variational Bayesian inference is that although the posterior distribution of model parameters and latent variables can be complicated, it is possible to approximate the posterior distribution with another distribution having a simpler form. Finding the best approximating distribution is the aim of variational Bayes learning algorithms.

In brief, we use a mean field variational approximation for the posterior, where all component distributions are in the exponential family. Two noteworthy details in the algorithm are: first, because our model is a mixture model we introduce latent variables telling which mixture component generated each data point, and optimize their posterior as well; second, because the logistic functions in the likelihood are not in the exponential family, during optimization we use an exponential-family approximation for them which is updated at each step.

The resulting algorithm has simple update rules: the full forms of the prior, posterior approximation, and resulting update rules are given in Appendix 1. The computational cost per update is again linear with respect to the number of samples (it is not linear with respect to dimensionality, because covariance matrices are used in modeling the posterior). We call this algorithm vb-RSL.

3 Experiments with maximum likelihood RSL

In this section, we compare the performance of the maximum likelihood-optimized version of our method (ml-RSL) against three comparison methods, on two continuums of toy data domains and on a more realistic news classification task. We first introduce the comparison methods and then present each experiment.

3.1 Comparison methods

We compare ml-RSL to three standard approaches which assume progressively stronger relationships between tasks, using simple but comparable models. Because ml-RSL is a maximum likelihood method, for these experiments all comparison methods are also optimized by maximum likelihod, that is, by maximizing the (conditional) class likelihood with a conjugate gradient algorithm. More advanced variational Bayes versions of the methods are compared in Section 4.

One of the most promising multi-task strategies is to assume that tasks come from task clusters, and parameters of tasks are shared within each cluster [32]. We implemented a simplified maximum likelihood-based clustering comparable to the other methods.

Assume that there is a fixed number K of task clusters. To keep complexity comparable to ml-RSL, each cluster k is a mixture of two logistic regression models[1]: $p(c|\mathbf{x}, k; \boldsymbol{\theta}) = \pi_k/(1+\exp(-c\mathbf{w}_{k,1}^T\mathbf{x})) + (1-\pi_k)/(1+\exp(-c\mathbf{w}_{k,2}^T\mathbf{x}))$ where the weight vectors $\mathbf{w}_{k,1}$ and $\mathbf{w}_{k,2}$ and the mixing weight π_k are the parameters of cluster k. Each task is fully generated by one cluster but it is not known which one. The likelihood for task S is $p_S(\boldsymbol{\theta}) = \sum_{k=1}^K \gamma_{k|S} \prod_{i \in D_S} p(c_i|\mathbf{x}_i, k; \boldsymbol{\theta})$ where the parameter $\gamma_{k|S}$ models the probability that task S comes from cluster k.

The parameters are optimized by maximizing the conditional class likelihood $L_{\mathrm{TCM}} = \sum_S \log p_S(\boldsymbol{\theta})$. We call this model "Task Clustering Model" (TCM). It is meant to be a maximum likelihood version of [32], but having a more complex model per cluster (mixture of two instead of one logistic regression model).

We additionally compute with two naive models. "Single-task learning" (ml-STL) uses data of the given task and does not exploit other tasks at all. This may work well if there is a lot of data, otherwise it will overfit. The model should also be good if the other tasks are known to be very different. We simply used a single logistic regression model for single-task learning. The "extreme"

[1] We have additionally checked (results not shown) that ml-RSL outperforms this task-clustering method when only one logistic regression model is used per cluster.

multi-task strategy, here called "all together" (ml-AllTogether), is to learn as if all data from all tasks came from the task-of-interest. This may work well if the tasks are very similar, otherwise the mixture will hide any special properties of the task-of-interest. This strategy is essentially TCM with a single cluster.

3.2 Experiments

We have three experimental settings. In the first two we study on toy data how ml-RSL and TCM tolerate deviations from their assumptions. We then apply the models to news recommendation for one user, when recommendations from other users are available. A note on terminology: A multi-task problem has several tasks, each with its own data. The multi-task problem comes from a *domain* specifying the data distribution in each task.

Experiment 1: When task clustering fails. We created a continuum of multi-task domains where the relationship between the task-of-interest and the other tasks changes. The continuum was set up so that the tasks always follow the assumptions of ml-RSL but the assumption of underlying task clusters in TCM starts to fail progressively. The setting is explained in a schematic diagram in Fig. 1 (top). We created 10 domains and generated 40 learning problems from each. Each problem had 10 tasks. As our aim is to train a classifier for a task-of-interest with too few samples and exploit data from other tasks, the task-of-interest was set to have fewer samples than the others; such a setup arises in several application domains.[2] Each supplementary task had on average the same

[2]For example, consider a news recommendation task where articles are labeled interesting or uninteresting for a specific person who recently started using the recommendation system. Only few articles will be available with labels assigned by that person; this is sometimes called the *new user problem*. In our multi-task approach these are the training data of the task-of-interest. In contrast, persons who have used the system longer may have provided many more labeled articles; such articles are the data of the supplementary tasks. We explore this scenario in more detail in Experiment 3.

number of samples, and the task-of-interest had on average one third of that number. The distribution of the inputs $\mathbf{x}_i$ was Gaussian, and the labels c_i were from a task-dependent mixture of two logistic regression models, with weight vectors chosen differently in each domain, so the domains form a continuum progressively worse for TCM. We lastly added some Gaussian noise to the $\mathbf{x}_i$.

Fig. 1 (bottom) shows mean results for all domains. Here ml-RSL maintains high performance, close to the upper limit.[3] TCM worsens as tasks become less clustered, as expected. The number of clusters in TCM was set to the correct value used when generating the data, to give some advantage to TCM. The naive methods perform poorly. "All together" places all data together which introduces noise as well as useful information. For single-task learning, poor performance and large variance are due to overfitting to the few "proper" training data.

Experiment 2: When relevant subtask modeling fails. Above we showed that when the assumptions of ml-RSL hold and assumptions of TCM do not, ml-RSL is better. Now we study what happens when the assumptions of ml-RSL go wrong. The setting is similar to Experiment 1 and is explained in Fig. 2 (top). Domains were set up so that the assumptions of TCM hold but those of ml-RSL become progressively worse: neither of the two logistic regression models needs to be common to all tasks.

The results are shown in Fig. 2 (bottom left). TCM has high performance for all domains, as expected because the tasks always come from task clusters. Here ml-RSL starts equally good but worsens as its assumptions begin to fail; however, it remains better than the naive methods which behave as in the first experiment.

So far the task-of-interest has had less data than the other tasks, and hence

[3]The bound ("appr. bound" in the figure) was computed by using the parameters with which the data was generated. It is approximate because noise has been added to the inputs.

the supplementary data which do not fit RSL assumptions were able to easily mislead RSL. When the task-of-interest has a comparable amount of data[4] The ml-RSL method performs well for all domains (Fig. 2, bottom right). It locates relevant tasks, that is, tasks from the same task cluster as the task-of-interest. The ml-RSL method does not overfit to the other tasks; it models them mostly with the task-specific model.

Experiment 3: Predicting document relevance. We now apply the methods to a recommendation task (see [1] for a general review of recommender systems, and [18, 26, 30] for details of a recent prize-winning movie rating method).

Here we compare the methods in a *news recommendation task*, where documents (news articles) must be classified as relevant or not to the interests of a specific user, and the classification is learned based on a set of articles labeled by that user and sets of articles labeled by other users. Because each user has different, subjective interests, learning to predict relevance for a specific user is a multi-task problem suitable for the relevant subtask learning approach.

