Computer Science and Artificial Intelligence Laboratory

Technical Report

MIT-CSAIL-TR-2007-053 November 6, 2007

Transfering Nonlinear Representations using Gaussian Processes with a Shared Latent Space

Raquel Urtasun, Ariadna Quattoni, and Trevor Darrell

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

Transfering Nonlinear Representations using Gaussian Processes with a Shared Latent Space

Raquel Urtasun **Ariadna Quattoni** **Trevor Darrell**
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
{rurtasun,ariadna,trevor}@csail.mit.edu

Abstract

When a series of problems are related, representations derived from learning earlier tasks may be useful in solving later problems. In this paper we propose a novel approach to transfer learning with low-dimensional, non-linear latent spaces. We show how such representations can be jointly learned across multiple tasks in a discriminative probabilistic regression framework. When transferred to new tasks with relatively few training examples, learning can be faster and/or more accurate. Experiments on a digit recognition task show significantly improved performance when compared to baseline performance with the original feature representation or with a representation derived from a semi-supervised learning approach.

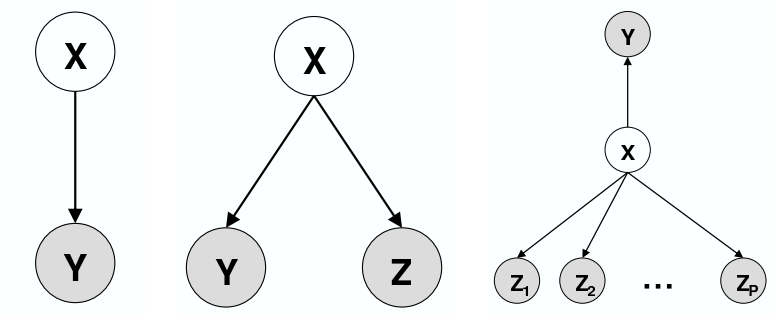
1 Introduction

When faced with a new task, it is advantageous to exploit knowledge and structures found useful in solving related problems. A common paradigm to exploit such knowledge is to learn a feature space from previous tasks and transfer that representation to a future task. Ideally, the transferred representation is of lower dimension than the raw feature space, and the set of functions implied by the new representation still contains the optimal classifier for the new task. When this is the case, the new task can be learned more robustly and/or with fewer training examples in the transferred space than in the raw space.

Several methods have been proposed which exploit a shared intermediate representation to constrain learning across multiple tasks [2, 3, 12]. Recently, intermediate representations have also been discovered using manifold learning over a classifier weight space [1]. Generally, these approaches have been limited to learning deterministic and/or linear representations. Transfer of probabilistic representations has been explored in a Gaussian Processes (GP) paradigm, by explicitly sharing a covariance function and/or kernel hyperparameters across tasks [7, 17]. However, previous methods for transfer learning with GPs did not explicitly discover a low-dimensional representation.

In this paper we propose a novel approach to transfer learning based on discovering a low-dimensional, non-linear latent space jointly across tasks in a discriminative probabilistic regression framework, and transferring that space to future tasks. We show that when there are relatively few training examples in the new task, learning can be improved significantly.

We build our discriminative probabilistic model on top of the Gaussian Process Latent Variable Model (GP-LVM) [5]. GP-LVMs can learn non-linear probabilistic low dimensional representations of observed data, generalizing Probabilistic Principal Components Analysis (PPCA). PPCA [13] marginalizes over the latent coordinates, making it impractical to share a latent space across multiple tasks; in contrast, the GP-LVM optimizes the latent coordinates, and marginalizes over weights. Optimization with respect to latent variables facilitates the introduction of additional constraints on the reduced dimensional representation, e.g., dynamical models [15], hierarchical constructions [6], preservation of local topologies [8] or joint optimization of regression problems [11].



(a) (b) (c)

Figure 1: **Graphical models** of (a) GPLVM, (b) our probabilistic discriminative model for a single task, and (c) our Discriminative Transfer model.

Here, we propose a novel approach to transfer learning which jointly optimizes latent variables to accurately reconstruct the data *and* solve multiple tasks. Latent spaces can be shared across tasks, and/or transferred to new tasks. Our method can be thought of as generalizing the deterministic linear latent spaces of [1], or as allowing GP covariances defined over low dimensional spaces to be transferred across tasks.

