Transfer Learning Using Task-Level Features with Application to Information Retrieval

Rong Yan
IBM T. J. Watson Research
Hawthorne, NY, USA
yanr@us.ibm.com

Jian Zhang
Purdue University
West Lafayette, IN, USA
jianzhan@stat.purdue.edu

Abstract

We propose a probabilistic transfer learning model that uses task-level features to control the task mixture selection in a hierarchical Bayesian model. These task-level features, although rarely used in existing approaches, can provide additional information to model complex task distributions and allow effective transfer to new tasks especially when only limited number of data are available. To estimate the model parameters, we develop an empirical Bayes method based on variational approximation techniques. Our experiments on information retrieval show that the proposed model achieves significantly better performance compared with other transfer learning methods.

1 Introduction

Recent years have seen a growing interest in developing algorithms to learn multiple related tasks and dynamically transfer existing models to new tasks. This is known as transfer learning [Thrun and Pratt, 1998], or multi-task learning [Caruana, 1997]. It has been empirically as well as theoretically shown that transfer learning can significantly improve the learning performance especially when only a small number of data are available for new tasks. Transfer learning has many practical applications, and one such example is information retrieval. In more details, information retrieval can be cast into binary classification that considers the relevant query-document pairs as positive data and irrelevant pairs as negative data. By viewing each query as a separate task, we can reformulate the retrieval task into a transfer learning problem, which aims to predict the relevance of each document for a new query based on the manual relevance judgment from the training queries.

Many transfer learning methods assume that the model parameters are generated from a uni-modal distribution (such as normal) for all given tasks. However, such an assumption is very likely to be violated in practice. As a matter of fact, tasks in many domains tend to be grouped into a number of clusters, and different tasks can be associated with different clusters. In such cases, it is of great importance for the transfer learning algorithms to effectively capture task-cluster associations and route a new task to the correct cluster. Although a simple mixture model instead of a uni-modal distribution can capture the above multi-cluster situation, the goal is still difficult to achieve when each task only possesses a limited number of training examples. Even worse, a new task can appear without any training examples at all. Unfortunately, transferring to a new task with few examples is typical in real applications such as information retrieval.

To approach this issue, we consider incorporating into transfer learning a set of features which can directly represent the properties of the task itself, called "task-level features", in additional to the commonly-used data features generated from training examples. Task-level features can be extracted in a flexible way. In information retrieval, task features can be defined as the properties of user queries, such as whether the query contains person names or not. Similarly, task features can also be generated from user profiles in collaborative filtering task, and school-level information for predicting student exam scores [Bakker and Heskes, 2003]. Intuitively, these task features provide helpful evidence to pinpoint which cluster a given task belongs to, especially when only a limited number of data examples are available for individual tasks. But the values of task features have not been fully explored in the previous work of transfer learning.

In this paper, we propose a probabilistic transfer learning(TL) model by introducing task-level features to a hierarchical Bayesian model for classification. We call it the TL-TF model in the rest of the paper. This model assumes the model parameters for different tasks are generated from a linear combination of basic logistic regression classifiers together with a small set of hidden sources. The distribution of hidden sources of each task is controlled by its task features, while the combination weights and the model parameters in basic classifiers are learned from the data examples. To estimate the parameters of TL-TF, we also develop an empirical Bayes method based on variational approximation techniques. When transferring to a new task, the weights of task mixtures can be directly derived from its task features, and thus the classification outputs can be computed without using any additional training data from the new task. As the first attempt to apply transfer learning with task features to information retrieval, our experiments show that the proposed model is much more effective than other TL methods that either discard the task features, or treat task features as another set of data features.

2 Related Work

In the literature, transfer learning has also been known as "multi-task learning" and "learning to learn". Many methods have been proposed for transfer learning and multi-task learning, such as neural networks [Caruana, 1997],transformation methods [Breiman and Friedman, 1997], regularization learning methods [Evgeniou *et al.*, 2005; Ando and Zhang, 2004], hierarchical Bayesian models [Heskes, 2000; Yu *et al.*, 2005; Zhang *et al.*, 2005] and etc.

In particular, the proposed model is closely related to the work based on hierarchical Bayesian models, which provide a flexible yet compact representation of the structure in the data space. Thus, they allow models to fit existing data well and generalize well on unseen data. For example, [Heskes, 2000] presented a model for multi-task learning by assuming that response variables of each task follow a normal distribution. Empirical Bayes techniques are used to learn the model hyper-parameters. In [Yu *et al.*, 2005], Gaussian processes are applied to model multiple related tasks and a single Gaussian component is used to capture the mean and covariance of all related tasks. [Zhang *et al.*, 2005] further proposed a latent variable model for multi-task learning which assumes that tasks are sparse linear combination of basic classifiers. A more recent work [Xue *et al.*, 2007] proposed a Dirichlet process based hierarchical Bayesian model for learning multiple related classifiers. However, none of above models take the task-level features into account in the learning process.