We have real news abstracts from the Reuters-21578 collection, and simulated users so that we can control the problem domain. Each "user" classifies articles as interesting or not. The goal is to learn to predict interestingness for a "user-of-interest."

We use a very simple artificial simulation of subjective interests of users: the "user-of-interest" labels news interesting if they belong to a specific topic. In the Reuters collection the abstracts come with labels of topic categories; we chose the category with the label "acq" as the topic for the user-of-interest. The other users are interested in "acq" part of the time, but otherwise they are interested in another category specific to each user.[5]

[4] A similar increase in the data amount was tested not to help TCM in Experiment 1.

[5] A more complex simulation could be done by bringing in advanced models of user behavior, or we could simply gather actual subjective labelings of article relevance from real users. With real users, we could ask the users to directly give labels, as in for instance the news filtering

The learning problem can be seen as combining collaborative filtering and content-based prediction. Earlier work on such a combination includes, for instance, [6] (partly heuristic kernel combination) and [23] (naive Bayes imputation followed by collaborative filtering). Note that ml-RSL is more general than [6] which needs samples rated by several users during learning (to estimate meaningful correlation kernels) whereas ml-RSL requires none.

In the experiments, we used a simplistic feature extraction, including stop-word removal, bag of words representation encoded as document vectors, selecting most "informative" words, discarding too sparse documents, and dimensionality reduction by linear discriminant analysis. As a design parameter we varied how often the other users labeled according to "acq" on average. The user-of-interest had less data than others; test data were left-out documents from the user-of-interest. We repeated the experiment 10 times to reduce variation due to initializations and small datasets.

The result is that ml-RSL performs best, as shown in Fig. 3. Since there is little data for the user-of-interest, single-task learning overfits badly. TCM[6] and "all together" perform about equally here. At the extreme limit where all data begins to be relevant, performances of ml-RSL, TCM and "all together" naturally converge.

4 Experiments with variational Bayes RSL

In this section, we compare the performance of the variational Bayes version of relevant subtask learning (vb-RSL) against three comparison methods, on one continuum of toy data domains and on two more realistic classification tasks. As before, we first present the comparison methods and then each experiment.

interface used in [14], or we could infer labels from user actions. Sample actions could be clicking of hyperlinks, from which rough evidence about subjective interests could be inferred.

[6] We used $K = 6$ clusters to have roughly equally many parameters in ml-RSL and TCM.

4.1 Comparison methods

We again compare our method to three others, all of which are here optimized by variational Bayes learning.

The "symmetric multitask learning" method (SMTL; [32]) is essentially a hierachical model where the prior causes information to be shared across tasks: it is assumed that each task (data set) can be modeled by one logistic regression model, and that the tasks arise from clusters, so that a single logistic regression model is used for all tasks inside each cluster. A Dirichlet process prior is used for the clusterings, allowing a potentially infinite number of clusters for the tasks. Posterior distributions over the clusterings and over the logistic regression parameters of each cluster are approximated by variational Bayes. We used the SMTL implementation provided by the authors of [32].

We additionally compute two naive variational methods. In single-task learning (vb-STL) we learn a single logistic regression model for the task-of-interest, from its own data set only; that is, we ignore all the other data sets. The learning algorithm of this naive method is essentially a special case of the vb-RSL algorithm, obtained by setting the prior relevance probability of all supplementary tasks to zero. The other naive method assumes all data to be relevant for the tasks of interest: then one can pool all data together and learn a single logistic regression model from the pooled data by variational Bayes. We call this method "all together" (vb-AllTogether); it can be obtained from the vb-RSL algorithm by setting the prior relevance probability of supplementary tasks to one.

4.2 Experiments

Experiment 1: Continuum of problem domains. For the first experiment we created a single continuum of multi-task classification problem domains, with the aim of showing that both vb-RSL and the main comparison

method SMTL have their own domains where they work well. It is then up to the analyst to try (or decide based on domain knowledge) which model is best for each particular real-life application.

At the left end of the continuum (Fig. 4, top), the tasks (data sets) come from clusters so that in each cluster, a single logistic regression model suffices to classify the data. Such multitask problems follow exactly the assumptions of SMTL. Conversely, at the right end of the continuum, all of the tasks are unique, but all of them contain some proportion of relevant data: that is, each task is generated by a mixture of two logistic regression models where one component is specific to the task and the other (relevant) component is shared by all tasks. Such multitask problems follow the assumptions of vb-RSL. The intermediate domains form a continuum between these two kinds of multitask problems.

For each domain in the continuum, we generated 30 multitask classification problems: each multitask problem has 19 data sets with 9 dimensions and 30 samples. We ran all methods on all multitask problems, and evaluated their mean performance on each domain.

The results are shown in Fig. 4 (bottom). We can see that, just as expected, both SMTL and vb-RSL work well for domains matching their assumptions. We also see that the naive methods (single-task learning and "all together") perform badly; note especially that using a variational Bayes algorithm for single-task learning (STL) is not enough to make it comparable to the multitask methods.

Experiment 2: Heart disease data. In the second experiment we apply the methods to the real-life "Heart" dataset from the UCI Machine Learning Repository. The data contain 13 attributes for 270 patients; the attributes include age, sex, clinical measurements such as resting blood pressure and serum cholesterol, and chest pain type. The patients have different chest pains (four types) and some of the patients have heart disease.

Our goal is to learn a separate predictor for patients having each chest pain

type, predicting whether the patient has heart disease; this is a multitask problem where each chest pain type is used as the task-of-interest in turn, and the other chest pain types are the supplementary tasks. Correspondingly, we divide the data to four sets according to chest pain type; there are 20, 42, 79, and 129 samples in the four sets.

When the goal is to learn for a particular chest pain type (task-of-interest), we divide the data of the task-of-interest to training and test samples; we learn the methods from the training samples of the task-of-interest and all samples of the supplementary tasks, and we evaluate the performance of the methods on the test samples of the task-of-interest. We generate 50 repetitions of this problem by re-dividing the task-of-interest data each time, and run all methods for each repetition.

For this data, the choice of the prior for the logistic regression parameters affects the performances. The usual wide prior (Gaussian prior with diagonal covariance matrix $\sigma^2\mathbf{I}$, $\sigma^2 = 10$; results shown in Fig. 5) seems to lead to overfitting, since the results are generally better when a stronger regularization is used by applying a narrower prior ($\sigma^2 = 0.1$; results shown in Fig. 6). In both figures, the results are shown separately for each of the four tasks (chest pain types), as a function of how many samples from the task-of-interest were used for training.

In most cases vb-RSL gives the best results; the difference is especially clear in Task 2 (Fig. 5). "All together" and vb-RSL are equally good in Task 1 (Fig. 6); SMTL is slightly better than vb-RSL in Task 4 when moderate numbers (30-45) of training samples are used, but vb-RSL is better with small numbers of samples (Fig. 6). In Task 2 the wide prior gives the best vb-RSL results, and for the other tasks the narrow prior is better. Note that we did not here study methods for choosing a good prior, but simply point out that standard methods such as cross-validation can be used.

Experiment 3: Predicting document relevance. For the last experiment we performed the same kind of document relevance prediction experiment as we did with the maximum likelihood methods in Section 3.2, but this time using the variational Bayes methods. To briefly recap the setting, the task is to predict relevance of news articles for a "user-of-interest," given a labeled data set from that user and several data sets from other users. The user-of-interest considers articles from a specific news topic category to be relevant (we used the topic category with the label "acq" in the Reuters collection), whereas the other users want such articles only part of the time, and at other times consider another (user-specific) category relevant. We generate several repetitions (here 10) of this problem and run all methods for each repetition, evaluating their performance on left-out news articles from the user-of-interest.