Experiments on digit recognition tasks indicate that the ability to transfer non-linear, low-dimensional features across problems can provide significant performance improvements, especially when the target task has relatively few training examples compared to the source problems used to learn the latent space. Baseline experiments confirm that learning the shared latent space discriminatively is important; PCA-based semi-supervised learning underperformed transfer learning with representations learned discriminatively.

In the remainder of the paper, we first describe our approach to learn a discriminative latent space jointly with a single classification task. We then extend this to the multi-task setting, jointly optimizing the latent space to account for each task as well as the underlying data. Finally, we present a transfer learning formalism, where a latent space learned on previous tasks is used in a new classification or regression task. We experiment with different digit recognition tasks, and conclude with a discussion of our method and avenues for future work.

2 Probabilistic Discriminative Latent Variable Model

Conventional classification methods suffer when applied to problems with high dimensional input spaces and very small training sets. If, however, the high dimensional data in fact lie on a low-dimensional manifold, accurate classification may be possible with a small amount of training data if that manifold is discovered by the classification method.

Here, we take advantage of the probabilistic nature of the Gaussian Process Latent Variable Model (GPLVM)[5] to perform classification by jointly learning a low dimensional generative model of the data and a discriminative regressor. Joint learning was previously used by Shon et al. [11] to learn a common structure between two regression tasks. Here, although the graphical model is the same (Fig. 1 (b)), the context is very different; we are interested in learning a latent space that can discriminate between classes. In particular we want to learn a low dimensional manifold to best generate the data and perform classification. Urtasun and Darrell [14] introduced a new method that learns a discriminative probabilistic low dimensional latent space by applying a prior distribution on the latent space based on LDA, but their method can not be directly extended to a multi-task setting.

2.1 Learning a discriminative low dimensional space

Jointly learning latent variables and classifier parameters is challenging when techniques such as PPCA [13] are used to model the low dimensional structure of the data, as they marginalize out the latent variables. The GPLVM is a generalization of PPCA to the non-linear case that marginalizes the weights and optimizes the latent coordinates; as a consequence, one can jointly learn the latent

space to reconstruct the data and perform classification as a regression task. This allows us to share latent variables across problems and transfer representations.

More formally, let $\mathbf{Y} = [\mathbf{y}_1, \cdots, \mathbf{y}_N]^T$ be the matrix composed of all the observations, with $\mathbf{y}_i \in \Re^D$, and let $\mathbf{Z} = [\mathbf{z}_1, \cdots, \mathbf{z}_N]^T$ be the matrix composed of all the labels, with $\mathbf{z}_i \in \Re^S$. Let $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]^T$ be the matrix whose rows represent the unobserved positions of the training data in latent space, $\mathbf{x}_i \in \Re^q$. We model the distribution of the observations given the latent coordinates as a Gaussian Process

$$p(\mathbf{Y} \,|\, \mathbf{X}, \bar{\beta}) \;=\; \frac{1}{\sqrt{(2\pi)^{ND}|\mathbf{K}_Y|^D}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{Y}^T\right)\right), \tag{1}$$

where elements of the kernel matrix $\mathbf{K}_Y$ are defined by the covariance function, $(\mathbf{K}_Y)_{i,j} = k_Y(\mathbf{x}_i, \mathbf{x}_j)$. In particular, we use a kernel that is the sum of an RBF, a bias or constant term, and a noise term,

$$k_Y(\mathbf{x}, \mathbf{x}') \;=\; \beta_1 \exp\left(-\frac{\beta_2}{2}||\mathbf{x} - \mathbf{x}'||^2\right) + \frac{\delta_{\mathbf{x},\mathbf{x}'}}{\beta_3} + \beta_4\,, \tag{2}$$

where $\bar{\beta} = \{\beta_1, \cdots, \beta_4\}$ comprises the kernel hyperparameters that govern the output variance, the RBF support width, the bias and the variance of the additive noise, respectively.

In our framework, instead of performing classification on the raw feature space we take advantage of the low dimensional representation of the data and perform classification using a mapping from latent coordinates to labels, $p(\mathbf{Z}|\mathbf{X})$.

We use a least squares classification technique for this mapping [10]. This may be suboptimal since the latent coordinates are more constrained than with Gaussian process classification (GPC),[1] however learning and inference can be performed analytically, avoiding the computational overhead of classical GPC methods. Despite this approximation, our method works well in practice, and we expect the performance to improve when extended to GPC.