By utilizing task features in a softmax gating function, [Bakker and Heskes, 2003] augmented the Bayesian neural network model to handle multi-task learning with real-valued outputs and additive noise. This approach has shown to be effective in multiple data collections. However, this study was mainly focusing on regression in a non-transfer setting. More recently, [Bonilla *et al.*, 2007] presented a kernel multi-task regression method using Gaussian process with task features. In contrast, the proposed approach is a hierarchical Bayesian model with focus on classification problems under the transfer learning setting, which is more typical for applications such as information retrieval. Moreover, to our best knowledge, our work is the first attempt to apply transfer learning with task-level features to the information retrieval domain.

3 Transfer Learning with Task-Level Features

In this section, we describe the details of the proposed TL-TF model by introducing mixture modeling and task-level features to hierarchical Bayesian models. First, let us assume there are K classification tasks. Each task is associated with a training data set $\mathcal{D}^{(k)} = \{\mathbf{X}^{(k)}, \mathbf{y}^{(k)}\}$, where the training data are $\mathbf{X}^{(k)} = [\mathbf{x}_1^{(k)}, \ldots, \mathbf{x}_{n_k}^{(k)}]^T \in \mathbb{R}^{n_k \times F}$ and the labels are $\mathbf{y}^{(k)} = [y_1^{(k)}, \ldots, y_{n_k}^{(k)}]^T$. $\mathbf{y}^{(k)} \in \{0,1\}^{n_k \times 1}$ for the classification tasks and $\mathbf{y}^{(k)} \in \mathbb{R}^{n_k \times 1}$ for the regression tasks. Furthermore, for each task we are able to obtain a set of task-level features $\mathbf{t}^{(k)} \in \mathbb{R}^{P \times 1}$ that describe the task properties. These features can be automatically extracted from the task description without knowing the data examples, such as the number of persons mentioned in queries in information retrieval and the age of users in collaborative filtering.

One building block for transfer learning is to define the learning models for each individual task. In this paper, we consider using the parametric model $f^{(k)}(\mathbf{x}) = f^{(k)}(\mathbf{x}|\boldsymbol{\theta}^{(k)})$ to generate the data labels for each task, where the parameter $\boldsymbol{\theta}^{(k)}$ is used to index the prediction function $f^{(k)}$.[1] Given the parameter $\boldsymbol{\theta}^{(k)}$, we can derive the following likelihood models for classification,

$$y_i^{(k)} \sim \mathsf{Bernoulli}(g(\langle \boldsymbol{\theta}^{(k)}, \mathbf{x}_i^{(k)} \rangle)) \tag{1}$$

where $g(t) = (1 + \exp(-t))^{-1}$ is the logistic function and $\langle \mathbf{x}, \mathbf{y} \rangle$ is used to denote the inner product between $\mathbf{x}$ and $\mathbf{y}$. This model becomes logistic regression if the model parameters $\boldsymbol{\theta}^{(k)}$ are estimated independently on each single task.

In order to learn multiple related tasks more effectively, we can transform individual task learning process into a joint learning problem, which explains the relatedness of multiple tasks through some hidden sources $\mathbf{s}^{(k)}$. To handle the multi-cluster task distribution, we assume the hidden source variable $\mathbf{s}^{(k)}$ is sampled from a multinomial distribution and use it to indicate which mixture component the task belongs to. Given that a set of task-level features $\mathbf{t}^{(k)}$ are available for each task, we further assume the distribution parameters of s_k are determined by its task features. Then a generative model can be constructed for $\boldsymbol{\theta}^{(k)}$'s in such a way that not only task dependency structure can be captured and jointly modeled, but also the information contained in $\mathbf{t}^{(k)}$ can be effectively utilized. Formally, we can have the following hierarchical Bayesian model for generating $\boldsymbol{\theta}^{(k)}$'s:

$$\begin{aligned} \boldsymbol{\theta}^{(k)} &= \boldsymbol{\Lambda}\mathbf{s}^{(k)} + \boldsymbol{e}^{(k)} \\ \mathbf{s}^{(k)} &\sim \mathsf{Multinomial}\,(p_1, \ldots, p_H) \\ \boldsymbol{e}^{(k)} &\sim \mathsf{Normal}\,(0, \boldsymbol{\Psi}) \\ p_h &= \frac{\exp(\boldsymbol{\gamma}_h^T \mathbf{t}^{(k)})}{\sum_{h'} \exp(\boldsymbol{\gamma}_{h'}^T \mathbf{t}^{(k)})} \end{aligned} \tag{2}$$