We found that the performance of the methods depends a lot on the parameters of the problem (dimensionality, number of tasks, amount of data, and proportion of relevant data). In terms of these problem parameters there seems to be a region where vb-RSL performs well whereas outside the region performance deteriorates. We investigated this, showing the performance as a function of these four problem parameters.

The results are shown in Fig. 7. The vb-RSL method gives the best results when there are many dimensions but few samples per data set (less than 100), which is a realistic scenario. SMTL is not much better than naive "all together" on this data. Results of vb-RSL improve when the dimensionality grows. Interestingly, vb-RSL worsens when the number of samples is too high: this may be because the number of latent variables in the RSL mixture model grows with data, and with too many variables the factorized variational posterior no longer approximates the real posterior well.

5 Conclusions

We introduced a new problem, *relevant subtask learning*, where multiple supplementary tasks are available to help learn one task-of-interest. We showed how a carefully constructed but generally applicable hierarchical model solves the problem; our model is a *multitask mixture model* where the idea is to model relevant parts of other tasks with a shared mixture component, and nonrelevant parts by (at least equally) flexible models, to avoid a performance tradeoff between the task-of-interest and the other tasks.

Using logistic regression as an example, we presented two learning algorithms for the resulting model: a maximum likelihood learning algorithm (ml-RSL) and a variational Bayes algorithm (vb-RSL).

Our approach is a viable alternative to the more traditional hierarchical modeling-based multitask learning. For the maximum likelihood method we showed that it outperforms a comparable maximum likelihood-based traditional multi-task learning model and two naive alternatives, both on toy domains and on more realistic text classification. For the variational Bayes method the natural comparison methods were a comparable variational Bayes-based traditional multi-task learning model (SMTL) and two naive alternatives; we studied toy domains, heart disease data, and text classification. We found that our method and SMTL both have their own domains where they do well; it is up to the analyst to decide which method to use for each application.

Our method is not restricted to logistic regression or to supervised learning; the method will be generalized to more general models in the next stage. In further work we will also improve the variational inference and investigate which applications are best suited for the relevant subtask learning approach.

Acknowledgments.

J. Peltonen and S. Kaski belong to Helsinki Institute for Information Technology HIIT and the Adaptive Informatics Research Centre, a Centre of Excellence of the Academy of Finland. They were supported by the Academy of Finland, decision numbers 207467 and 123983. Y. Yaslan was supported by the Center for International Mobility CIMO. This work was also supported in part by the IST Programme of the European Community, PASCAL2 Network of Excellence. This publication only reflects the authors' views. We thank Y. Xue for providing SMTL code.

Appendix 1: Variational Bayes learning for RSL

In brief, a variational Bayes learning algorithm minimizes the Kullback-Leibler divergence between the approximating distribution and the true posterior distribution, that is,

$$D_{KL}(q,p) = \int_{\Theta} q(\Theta;\Phi) \log \frac{q(\Theta;\Phi)}{p(\Theta|D)} d\Theta \propto \int_{\Theta} q(\Theta;\Phi) \log \frac{q(\Theta;\Phi)}{p(D|\Theta)p(\Theta)} d\Theta \quad (3)$$

where Θ denotes all parameters and hidden variables of the model, q is the variational approximation of the posterior distribution of Θ, and $p(\Theta|D)$ is the true posterior distribution given the data. The variational approximation is controlled by hyperparameters Φ, and the learning algorithm minimizes the divergence with respect to Φ.

Form of the prior. For RSL the model parameters are the shared logistic regression vector $\mathbf{w}_U$, the task-specific logistic regression vectors $\mathbf{w}_S$, and the mixing probabilities π_S. The latent variables are the relevance indicators $k_{S,i}$ for each data point i in each task S, except for the task of interest where all data are known to be relevant. We must choose a prior and a posterior approximation

for the parameters and latent variables. We will use the following prior:

$$p(\Theta) = p(\mathbf{w}_U) \prod_S \left(p(\mathbf{w}_S) p(\pi_S) \prod_{i=1}^{N_S} p(k_{S,i}|\pi_S) \right) . \tag{4}$$

We set all the components of the above prior to be in the exponential family; we use Normal priors for the logistic regression vectors, Beta priors for the mixing probabilities and Bernoulli priors for the latent variables (relevance indicators):

$$p(\mathbf{w}_U) = N(\mathbf{w}_U; \boldsymbol{\mu}_{U,0}; \boldsymbol{\Sigma}_{U,0}) \tag{5}$$

$$p(\mathbf{w}_S) = N(\mathbf{w}_S; \boldsymbol{\mu}_{S,0}; \boldsymbol{\Sigma}_{S,0}) \tag{6}$$

$$p(\pi_S) = Beta(\pi_S; \phi_{S,A0}, \phi_{S,B0}) = \frac{\Gamma(\phi_{S,A0} + \phi_{S,B0})}{\Gamma(\phi_{S,A0})\Gamma(\phi_{S,B0})} \pi_S^{\phi_{S,B0}} (1 - \pi_S)^{\phi_{S,A0}} \tag{7}$$

$$p(k_{S,i}|\pi_S) = Bernoulli(k_{S,i}; \pi_S) = \pi_S^{k_{S,i}} (1 - \pi_S)^{1 - k_{S,i}} \tag{8}$$

where Γ denotes the standard gamma function.

The specific values of the prior parameters can in principle be set by domain knowledge if available. In the experiments we simply set the $\boldsymbol{\mu}$ to zero vectors and the $\boldsymbol{\Sigma}$ to diagonal matrices with a suitably large variance (here 10); noninformative priors $p(\pi_S)$ were used by setting the pseudodata counts to one, $\phi_{S,A0} = \phi_{S,B0} = 1$.

Form of the posterior approximation. We will use a mean-field posterior approximation, that is, a completely factorized functional form

$$q(\Theta; \Phi) = q(\mathbf{w}_U) \prod_S \left(q(\mathbf{w}_S) q(\pi_S) \prod_{i=1}^{N_S} q(k_{S,i}) \right) . \tag{9}$$

This is almost the same functional form as the prior; the only difference is that the posteriors of the latent variables $k_{S,i}$ do not depend on π_S. All the component distributions are again in the exponential family, with Normal distributions for logistic regression vectors, Beta distributions for mixing parameters and Bernoulli distributions for latent variables:

$$q(\mathbf{w}_U) = N(\mathbf{w}_U; \boldsymbol{\mu}_U; \boldsymbol{\Sigma}_U) \tag{10}$$

$$q(\mathbf{w}_S) = N(\mathbf{w}_S; \boldsymbol{\mu}_S; \boldsymbol{\Sigma}_S) \tag{11}$$

$$q(\pi_S) = Beta(\pi_S; \phi_{S,A}, \phi_{S,B}) = \frac{\Gamma(\phi_{S,A} + \phi_{S,B})}{\Gamma(\phi_{S,A})\Gamma(\phi_{S,B})} \pi_S^{\phi_{S,B}} (1 - \pi_S)^{\phi_{S,A}} \tag{12}$$

$$q(k_{S,i}) = Bernoulli(k_{S,i}; \gamma_{S,i}) = \gamma_{S,i}^{k_{S,i}} (1 - \gamma_{S,i})^{1 - k_{S,i}} \tag{13}$$