Under these assumptions the likelihood of the labels is

$$p(\mathbf{Z} \,|\, \mathbf{X}, \bar{\gamma}) \;=\; \frac{1}{\sqrt{(2\pi)^{NS}|\mathbf{K}_Z|^S}} \exp\left(-\frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Z^{-1}\mathbf{Z}\mathbf{Z}^T\right)\right). \tag{3}$$

where $\bar{\gamma}$ are the kernel hyperparameters.

Assuming independence of the observations and labels given the latent coordinates (see Fig. 1 (b)), learning the joint model is equivalent to minimizing the negative log likelihood $\mathcal{L}$ with respect to the latent coordinates $\mathbf{X}$, and the kernel hyperparameters of both mappings, $\bar{\beta}, \bar{\gamma}$,

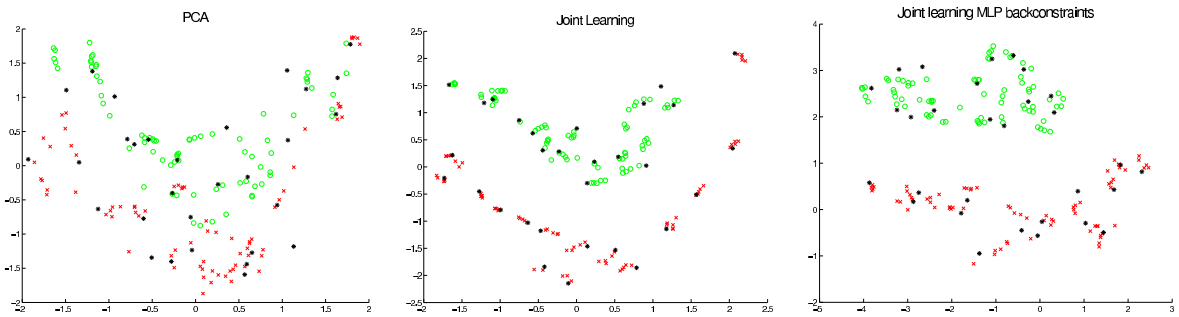
$$\mathcal{L} \;=\; \frac{D}{2}\ln|\mathbf{K}_Y| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Y^{-1}\mathbf{Y}\mathbf{Y}^T\right) + \frac{S}{2}\ln|\mathbf{K}_Z| + \frac{1}{2}\mathrm{tr}\left(\mathbf{K}_Z^{-1}\mathbf{Z}\mathbf{Z}^T\right) \tag{4}$$

$$+ \sum_i \ln\beta_i + \sum_i \ln\gamma_i + C_1 \tag{5}$$

where C_1 is a constant.

This independence assumption allows the use of different kernels for the different mappings. For example, smoothness might be important for the latent to observation mapping, an RBF being a good kernel choice. For the latent to labels mapping, smoothness might not be that important and one can use other kernels, such as the Matérn kernel [10]. Here we use the kernel in (2) for both mappings.

[1] Gaussian process classification discriminatively models $p(\mathbf{Z}|\mathbf{Y})$ as a Bernouilli distribution. The probability of success is related to an unconstrained intermediate function, which is mapped to the unit interval by a sigmoid function (e.g., logit, probit) to yield a probability. Unlike the regression case, in the GPC, neither the posterior, the marginal likelihood, nor the predictions can be computed analytically. Several approximations [4, 9, 16] have been proposed based on sampling or on analytic approximations (e.g., Laplace, Expectation-Propagation), but are in general computationally expensive. The extension of our joint learning framework to this more complex approach is the subject of future research.



(a) (b) (c)

Figure 2: **Probabilistic discriminative model**. Low dimensional representations learned using (a) PCA, (b) our probabilistic discriminative model (section 2), and (c) our probabilistic discriminative model with back-constraints. The data is composed of two different classes. The training examples of the two classes are depicted in red and green. The latent coordinates estimated for the test data are shown in black. Note how the discriminative model separates the classes with or without back-constraints. The inclusion of back-constraints significantly speeds inference.

