Learning from Relevant Tasks Only

Samuel Kaski and Jaakko Peltonen**

Laboratory of Computer and Information Science, Helsinki University of Technology,
P.O. Box 5400, FI-02015 TKK, Finland
{samuel.kaski,jaakko.peltonen}@tkk.fi

Abstract. We introduce a problem called relevant subtask learning, a variant of multi-task learning. The goal is to build a classifier for a task-of-interest having too little data. We also have data for other tasks but only some are relevant, meaning they contain samples classified in the same way as in the task-of-interest. The problem is how to utilize this "background data" to improve the classifier in the task-of-interest. We show how to solve the problem for logistic regression classifiers, and show that the solution works better than a comparable multi-task learning model. The key is to assume that data of all tasks are 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.

Key words: multi-task learning, relevant subtask learning

1 Introduction

All too often in classification tasks there is too little training data to estimate sufficiently powerful models. This problem is ubiquitous in bioinformatics; it appears also in image classification from few examples, finding of relevant texts, etc. Possible solutions are 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; e.g., 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?*

This is a special type of multi-task learning problem. In multi-task learning [1], 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. [2, 3]). By contrast, our problem is fundamentally asymmetric and more structured: test data fits one task, the "*task-of-interest*,"

** The authors contributed equally to the work.

and other tasks may contain *subtasks* relevant for the task-of-interest, but no other task needs to be wholly relevant. Our models are better suited in the task-of-interest 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. Tasks may be learned together [4–6] or a new task can use a prior learned from previous tasks [2, 3]. Both approaches model all tasks symmetrically. Support vector machines (SVMs) have been used in symmetric hierarchical modeling as well (e.g. [7]). 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 [8] tasks are assumed to come in clusters, and tasks in the same cluster are generated with the same parameters. Tasks are also clustered or gated in [9], and based on SVMs in [7]. In all these approaches all tasks are equally important with respect to the clustering so there is no specific task-of-interest.

In some interesting partly heuristic approaches a single task-of-interest is assumed. In [10] a global parameter controls the weight of auxiliary samples in nonparametric classification, or background data are used as support vectors or constraints in SVMs. In [11] 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.

2 Relevant Subtask Learning

Consider a set of classification tasks indexed by $S = 1, ..., M$. Each task S has a training data set $D_S = \{\mathbf{x}_i, c_i\}_{i=1}^{N_S}$ where $\mathbf{x}_i \in \mathbb{R}^d$ are d-dimensional input features, 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 are two-class classification tasks (c_i is +1 or -1) with the same d, but the process that generates the classes is different in each task. One task, with index U, is the *task-of-interest*. The other tasks are *supplementary* tasks; in each, some 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, potentially different for each supplementary task.

We wish to learn to predict classes well for data coming from the task-of-interest. We are not interested in the other tasks except as a source of information for the task-of-interest. There are no paired samples between tasks; the only connections between tasks are possible similarities in their underlying distributions.

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, because 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 come from the same 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 collecting prior infomation about all tasks. Denote the model for the non-relevant samples of subtask S by $p_{\text{nonrelevant}}(c|\mathbf{x}, S; \mathbf{w}_S)$. Since task S is a mix of relevant and nonrelevant data, its model should be

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

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 auxiliary subtasks, and hence $p(c|\mathbf{x}, U; \mathbf{w}_U)$ learns only on the relevant data. By forcing one of the subtasks to use the same parameters in all tasks, we force the model to find from the other tasks the common 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. We call this method *Relevant Subtask Model* (RSM).

We introduce our solution with a simple parametric model; it can easily be generalized to more general parametric or semiparametric models. We model the task-of-interest U with logistic regression, $p(c|\mathbf{x}, U; \boldsymbol{\theta}) = (1 + \exp(-c\mathbf{w}_U^T\mathbf{x}))^{-1}$. We include the bias in the weights $\mathbf{w}_U$, yielding standard logistic regression when one element in the inputs $\mathbf{x}$ is constant.