Approximation for the likelihood. For variational Bayes learning all the distributions should be in the exponential family; however, for RSL there is a complication because the likelihood term includes logistic probabilities which are not in the exponential family. To make variational Bayes learning possible, we will use a separate lower bound for each of those logistic probabilities as in [32], as follows. To simplify the notation, we denote $\mathbf{y}_{S,i} = c_{S,i}\mathbf{x}_{S,i}$ and define the following function:

$$g(\xi) = \frac{2}{\xi} \cdot \frac{1 - \exp(-\xi)}{1 + \exp(-\xi)} \,. \tag{14}$$

With this notation we can write a lower bound for each logistic term as

$$\begin{aligned} &\frac{1}{1 + \exp(-\mathbf{w}_S^T \mathbf{y}_{S,i})} \\ &\geq \frac{1}{1 + \exp(-\xi_{S,i,S})} \exp\left(\frac{\mathbf{w}_S^T \mathbf{y}_{S,i} - \xi_{S,i,S}}{2} - g(\xi_{S,i,S}) \frac{(\mathbf{w}_S^T \mathbf{y}_{S,i})^2 - \xi_{S,i,S}^2}{8} \right) \text{ and} \end{aligned} \tag{15}$$

$$\begin{aligned} &\frac{1}{1 + \exp(-\mathbf{w}_U^T \mathbf{y}_{S,i})} \\ &\geq \frac{1}{1 + \exp(-\xi_{S,i,U})} \exp\left(\frac{\mathbf{w}_U^T \mathbf{y}_{S,i} - \xi_{S,i,U}}{2} - g(\xi_{S,i,U}) \frac{(\mathbf{w}_U^T \mathbf{y}_{S,i})^2 - \xi_{S,i,U}^2}{8} \right) \end{aligned} \tag{16}$$

where the $\xi_{S,i,S}$ and the $\xi_{S,i,U}$ are parameters of the approximation. In the variational Bayes algorithm we then iteratively optimize the cost function, first with respect to the $\xi_{S,i,S}$ and the $\xi_{S,i,U}$, then with respect to the parameters and latent variables, and so on. The resulting update rules for all parameters are given below.

Result: Update rules for RSL. For the task of interest U, the covariance matrix $\mathbf{\Sigma}_U$ and mean $\boldsymbol{\mu}_U$ of the logistic regression vector are updated, one after the other, by

$$\mathbf{\Sigma}_U = \left(\frac{1}{4} \sum_S \sum_{i=1}^{N_S} (1 - \gamma_{S,i}) g(\xi_{S,i,U}) \mathbf{x}_{S,i} \mathbf{x}_{S,i}^T + \mathbf{\Sigma}_{U,0}^{-1} \right)^{-1} \text{ and} \tag{17}$$

$$\boldsymbol{\mu}_U = \mathbf{\Sigma}_U \cdot \left(\frac{1}{2} \sum_S \sum_{i=1}^{N_S} (1 - \gamma_{S,i}) \mathbf{y}_{S,i} + \mathbf{\Sigma}_{U,0}^{-1} \boldsymbol{\mu}_{U,0} \right) . \tag{18}$$

For each of the other tasks S, the covariance matrix $\mathbf{\Sigma}_S$ and mean $\boldsymbol{\mu}_S$ of the task specific classifier vector are updated, one after the other, by

$$\mathbf{\Sigma}_S = \left(\frac{1}{4} \sum_{i=1}^{N_S} \gamma_{S,i} g(\xi_{S,i,S}) \mathbf{x}_{S,i} \mathbf{x}_{S,i}^T + \mathbf{\Sigma}_{S,0}^{-1} \right)^{-1} \text{ and} \tag{19}$$

$$\boldsymbol{\mu}_S = \mathbf{\Sigma}_S \cdot \left(\frac{1}{2} \sum_{i=1}^{N_S} \gamma_{S,i} \mathbf{y}_{S,i} + \mathbf{\Sigma}_{S,0}^{-1} \boldsymbol{\mu}_{S,0} \right) . \tag{20}$$

For these other tasks S, the pseudodata counts $\phi_{S,B}$ and $\phi_{S,B}$ (for the overall relevance parameter) are updated by

$$\phi_{S,B} = \phi_{S,B0} + \sum_{i=1}^{N_S} \gamma_{S,i} \quad \text{and} \quad \phi_{S,A} = \phi_{S,A0} + \sum_{i=1}^{N_S} (1 - \gamma_{S,i}) . \tag{21}$$

For samples from the task of interest, the probability $\gamma_{S,i}$ of being irrelevant for the task-of-interest is naturally kept at zero, but for each data point i in the other tasks S, the probability is updated by

$$\gamma_{S,i} = \frac{\exp(h_{S,i,1})}{\exp(h_{S,i,1}) + \exp(h_{S,i,2})} \tag{22}$$

where the term $h_{S,i,1}$ measures how accurately the class of the point is predicted by the task-specific model, that is,

$$\begin{aligned} h_{S,i,1} = \frac{1}{2} \boldsymbol{\mu}_S^T (\mathbf{y}_{S,i}) - \frac{g(\xi_{S,i,S})}{8} \mathbf{x}_{S,i}^T \left[\mathbf{\Sigma}_S + \boldsymbol{\mu}_S \boldsymbol{\mu}_S^T \right] \mathbf{x}_{S,i} + \Psi(\phi_{S,B}) \\ - \log(1 + \exp(-\xi_{S,i,S})) - \frac{\xi_{S,i,S}}{2} + \frac{g(\xi_{S,i,S})}{8} \xi_{S,i,S}^2 \end{aligned} \tag{23}$$

and the term $h_{S,i,2}$ measures how accurately the class of the point is predicted by the task-of-interest model, that is,

$$h_{S,i,2} = \frac{1}{2}\boldsymbol{\mu}_U^T\left(\mathbf{y}_{S,i}\right) - \frac{g(\xi_{S,i,U})}{8}\mathbf{x}_{S,i}^T\left[\boldsymbol{\Sigma}_U + \boldsymbol{\mu}_U\boldsymbol{\mu}_U^T\right]\mathbf{x}_{S,i} + \Psi(\phi_{S,A}) \\ - \log(1+\exp(-\xi_{S,i,U})) - \frac{\xi_{S,i,U}}{2} + \frac{g(\xi_{S,i,U})}{8}\xi_{S,i,U}^2 \ . \quad (24)$$

In equations (23) and (24), Ψ is the standard psi function (derivative of the logarithm of the standard gamma function).

Lastly, for each data point i in data set S, the approximation parameters $\xi_{S,i,S}$ for logistic predictions yielded by task-specific models are updated by

$$\xi_{S,i,S} = \left(\mathbf{x}_{S,i}^T\left(\boldsymbol{\Sigma}_S + \boldsymbol{\mu}_S\boldsymbol{\mu}_S^T\right)\mathbf{x}_{S,i}\right)^{1/2} \quad (25)$$

where the quadratic form inside the parentheses is simply a scalar. Similarly, the approximation parameters $\xi_{S,i,U}$ for logistic predictions yielded by the task-of-interest model are updated by

$$\xi_{S,i,U} = \left(\mathbf{x}_{S,i}^T\left(\boldsymbol{\Sigma}_U + \boldsymbol{\mu}_U\boldsymbol{\mu}_U^T\right)\mathbf{x}_{S,i}\right)^{1/2} \ . \quad (26)$$

Only the latter equation is needed in the task of interest since all points are known to be relevant, so all predictions are made by the task-of-interest model.

References

[1] G. Adomavicius and A. Tuzhilin, Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions, *IEEE Transactions on Knowledge and Data Engineering* **17** (2005) 734-749.

[2] A. Argyriou, T. Evgeniou and M. Pontil, Convex multi-task feature learning, *Machine Learning* **73** (2008) 243-272.