2.2 Inference

To speed up inference, we take advantage of the back-constrained GPLVM [8] and constrain the latent space to be a function of the observations

$$x_{nj} = g_j(\mathbf{y}_n; \mathbf{w}) \tag{6}$$

where x_{nj} is the n-th component of $\mathbf{x}_n$. In particular we use a multilayer perceptron to represent this mapping

$$g_j(\mathbf{y}) = w_{ij} \sum_{i=1}^{h} \sigma(\mathbf{u}_i^T \mathbf{y}) \,, \tag{7}$$

where $\sigma(u) = 1/(1+\exp(-u))$ is the sigmoid function and $\mathbf{w} = \{w_{ij}\}$ is the set of parameters. When learning the back-constraint model, the minimization is done with respect to $\mathbf{w}$ instead of the latent coordinates. As a consequence of back-constraining the model, optimization of the test latent coordinates is no longer necessary and inference is very fast. Given a new test point, $\mathbf{y}'$, its latent position, $\mathbf{x}'$, is obtained by evaluating (6). The test labels are estimated by thresholding the mean prediction, $\mu(\mathbf{x}')$, of the latent to label mapping that is

$$\mu(\mathbf{x}') = \mathbf{Z}^T K_Z^{-1} k_z(\mathbf{x}') \tag{8}$$

where $k_z(\mathbf{x}')$ is the vector with elements $k_z(\mathbf{x}', \mathbf{x}_i)$ for all the latent positions $\mathbf{x}_i$ in the model.

Fig. 2 shows a synthetic example where a two-class problem is learned using (a) PCA and our probabilistic discriminative model (b) *with* or (c) *without* back-constraints. Jointly learning the reconstruction and classification mapping splits the different classes in the latent space and results in better classification performance.

3 Transfer Learning with a Shared Latent Space

In our transfer learning scenario, we assume that if a latent low dimensional representation was found to be useful for a set of tasks, it will be useful for future related tasks. When we say that a latent representation is useful we mean that there is a function (i.e. a classifier) from the latent coordinates to the labels that performs accurate classification. It is easy to see that the joint learning framework described in the previous section can find useful representations, since it is learned discriminatively.

One of the advantages of using a low dimensional representation of the data for classification is that fewer parameters need to be estimated, thus reducing the number of examples required for training. Therefore, if we can find a useful low dimensional representation using previous tasks we can effectively reduce the number of examples needed for learning the new task. In this section we show how the proposed probabilistic discriminative model can be extended to the transfer learning scenario.

3.1 Jointly learning P related problems

Let $\mathbf{Y}^{(p)} = [\mathbf{y}_1^{(p)}, \cdots, \mathbf{y}_{N_p}^{(p)}]^T$ be the set of N_p observations associated to problem p, and $\mathbf{Y} = [\mathbf{Y}^{(1)}, \cdots, \mathbf{Y}^{(P)}]^T$ the set of all observations. Similarly let $\mathbf{X} = [\mathbf{X}^{(1)}, \cdots, \mathbf{X}^{(P)}]^T$ be the latent coordinates for all problems, and $\mathbf{Z}^{(p)} = [\mathbf{z}_1^{(p)}, \cdots, \mathbf{z}_{N_p}^{(p)}]^T$ be the labels for the p-th problem. Our model (Fig. 1 (c)) is a generative model of the data with a shared latent space across problems and a set of independent (given the latent variables) regressors modeling each classification task. Learning this model is equivalent to minimizing the negative log likelihood with respect to the latent coordinates, and the kernel hyperparameters of each individual mapping.

$$\begin{aligned} \mathcal{L}_{\mathcal{TL}} \;=\; & + \sum_{p=1}^{P} \left(\frac{S}{2} \ln |\mathbf{K}_Z^{(p)}| + \frac{1}{2} \mathrm{tr} \left(\mathbf{K}_Z^{(p)^{-1}} \mathbf{Z}^{(p)} \mathbf{Z}^{(p)^T} \right) + \sum_i \ln \gamma_i^{(p)} \right) \\ & \frac{D}{2} \ln |\mathbf{K}_Y| + \frac{1}{2} \mathrm{tr} \left(\mathbf{K}_Y^{-1} \mathbf{Y} \mathbf{Y}^T \right) + \sum_i \ln \beta_i + C_2 \end{aligned} \tag{9}$$

where C_2 is a constant and $\hat{\gamma}^{(p)}$ are the hyperparameters associated with the classification regressor of problem p. Note that now, there are $P * N_p + N_Y$ hyperparameters to estimate, where N_p is the number of hyperparameters to model a single latent to labels mapping, and N_Y is the number of hyperparameter of the mapping from the latent space to the observations. We call this model the *Multi-Task Latent Gaussian Process (Multi-Task LGP)*.