where $\boldsymbol{\theta}^{(k)} \in \mathbb{R}^{F \times 1}$ are the model parameters for the k^{th} task, $\mathbf{t}^{(k)} \in \mathbb{R}^{P \times 1}$ are the task features, $\mathbf{s}^{(k)} \in \mathbb{R}^{H \times 1}$ are the hidden source vector[2] where H is the number of task clusters and $\boldsymbol{\gamma}_h$ denotes the distribution parameters for the softmax function p_h, $\boldsymbol{\Lambda} \in \mathbb{R}^{F \times H}$ is the linear transformation matrix and $\boldsymbol{e}^{(k)}$ is a multi-variate Gaussian noise vector with a covariance matrix $\boldsymbol{\Psi} \in \mathbb{R}^{F \times F}$. To further simplify derivation, we use $\boldsymbol{\Gamma} = [\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_H] \in \mathbb{R}^{P \times H}$ to denote the parameter matrix of the multi-class logistic regression where $\boldsymbol{\gamma}_h$ is its h-th column, and similar notation is used for $\boldsymbol{\Lambda} = [\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_H]$ where $\boldsymbol{\lambda}_h$ is the h-th column of $\boldsymbol{\Lambda}$. Also note that due to the properties of multi-class logistic regression [Hastie *et al.*, 2001] we can fix $\boldsymbol{\gamma}_1$ to be the all zero vector $\mathbf{0}$.

The probabilistic model is complete by combining Eqn (1) and (2). From the model definition, we can find that: 1. $\boldsymbol{\theta}^{(k)}$

[1] Note that it is not necessary to require all $f^{(k)}$'s belonging to the same parametric family, however, parameters $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(K)} \in \Theta$ are assumed to belong to the same metric space.

[2] The vector $\mathbf{s}^{(k)}$ takes the forms of $[0, \ldots, 0, 1, 0, \ldots, 0]$ where only one element is 1 and the rest are 0's. We also use $z^{(k)} \in \{1, \ldots, H\}$ instead of $\mathbf{s}^{(k)}$ to denote the 1's index for simplicity.

is assumed to be normally distributed with the same covariance $\mathbf{\Psi}$. Its mean is a linear combination of columns of $\mathbf{\Lambda}$ shared by all K tasks; 2. The combination weights for $\mathbf{\Lambda}$ are determined by the latent variable $\mathbf{s}^{(k)}$ sampled from a multinomial distribution; 3. $\mathbf{s}^{(k)}$ is controlled by the task-level features $\mathbf{t}^{(k)}$ through a multi-class logistic regression model, where in some sense $\mathbf{t}^{(k)}$ serves as a gating function to adapt the distribution of hidden sources task by task.

Our model has connections to several previous approaches. For instance, in the case when $H = 1$ and without considering the task-level features, the proposed model is closely related to the multi-task linear models proposed in [Yu *et al.*, 2005]. Furthermore, when $H > 1$ it is closely related to the latent independent component analysis (LICA) model proposed in [Zhang *et al.*, 2005] which assumes the latent variable $\mathbf{s}^{(k)}$ to follow a sparse prior distribution. However, in contrast to existing approaches, the proposed model distinguishes itself from the fact that task-level features $\mathbf{t}^{(k)}$ are naturally incorporated into the generation process of latent variables $\mathbf{s}^{(k)}$. This allows us to model the task-mixture association more accurately and predict the mixture assignment for a new task even when few data examples are provided.

As a next step, we need to transfer the parameters of the learned model to a new task (e.g., a new query or a new user) with a limited number of training data or even no training data. We are interested in investigating whether the learning of a new task can benefit from generalizing the previous task parameters and whether the task features can be helpful to provide more accurate predictions. In this case, it is key to develop a generative model, e.g. we have to make explicit assumptions about how tasks are related. We can observe from our generative model that given the learned parameters $\mathbf{\Gamma}$, $\mathbf{\Lambda}$ and $\mathbf{\Psi}$ from previous K tasks, we can naturally extend the generation process for the $(K+1)$-th task to be

$$\begin{aligned}\boldsymbol{\theta}^{(K+1)} &\sim \mathsf{Normal}(\mathbf{\Lambda}\mathbf{s}^{(K+1)}, \mathbf{\Psi}) \\ \mathbf{s}^{(K+1)} &\sim \mathsf{Multinomial}(p_1, \ldots, p_H), \\ p_h &= \frac{\exp(\boldsymbol{\gamma}_h^T \mathbf{t}^{(k+1)})}{\sum_{h'} \exp(\boldsymbol{\gamma}_{h'}^T \mathbf{t}^{(k+1)})},\end{aligned}$$

where $\mathbf{t}^{(k+1)}$ is the task feature for the new task. Finally, for a given input data vector $\mathbf{x}$, its prediction is given by

$$p(y|\mathbf{x}) = \sum_{h=1}^{H} p_h \int p(\boldsymbol{\theta}^{(K+1)}|\boldsymbol{\lambda}_h, \mathbf{\Psi}) p(y|\mathbf{x}, \boldsymbol{\theta}^{(K+1)}) d\boldsymbol{\theta}^{(K+1)}.$$