[3] A. Argyriou, A. Maurer and M. Pontil, An Algorithm for Transfer Learning in a Heterogeneous Environment, in: *Machine Learning and Knowledge Dis-*

covery in Databases (proceedings of ECML PKDD 2008), Part I, W. Daelemans, B. Goethals and K. Morik, eds., Springer, Berlin Heidelberg, 2008, pp. 71-85.

[4] A. Argyriou, C.A. Micchelli, M. Pontil and Y. Ying, A Spectral Regularization Framework for Multi-Task Structure Learning, in: *Advances in Neural Information Processing Systems 20*, J.C. Platt, D. Koller, Y. Singer and S. Roweis, eds., MIT Press, Cambridge, MA, 2008, pp. 25-32.

[5] B. Bakker and T. Heskes, Task clustering and gating for Bayesian multitask learning, *Journal of Machine Learning Research* **4** (2003) 83-99.

[6] J. Basilico and T. Hofmann, Unifying Collaborative and Content-Based Filtering, in: *Proceedings of the 21st International Conference on Machine Learning (ICML 2004)*, R. Greiner and D. Schuurmans, eds., Omnipress, Madison, WI, 2004, pp. 65-72.

[7] J. Bi, T. Xiong, S. Yu, M. Dundar and R.B. Rao, An Improved Multi-task Learning Approach with Applications in Medical Diagnosis, in: *Machine Learning and Knowledge Discovery in Databases (proceedings of ECML PKDD 2008), Part I*, W. Daelemans, B. Goethals and K. Morik, eds., Springer, Berlin Heidelberg, 2008, pp. 117-132.

[8] S. Bickel, J. Bogojeska, T. Lengauer and T. Scheffer, Multi-task learning for HIV therapy screening, in: *Proceedings of the 25th Annual International Conference on Machine Learning (ICML 2008)*, A. McCallum and S. Roweis, eds., Omnipress, 2008, pp. 56–63.

[9] S. Bickel, C. Sawade and T. Scheffer, Transfer Learning by Distribution Matching for Targeted Advertising, in: *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio and L. Bottou, eds., 2009, pp. 145-152.

[10] E. Bonilla, K.M.A. Cai and C.K.I. Williams, Multi-task Gaussian Process Prediction, in: *Advances in Neural Information Processing Systems 20*, J.C. Platt, D. Koller, Y. Singer and S. Roweis, eds., MIT Press, Cambridge, MA, 2008, pp. 153-160.

[11] E.V. Bonilla, F.V. Agakov and C.K.I. Williams, Kernel Multi-task Learning using Task-specific Features, in: *Proceedings of AISTATS 2007, the 11th International Conference on Artificial Intelligence and Statistics*, M. Meila and X. Shen, eds., JMLR Workshop and Conference Proceedings, 2007, pp. 43-50.

[12] A. Carlson, J. Betteridge, E.R.H. Jr. and T.M. Mitchell, Coupling Semi-Supervised Learning of Categories and Relations, in: *Proceedings of the NAACL HLT 2009 Workshop on Semi-supervised Learning for Natural Language Processing*, 2009.

[13] R. Caruana, Multitask Learning, *Machine Learning* **28** (1997) 41-75.

[14] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes and M. Sartin, Combining Content-Based and Collaborative Filters in an Online Newspaper, in: *Proceedings of the ACM SIGIR Workshop on Recommender Systems—Implementation and Evaluation*, 1999.

[15] T. Evgeniou, C.A. Micchelli and M. Pontil, Learning Multiple Tasks with Kernel Methods, *Journal of Machine Learning Research* **6** (2005) 615-637.

[16] J. Fang, S. Ji, Y. Xue and L. Carin, Multitask Classification by Learning the Task Relevance, *IEEE Signal Processing Letters* **15** (2008) 593-596.

[17] S. Kaski and J. Peltonen, Learning from Relevant Tasks Only, in: *Machine Learning: ECML 2007*, J.N. Kok, J. Koronacki, R.L. Mantaras, S. Matwin, D. Mladenič and A. Skowron, eds., Springer, Berlin Heidelberg, 2007, pp. 608–615.

[18] Y. Koren, The BellKor Solution to the Netflix Grand Prize, in: *Netflix prize documentation*, 2009.

[19] S.I. Lee, V. Chatalbashev, D. Vickrey and D. Koller, Learning a Meta-Level Prior for Feature Relevance from Multiple Related Tasks, in: *Proceedings of the 24th International Conference on Machine Learning*, Z. Ghahramani, eds., ACM, New York, NY, 2007, pp. 489-496.

[20] X. Liao, Y. Xue and L. Carin, Logistic Regression with an Auxiliary Data Source, in: *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, L. De Raedt and S. Wrobel, eds., Omnipress, Madison, WI, 2005, pp. 505-512.

[21] Q. Liu, X. Liao, H. Li, J.R. Stack and L. Carin, Semisupervised Multitask Learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **31**(6) (2009) 1074-1086.

[22] Z. Marx, M.T. Rosenstein and L.P. Kaelbling, Transfer Learning with an Ensemble of Background Tasks, in: *Inductive Transfer: 10 Years Later, NIPS 2005 workshop*, 2005.

[23] P. Melville, R.J. Mooney and R. Nagarajan, Content-Boosted Collaborative Filtering for Improved Recommendations, in: *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI-2002)*, AAAI Press, 2002, pp. 187-192.

[24] A. Niculescu-Mizil and R. Caruana, Inductive Transfer for Bayesian Network Structure Learning, in: *Proceedings of AISTATS 2007, the 11th International Conference on Artificial Intelligence and Statistics*, M. Meila and X. Shen, eds., JMLR Workshop and Conference Proceedings, 2007, pp. 339-346.

[25] B. Piccart, J. Struyf and H. Blockeel, Empirical Asymmetric Selective Transfer in Multi-objective Decision Trees, in: *Discovery Science (proceedings*

of DS 2008), J.F. Bouliçaut, M.R. Berthold and T. Horváth, eds., Springer, Berlin Heidelberg, 2008, pp. 64-75.

[26] M. Piotte and M. Chabbert, The Pragmatic Theory Solution to the Netflix Grand Prize, in: *Netflix prize documentation*, 2009.

[27] R. Raina, A.Y. Ng and D. Koller, Transfer learning by constructing informative priors, in: *Inductive Transfer: 10 Years Later, NIPS 2005 workshop*, 2005.

[28] M.T. Rosenstein, Z. Marx and L.P. Kaelbling, To Transfer or Not To Transfer, in: *Inductive Transfer: 10 Years Later, NIPS 2005 workshop*, 2005.

[29] U. Rückert and S. Kramer, Kernel-Based Inductive Transfer, in: *Machine Learning and Knowledge Discovery in Databases (proceedings of ECML PKDD 2008), Part II*, W. Daelemans, B. Goethals and K. Morik, eds., Springer, Berlin Heidelberg, 2008, pp. 220-233.

[30] A. Töscher, M. Jahrer and R.M. Bell, The BigChaos Solution to the Netflix Grand Prize, in: *Netflix prize documentation*, 2009.

[31] P. Wu and T.G. Dietterich, Improving SVM Accuracy by Training on Auxiliary Data Sources, in: *Proceedings of the 21st International Conference on Machine Learning (ICML 2004)*, R. Greiner and D. Schuurmans, eds., Omnipress, Madison, WI, 2004, pp. 871–878.

[32] Y. Xue, X. Liao, L. Carin and B. Krishnapuram, Multi-Task Learning for Classification with Dirichlet Process Priors, *Journal of Machine Learning Research* **8** (2007) 35–63.