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: $p_{\text{nonrelevant}}(c|\mathbf{x}, S; \boldsymbol{\theta}) = (1 + \exp(-c\mathbf{w}_S^T\mathbf{x}))^{-1}$, 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):

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

In this first paper we use simple optimization and spend effort in designing controlled experiments. More advanced methods will be added in later papers.

Since the task is to model the distribution of classes given data, the objective function is the conditional log-likelihood $L_{\text{RSM}} = \sum_S \sum_{i \in D_S} \log p(c_i|\mathbf{x}_i, S; \boldsymbol{\theta})$ 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 RSM, we use standard conjugate gradient to maximize

L_{RSM} 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 dimensionality and number of samples.

3 Comparison Methods

As a proof-of-concept we compare RSM to three standard approaches, which assume progressively stronger relationships between tasks, using simple but comparable models, all optimized by maximizing the (conditional) likelihood with a conjugate gradient. More advanced versions will be compared in later work.

One of the most promising multi-task strategies is to assume tasks come from task clusters, and parameters of tasks are shared within each cluster [8]. We implement a simplified maximum likelihood-based clustering comparable to the other methods. Generality is not reduced: all approaches can in principle be given a state-of-the-art full-Bayesian treatment.

Assume there is a fixed number K of task clusters. To keep complexity comparable to RSM, 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 unknown which. The class probability of 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 [8], but having a more complex model per cluster (mixture of two instead of one logistic regression model).

We try two naive models. "Single-task learning" uses data of the given task, but does not exploit other tasks. This may work well if there is a lot of data, otherwise it will overfit. It is also 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" multi-task strategy, here called "all together", is: learn as if all data from all tasks came from the task-of-interest. This may work well if tasks are very similar, otherwise the mixture will hide the features of the task-of-interest. This strategy is essentially TCM with a single cluster.

4 Experiments

We have three experimental settings. In the first two we study how RSM and TCM tolerate deviations from their assumptions. We then study news classification according to the interest of one user, when classifications 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, and the relationships of the tasks.

[1] We have checked that RSM outperforms a regular one-submodel clustering.

Experiment 1: When Task Clustering Fails. Here 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 the tasks always follow the assumptions of RSM but the assumption of underlying task clusters in TCM starts to fail. The setting is explained in a schematic diagram in Fig. 1 (left). We created 10 domains and generated 40 learning problems from each. Each problem had 10 tasks; the task-of-interest had less samples than others. Inputs $\mathbf{x}_i$ were Gaussian and 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 $\mathbf{x}_i$.[2]

Fig. 1 (right) shows average results for all domains. RSM 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 small "proper" training data.

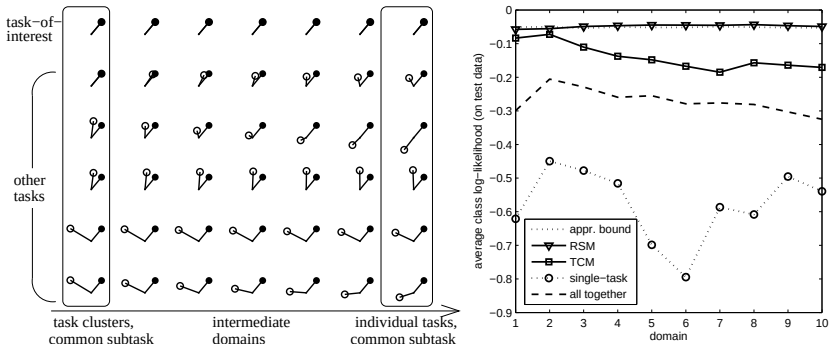

Fig. 1. Comparing methods on domains progressively less suited for TCM. **Left**: conceptual illustration; columns are domains and rows are tasks within a domain. Tasks (data sets) are generated from a mixture of two logistic regression models (weight vectors shown as lines). One subtask (line with closed 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. **Right**: Results, averaged over 40 problems for each domain. RSM performs well; TCM worsens progressively. The difference at right is significant (Wilcoxon signed rank test).