Given that we can find a low dimensional representation that is useful for multiple tasks, we turn our attention to the problem of using that representation in a future task.

3.2 Transfer learning with the joint model

In our transfer learning scenario we are interested in training a classifier for one particular task, which we call the target task, using a very small training set. We assume that in addition to the training set of the target task, we are given labeled training sets for related tasks. In our case we can informally say that a set of tasks are related if there exists a latent space (i.e. a representation) such that we can find good classifiers for all tasks on that space (i.e. using the shared representation).

The input to our transfer learning algorithm is a target training set $t_{target} = \{\mathbf{Y}^{(P+1)}, \mathbf{Z}^{(P+1)}\}$ and a collection of source training sets $C = \{t_1, \cdots, t_P\}$ from related problems, where $t_i = \{\mathbf{Y}^{(i)}, \mathbf{Z}^{(i)}\}$. Our method proceeds as follows. We first train a Multi-Task LGP using the training sets in C and learn a shared latent space (as described in section 3.1), and then project the target samples $\mathbf{Y}^{P+1}$ using the learned back-constraints to create the target latent coordinates[2]. We then train a GP regressor from the target latent coordinates to the target labels. We call our method *Discriminative Transfer (DT)*.

Of course, one can jointly train the set $C' = \{t_1, \cdots, t_P, t_{target}\}$ composed of the target and related problems. However, in practice, when the target problem has few examples compared to the problems in C, the results are very similar. The advantage of the two step optimization is that if the latent space has already been learned (from a potentially very large collection of related problems) training the target task becomes very fast. Similar optimization strategies have been previously applied in the literature [1].

Inference in our model is done as described in section 2.2; the back-constraints are learned from the P related problems, and used to estimate the latent positions of the target problem. The test labels are then obtained by thresholding the output of the latent-to-labels regressor for the target task.

4 Experimental Results

We conducted experiments on the first five digits of the USPS dataset from the UCI repository; we regard the binary detection of each digit as a separate task. Each of the tasks consists of detecting

[2]There is a single set of back-constraints for all the related problems that are learned jointly with the individual discriminative regressors using (9).