[33] K. Yu, V. Tresp and A. Schwaighofer, Learning Gaussian Processes from Multiple Tasks, in: *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, L. De Raedt and S. Wrobel, eds., ACM Press, 2005, pp. 1012-1019.

[34] J. Zhang, Z. Ghahramani and Y. Yang, Flexible latent variable models for multi-task learning, *Machine Learning* **73** (2008) 221-242.

Figure captions

Fig. 1: Comparison of methods on multitask problem domains progressively less suited for TCM. **Top**: conceptual illustration; columns are domains and rows are tasks within a domain. Tasks (data sets) are generated from a mixture of two logistic regression models (weight vectors shown as lines). One subtask (line with closed ball) corresponds to the task-of-interest and appears in all tasks. The other subtask (line with open ball) is common to task clusters in the leftmost domain; in the rightmost domain it differs for each task. **Bottom**: Results, averaged over 40 multitask problems for each domain. Here ml-RSL performs well; TCM worsens progressively. The difference at right is significant (Wilcoxon signed rank test).

Fig. 2: Comparison of methods on multitask problem domains progressively less suited for ml-RSL. **Top**: conceptual illustration. Tasks are always clustered; tasks in a cluster are generated with the same model. In the leftmost domain, one subtask (equaling the task-of-interest) is the same in all clusters. In the rightmost domain, clusters are completely different. All domains can be learned by TCM; ml-RSL fits the leftmost domain well but not the rightmost one. **Bottom left**: Results for a continuum of 10 domains (10 tasks in each; results are averages over 40 replicates); only little data in the task-of-interest. **Bottom right**: Results when the amount of data in the task-of-interest is comparable to the other tasks.

Fig. 3: Comparison of ml-RSL to TCM and two naive maximum likelihood methods on Reuters data. Average results over 10 generated problems are shown, as a function of one design parameter, the average probability that a sample is relevant to the task-of-interest. The ml-RSL method performs the best. Performance of single-task learning varies highly due to overlearning; the

worst results (at design parameter values 0.75 and 0.95) do not fit in the figure.

Fig. 4: Comparison of vb-RSL to SMTL and two naive variational Bayes methods on a continuum of toy multitask problem domains. **Top**: conceptual illustration. At the left end tasks come in clusters, with a single logistic regression per cluster, which is ideal for SMTL; at the right end all tasks have an individual component and a shared component, which is ideal for vb-RSL. The gray shades denote the mixing proportion of the shared component, which vanishes at the left end of the continuum. **Bottom**: Results, averaged over 30 multitask problems for each domain. Numbers are performance differences compared to optimal results (difference of average class log-likelihood, compared to results obtained with known generating parameters); lines show the average difference on each domain, and error bars show standard deviation of the mean. At the left end SMTL performs best, and at the right end vb-RSL performs best, which matches the design of the problem domains. The naive methods perform poorly as expected.

Fig. 5: Comparison of vb-RSL to SMTL and two naive methods on heart disease data, using a wide prior for logistic regression parameters. The task-of-interest is to predict presence of heart disease for a given type of chest pain, using other chest pain types as supplementary tasks. This multitask problem has 12 input dimensions and 4 tasks (chest pain types). In each subfigure we select one task (chest pain type) as the task of interest and vary the number of training data from that task, using the rest as test data.

Fig. 6: Comparison of vb-RSL to SMTL and two naive methods on heart disease data, using a narrow prior for logistic regression parameters. The setting is the same as in Fig. 5, but the prior is narrower. In each subfigure we select one task (chest pain type) as the task of interest and vary the number of

training data from that task, using the rest as test data.

Fig. 7: Comparison of vb-RSL to SMTL and two naive methods on Reuters data. We studied performance as the domain deviates from the "default settings" of 150 input dimensions, 10 tasks (data sets), 75 samples per data set where the relevance proportion (proportion of samples that are relevant for the task-of-interest) is 0.5. In each subfigure we vary one of these parameters while keeping the others fixed; the curves show test-set class prediction performance. **Top left**: results as a function of the relevance proportion (dimensionality, number of tasks, and number of samples are fixed). **Top right**: results as a function of data dimensionality. **Bottom left**: results as a function of the number of samples per data set. **Bottom right**: results as a function of the number of tasks (data sets).

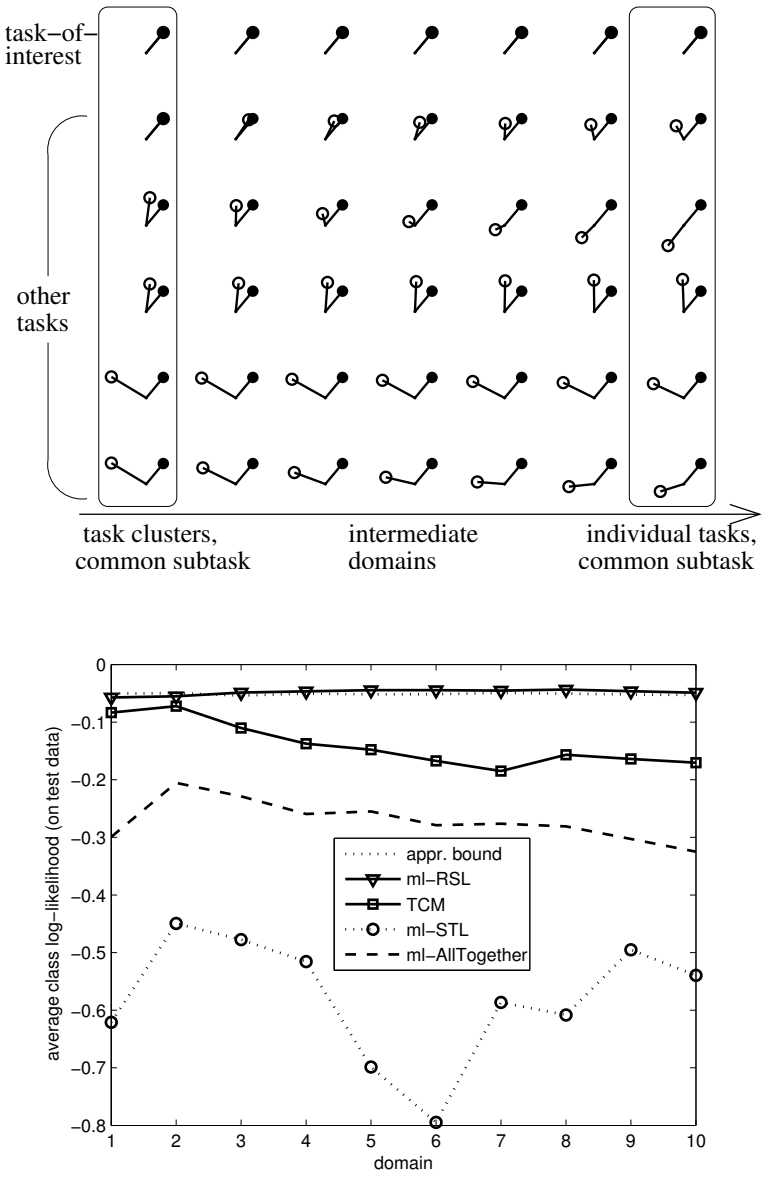

task−of−
interest
other
tasks
task clusters,
common subtask
intermediate
domains
individual tasks,
common subtask
average class log−likelihood (on test data)
domain
appr. bound
ml−RSL
TCM
ml−STL
ml−AllTogether
0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
1
2
3
4
5
6
7
8
9
10