[2] More details about all experiments can be found in [12].

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

Experiment 2: When Relevant Subtask Modeling Fails. Above we showed that when the assumptions of RSM hold better it outperforms TCM. Now we show what happens when the assumptions of RSM go wrong. The setting is similar to experiment 1 and is explained in Fig. 2 (left). Domains were set up so that assumptions of TCM hold but those of RSM become progressively worse: neither of the two logistic regression models needs to be common to all tasks.

The results are shown in Fig. 2 (middle). TCM has high performance for all domains, as expected because the tasks always come from task clusters. RSM 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 had less data than the others, meaning that RSM tries to retrieve relevant tasks with little information for the "query." When the task-of-interest has a comparable amount of data[4] RSM performs well for all domains (Fig. 2 (right)). It locates relevant tasks (ones from the same task cluster as the task-of-interest). RSM does not overfit to the other tasks; it models them mostly with the task-specific model. This demonstrates successful "information retrieval" of relevant tasks.

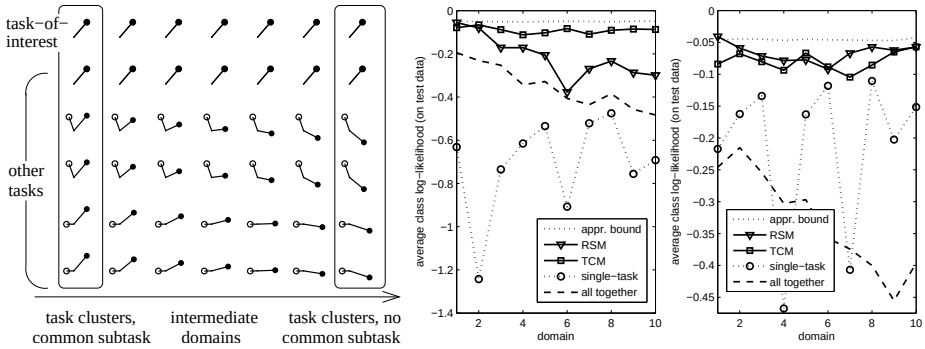

Fig. 2. Comparison of methods on domains progressively less suited for RSM. **Left**: conceptual illustration. Tasks are always clustered; tasks in a cluster are generated with the same model. In the leftmost domain, one subtask (equaling the task-of-interest) is the same in all clusters. In the rightmost domain, clusters are completely different. All domains can be learned by TCM; RSM fits the leftmost domain well but not the rightmost one. **Middle**: 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. **Right**: Results when the amount of data in the task-of-interest is comparable to the other tasks.

Experiment 3: Predicting Document Relevance. Here we have real news from the Reuters-21578 collection but simulated users to 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," who labels news interesting if they

[4] A similar increase in the data amount does not help TCM in experiment 1.

belong to the category "acq." The other users are interested in "acq" part of the time, but otherwise they are interested in another category specific to each user. The problem can be seen as combining collaborative filtering and content-based prediction. Earlier work includes e.g. [13] (partly heuristic kernel combination) and [14] (naive Bayes imputation followed by collaborative filtering). Note that RSM is more general than [13] which needs samples rated by several users (to estimate meaningful correlation kernels) whereas RSM requires none.

We used a simplistic feature extraction, including stopword removal etc., vector representation, selecting most "informative" words, discarding too sparse documents, and dimensionality reduction by linear discriminant analysis; see [12] for details. 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.

Results are shown in Fig. 3. RSM performs best. Since there is little data for the user-of-interest, single-task learning overfits badly. TCM[5] and "all together" perform about equally here. At the extreme where all data begins to be relevant, performances of RSM, TCM and "all together" naturally converge.