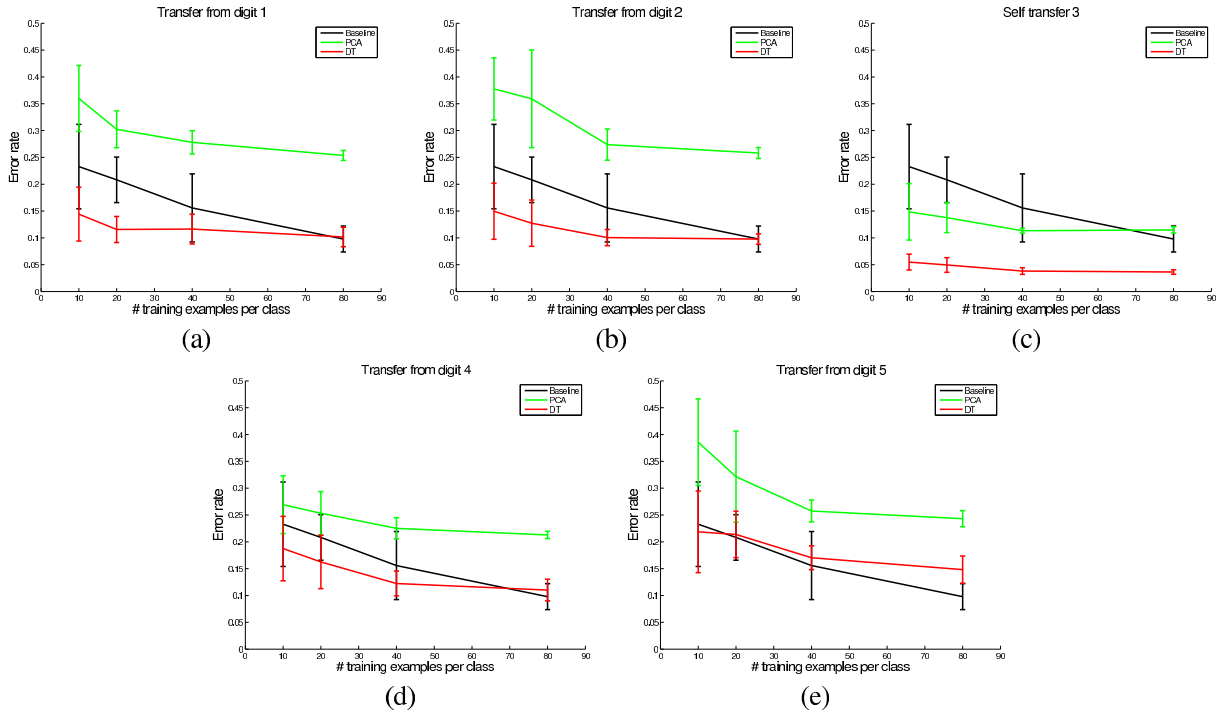


Figure 3: **Transfering the latent space** learned from a single source problem to a target problem. The target problem is to detect 3's from other digits, and the source problems are to similarly detect (a) 1's, (b) 2's, (c) 3's, (d) 4's, and (e) 5's. Each related problem is trained with 300 positive examples and 300 negative examples of the other four digits. Our algorithm (red) is compared against two baselines: a single-task probabilistic discriminative model that learns a latent space only from the target data (black) and the result of PCA-based semi-supervised learning with unlabeled data from the source problem (green). Note that the 5's problem might be less related to the 3's since the transfer learning does not help. In all the cases except the self-transfer (c), the PCA baseline underperforms the discriminative probabilistic model, showing the importance of discriminatively learning the latent space.

one digit from the other digits[3]. For every task we generate a labeled training set $t_i = \{\mathbf{Y}^{(i)}, \mathbf{Z}^{(i)}\}$ by picking 300 positive examples and 300 negative examples (sampled from the other four digits). We generate a testing set for each task in a similar manner.

We compare our approach to two baselines. The first ignores the related problems (i.e. information in C), and learns a single-task model using only the target data. The second is based on semi-supervised learning using the data in C but not the labels. More specifically, it learns a PCA space using data from the related problems and projects the target observations onto that space. It then uses a GP regressor to learn a mapping from the projected target samples to the target labels. For all experiments we used a two dimensional latent space; all optimizations were performed with conjugate gradient descent and run for a maximum of 100 iterations.

In the first set of experiments we transfer a shared representation learned from a single related problem, with 600 training examples (300 positive and 300 negative) for the related problem. We evaluate the performance of our algorithm and the baselines for different sizes of the training data for the target problem using 10 random partitions of the data. Fig. 3 (a–e) shows how the error rate decreases as more examples are used. In all cases Discriminative Transfer gives lower error rates than the PCA-based semi-supervised baseline, illustrating the importance of learning the latent space discriminatively. We also see that transferring from any task except from digit 5 increases performance relative to the first baseline, suggesting that digit 5 might not be as related. The *self-transfer*

[3]In particular we focus on detecting 3's from the other digits (i.e., 1's, 2's, 4's, 5's) since it is known to be one of the hardest problems.

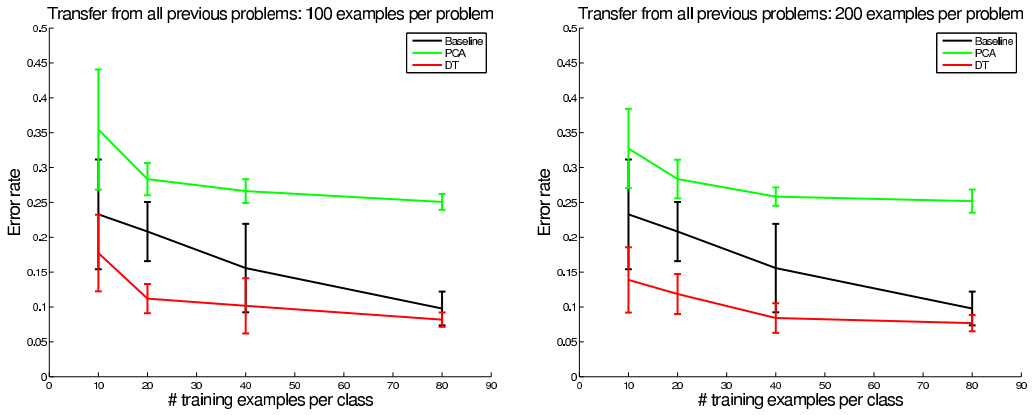


Figure 4: **Joint learning** of the latent space from multiple source problems and transfer to the target task. The source and target problems are as above. Results using Discriminative Transfer are shown in red and are compared against two baselines, PCA-based semi-supervised learning (green) and a single-task probabilistic discriminative model trained on the target problem (black). Transfer from other related (and less related) problems improves the performance with respect to learning with a single-task model, especially when the number of examples is small. PCa-based semi-supervised learning performs poorly in this case. Figure (a) shows results when using 100 positive examples and 100 negative examples for each related problem, and (b) shows results with 200 positive and 200 negative examples.

case gives an upperbound on the performance of transfer learning; it tells us what the performance would be if the problems were *fully* related (Fig. 3 (c)).