Fig. 1

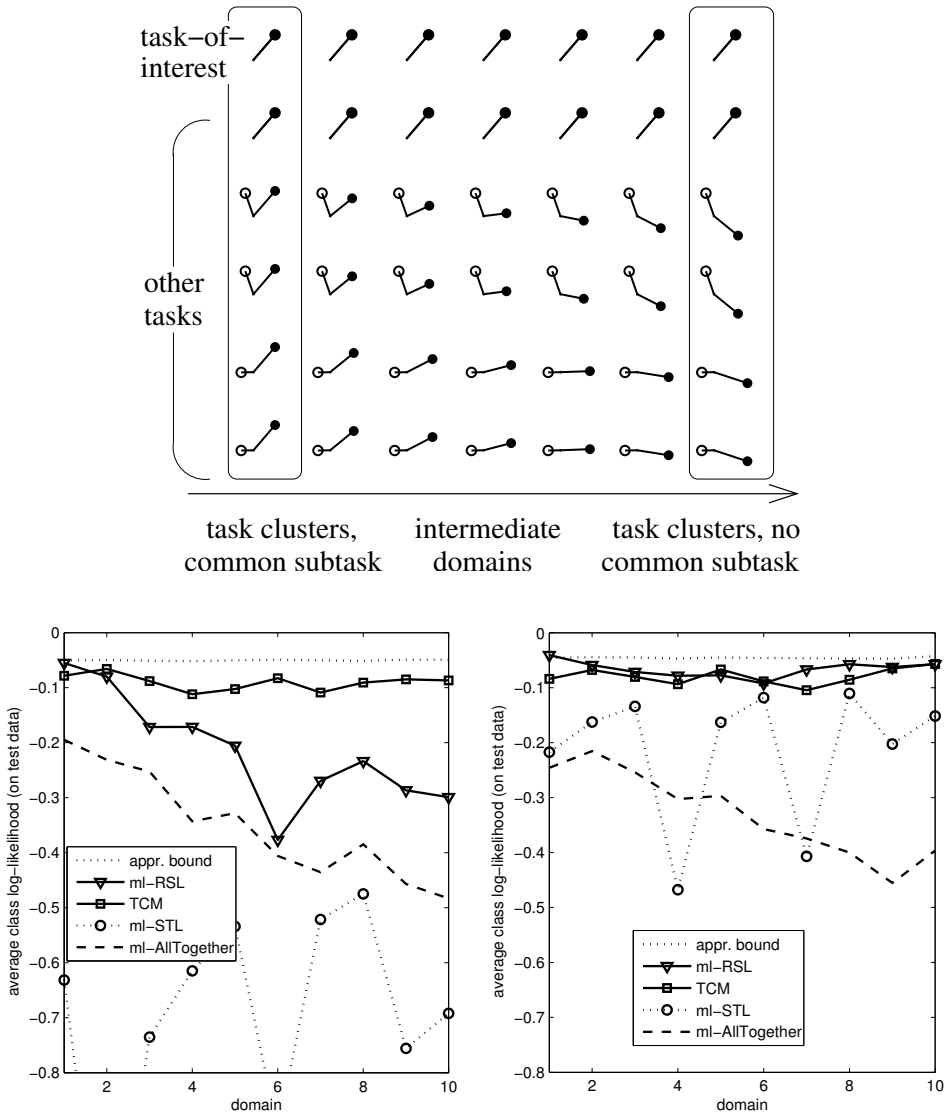

task−of−
interest
other
tasks
task clusters,
common subtask
intermediate
domains
task clusters, no
common subtask
average class log−likelihood (on test data)
domain
appr. bound
ml−RSL
TCM
ml−STL
ml−AllTogether

Fig. 2

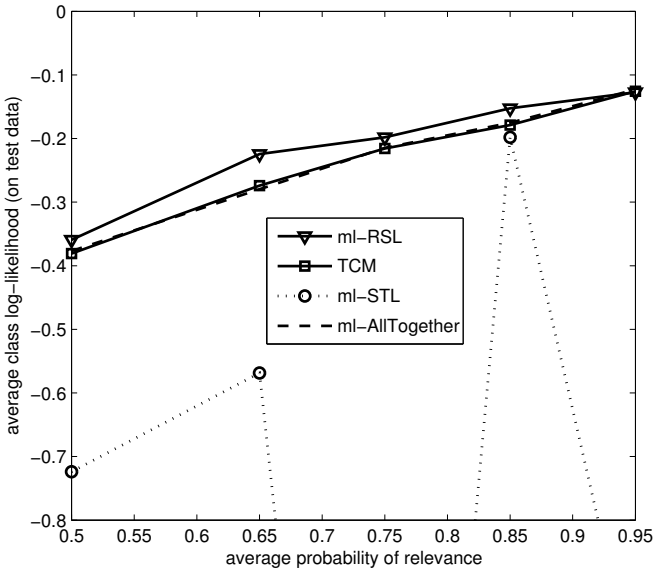
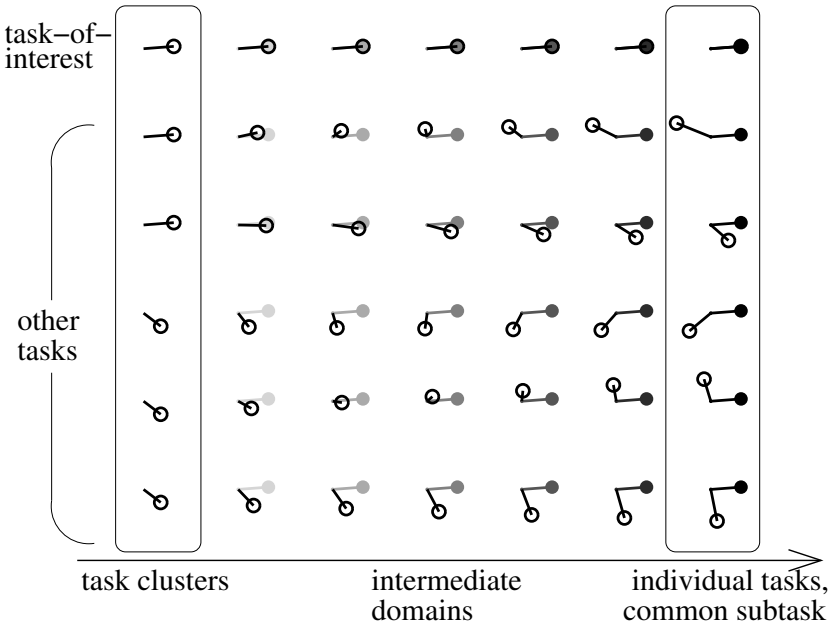

average class log−likelihood (on test data)
average probability of relevance
0
−0.1
−0.2
−0.3
−0.4
−0.5
−0.6
−0.7
−0.8
0.5
0.55
0.6
0.65
0.7
0.75
0.8
0.85
0.9
0.95
ml−RSL
TCM
ml−STL
ml−AllTogether

Fig. 3


task−of−
interest
other
tasks
task clusters
intermediate
domains
individual tasks,
common subtask