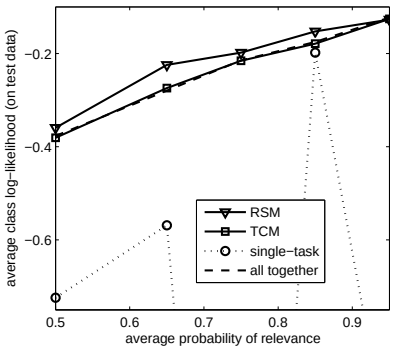

Fig. 3. Comparison of RSM to TCM and two naive methods on Reuters data. Average results over 10 generated problems are shown, as a function of one design parameter, the average probability that a sample is relevant to the task-of-interest. RSM performs the best. Performance of single-task learning varies highly due to overlearning; the worst results (at design parameter values 0.75 and 0.95) do not fit in the figure.

5 Conclusions

We introduced a new problem, *relevant subtask learning*, where multiple background tasks are used to learn one task-of-interest. We showed how a carefully

[5] We used $K = 6$ clusters to have roughly equally many parameters in RSM and TCM.

constructed but generally applicable graphical model solves the problem; 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 showed that the resulting "Relevant Subtask Model" (RSM) outperforms a comparable traditional multi-task learning model and two naive alternatives, on toy domains and on more realistic text classification.

The method is not restricted to logistic regression or to supervised learning. Here we used simple maximum conditional likelihood estimators, which will be generalized to full-Bayesian treatments of more general models in the next stage.

Acknowledgments. The authors belong to Helsinki Institute for Information Technology and the Adaptive Informatics Research Centre. They were supported by the Academy of Finland, decision numbers 108515 and 207467. This work was also supported in part by the IST Programme of the European Community, PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views. All rights are reserved because of other commitments.

References

1. Caruana, R.: Multitask learning. Mach. Learn. **28** (1997) 41–75
2. Marx, Z., Rosenstein, M.T., Kaelbling, L.P.: Transfer learning with an ensemble of background tasks. In: Inductive Transfer: 10 Years Later, NIPS workshop. (2005)
3. Raina, R., Ng, A.Y., Koller, D.: Transfer learning by constructing informative priors. In: Inductive Transfer: 10 Years Later, NIPS workshop. (2005)
4. Niculescu-Mizil, A., Caruana, R.: Inductive transfer for Bayesian network structure learning. In: Proceedings of AISTATS. (2007) Electronic proceedings.
5. Rosenstein, M.T., Marx, Z., Kaelbling, L.P.: To transfer or not to transfer. In: Inductive Transfer: 10 Years Later, NIPS workshop. (2005)
6. Yu, K., Tresp, V., Schwaighofer, A.: Learning Gaussian processes from multiple tasks. In: Proceedings of ICML 2005. ACM Press (2005) 1012–1019
7. Evgeniou, T., Micchelli, C.A., Pontil, M.: Learning multiple tasks with kernel methods. J. Mach. Learn. Res. **6** (2005) 615–637
8. Xue, Y., Liao, X., Carin, L., Krishnapuram, B.: Multi-task learning for classification with Dirichlet process priors. J. Mach. Learn. Res. **8** (2007) 35–63
9. Bakker, B., Heskes, T.: Task clustering and gating for Bayesian multitask learning. J. Mach. Learn. Res. **4** (2003) 83–99
10. Wu, P., Dietterich, T.G.: Improving SVM accuracy by training on auxiliary data sources. In: Proceedings of ICML 2004. Omnipress, Madison, WI (2004) 871–878
11. Liao, X., Xue, Y., Carin, L.: Logistic regression with an auxiliary data source. In: Proceedings of ICML 2005. Omnipress, Madison, WI (2005) 505–512
12. Kaski, S., Peltonen, J.: Learning from relevant tasks only. Technical Report E11, Helsinki University of Technology, Lab. of Comp. and Information Science (2007)
13. Basilico, J., Hofmann, T.: Unifying collaborative and content-based filtering. In: Proceedings of ICML 2004. Omnipress, Madison, WI (2004) 65–72
14. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: Proceedings of AAAI-2002. AAAI Press (2002) 187–192