The previous experiment showed that transferring a shared latent space from a related problem can significantly reduce the number of training examples needed to learn the target problem. In practice however, we do not know what problem to choose for transfer because we do not know a priori which problems are related. What we need is a transfer learning algorithm that takes a set C containing mostly related problems and learns good shared latent spaces without being adversely affected by the presence of a few less-related problems. In our second set of experiments we test the robustness of our transfer learning algorithm in this more realistic scenario and transfer a shared representation from all previous problems (i.e., detecting 1's, 2's, 4's, 5's). The results in Fig. 4 show that Discriminative Transfer improves performance significantly compared to PCA and GP baselines. Our algorithm performs similarly using 200 (Fig. 4(a)) or 400 (Fig. 4(b)) examples for each related problem.

5 Conclusion

We have presented a new method for transfer learning based on shared non-linear latent spaces. Our method exploits joint optimization within a GP-LVM framework, and discovers latent spaces which are simultaneously effective at describing the observed data and solving several classification or regression tasks. When transferred to new tasks with relatively few training examples, learning can be faster and/or more accurate with this approach. Experiments on a digit recognition task demonstrated significantly improved performance when compared to baseline performance with the original feature representation or with a representation derived from a semi-supervised learning approach. As future work we plan to extend our method to a GPC formalism, investigate sparcification techniques to speed learning and inference, as well as explore the use of discriminative priors on the latent space [14] in the context of our model.

References

[1] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:817–8535, 2005.

[2] J. Baxter. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, 28, 1997.

[3] R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

[4] M. Kuss and C. E. Rasmussen. Assessing approximations for gaussian process classification. In *Neural Information Processing Systems*, pages 699 – 706. MIT Press, 2006.

[5] N. D. Lawrence. Gaussian Process Models for Visualisation of High Dimensional Data. In *Neural Information Processing Systems*. MIT Press, Cambridge, MA, 2004.

[6] N. D. Lawrence and A. J. Moore. Hierarchical Gaussian process latent variable models. In *International Conference in Machine Learning*, 2007.

[7] N. D. Lawrence and J. C. Platt. Learning to learn with the informative vector machine. In *International Conference in Machine Learning*, 2004.

[8] N. D. Lawrence and J. Quiñonero-Candela. Local distance preservation in the GP-LVM through back constraints. In *International Conference in Machine Learning*, volume 69, pages 96–103, Banff, Alberta, Canada, July 2006.

[9] T. P. Minka. *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, MIT, 2001.

[10] C. E. Rasmussen and C. K. Williams. *Gaussian Process for Machine Learning*. MIT Press, 2006.

[11] A. P. Shon, K. Grochow, A. Hertzmann, and R. Rao. Learning Shared Latent Structure for Image Synthesis and Robotic Imitation. In *Neural Information Processing Systems*. MIT Press, Cambridge, MA, 2006.

[12] S. Thrun. Is learning the n-th thing any easier than learning the first? In *Neural Information Processing Systems*, 1996.

[13] M. Tipping and C. Bishop. Probabilistic principal component anlaysis. *Journal of the Royal Statistical Society, B*, 61(3):611–622, 1999.

[14] R. Urtasun and T. Darrell. Discriminative Gaussian Process Latent Variable Models for Classification . In *International Conference in Machine Learning*, 2007.

[15] J. Wang, D. J. Fleet, and A. Hertzman. Gaussian Process dynamical models. In *Neural Information Processing Systems*. MIT Press, Cambridge, MA, 2005.

[16] C. K. I. Williams and D. Barber. Bayesian classification with Gaussian processes . *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342–1351, 1998.

[17] K. Yu, V. Tresp, and A. Schwaighofer. Learning Gaussian processes from multiple tasks. In *International Conference in Machine Learning*, pages 1012–1019, 2005.