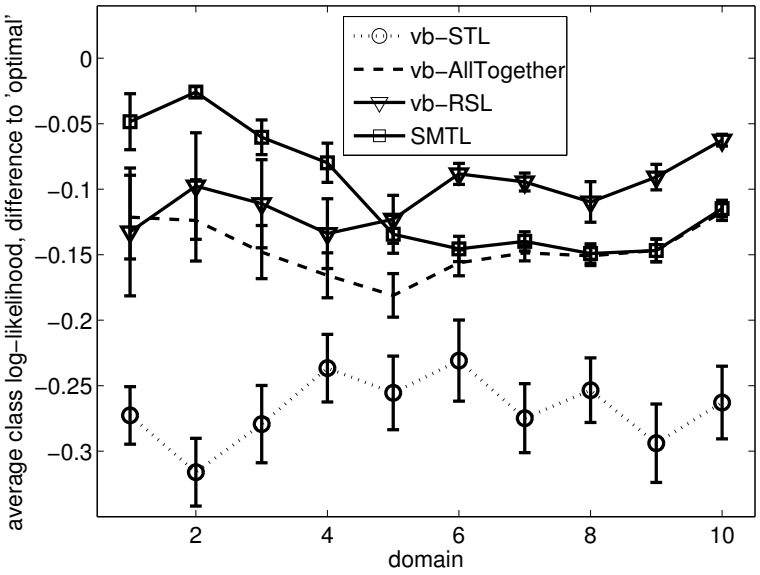

vb−STL
vb−AllTogether
vb−RSL
SMTL
average class log−likelihood, difference to 'optimal'
0
−0.05
−0.1
−0.15
−0.2
−0.25
−0.3
2
4
6
8
10
domain

Fig. 4

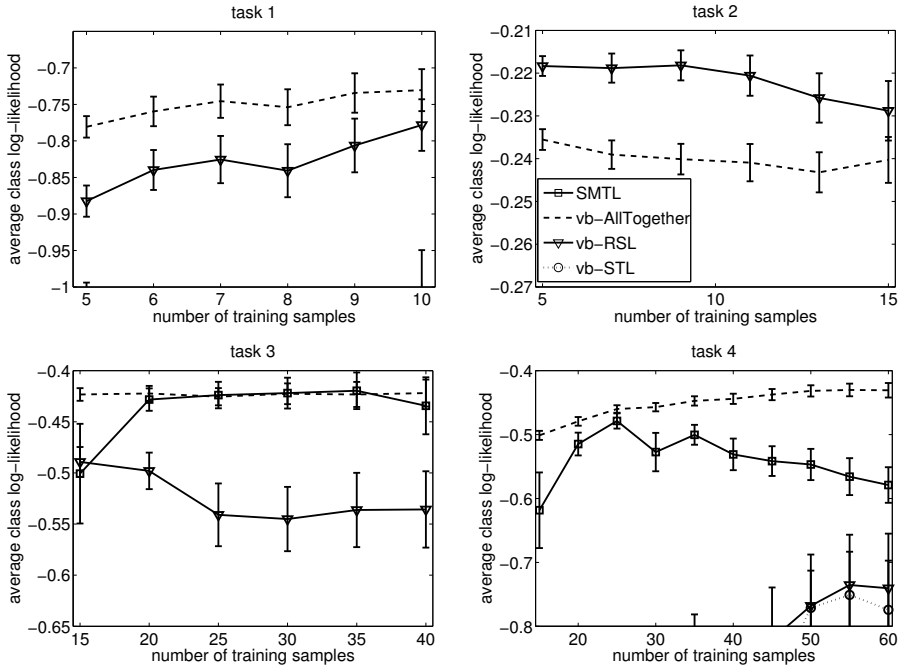

task 1
task 2
task 3
task 4
average class log−likelihood
number of training samples
SMTL
vb−AllTogether
vb−RSL
vb−STL

Fig. 5

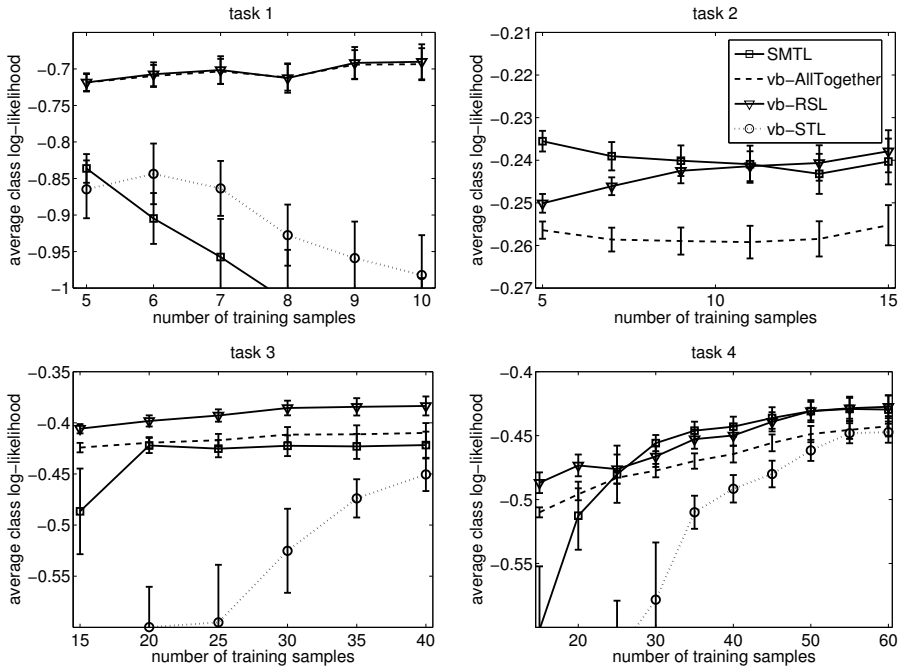

task 1
task 2
task 3
task 4
average class log−likelihood
number of training samples
SMTL
vb−AllTogether
vb−RSL
vb−STL

Fig. 6

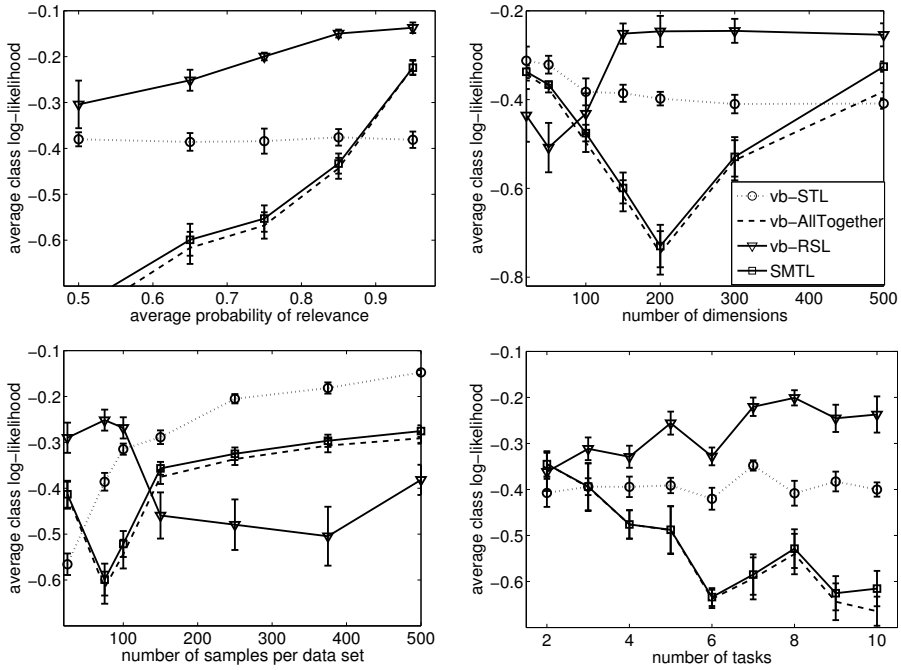

average class log−likelihood
average probability of relevance
number of dimensions
number of samples per data set
number of tasks
vb−STL
vb−AllTogether
vb−RSL
SMTL

Fig. 7