If we want to reduce the computational complexity in the prediction step, an alternative is to use the MAP estimation of $\boldsymbol{\theta}^{(K+1)}$ to avoid the computation of the integral, where the prediction function can be rewritten as,

$$p(y|\mathbf{x}) = \sum_{h=1}^{H} p_h p(y|\mathbf{x}, \boldsymbol{\lambda}_h).$$

This formula can be efficiently computed and thus we adopt it in our following experiments.

4 Learning and Inference

We present the learning and inference algorithms for TL-TF in this section. Given the model definition of TL-TF, we need to estimate the parameters $\mathbf{\Lambda}$, $\mathbf{\Gamma}$ and $\mathbf{\Psi}$. Here we take the empirical Bayes approach by integrating out the random variables $\mathbf{s}^{(k)}$'s and $\boldsymbol{\theta}^{(k)}$'s. Thus, the log-likelihood of the parameters $\Omega = \{\mathbf{\Lambda}, \mathbf{\Gamma}, \mathbf{\Psi}\}$ for observed data $\{\mathbf{X}^{(k)}, \mathbf{y}^{(k)}, \mathbf{t}^{(k)}\}_{k=1}^{K}$ is

$$\begin{aligned}&\log p\left(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(K)} \mid \Omega, \mathbf{X}^{(1)}, \mathbf{t}^{(1)}, \ldots, \mathbf{X}^{(K)}, \mathbf{t}^{(K)}\right) \\ =& \sum_{k=1}^{K} \log \sum_{\mathbf{s}} p(\mathbf{s}^{(k)}|\mathbf{\Gamma}, \mathbf{t}^{(k)}) \\ &\times \int p(\boldsymbol{\theta}^{(k)}|\mathbf{\Lambda}, \mathbf{s}^{(k)}, \mathbf{\Psi}) \prod_{i=1}^{n_k} p(y_i^{(k)}|\boldsymbol{\theta}^{(k)}, \mathbf{x}_i^{(k)}) d\boldsymbol{\theta}^{(k)},\end{aligned}$$

where $p(\mathbf{s}^{(k)}|\mathbf{\Gamma}, \mathbf{t}^{(k)})$ is the multinomial distribution of the hidden sources, $p(\boldsymbol{\theta}^{(k)}|\mathbf{\Lambda}, \mathbf{s}^{(k)})$ is a normal distribution with mean $\boldsymbol{\lambda}_{z^{(k)}}$ and covariance matrix $\mathbf{\Psi}$, and $p(y_i^{(k)}|\boldsymbol{\theta}^{(k)}, \mathbf{x}_i^{(k)})$ is the likelihood function of classification in Eqn (1).

Such an estimation problem can be solved by the EM algorithm, of which the detailed derivations are given as follows. To be more specific, the goal of learning is to estimate the parameters Ω by maximizing the log-likelihood over all K tasks. Since the log-likelihood function involves two set of hidden variables, i.e., $\mathbf{s}^{(k)}$'s and $\boldsymbol{\theta}^{(k)}$'s, EM is applied to iteratively solve a series of simpler problems.

E-step: Given the parameters Ω all tasks are decoupled, the E-step can be conducted for each task separately. Thus we only need to consider one task per time and we can omit the superscript (k) for simplicity. But because it is generally intractable to do an exact inference for classification problems, we decided to apply variational methods as one type of approximate inference techniques to optimize the objective function. The basic idea of variational methods is to use a tractable family of distributions $q(\boldsymbol{\theta}, \mathbf{s})$ to approximate the true posterior distribution. Specifically we assume an auxiliary distribution $q(\boldsymbol{\theta}, \mathbf{s}) = q_1(\mathbf{s})q_2(\boldsymbol{\theta})$, e.g. the mean field approximation, as a surrogate to approximate the true posterior distribution $p(\boldsymbol{\theta}, \mathbf{s}|\Omega, \mathbf{X}, \mathbf{t}, \mathbf{y})$.

Furthermore, we assume that $q_1(\mathbf{s}) = q_1(\mathbf{s}|\mathbf{\Phi})$ has the form of multinomial distribution with parameters $\boldsymbol{\phi} = [\phi_1, \ldots, \phi_H]^T$ such that $\phi_h \geq 0$ and $\sum_{h=1}^{H} \phi_h = 1$. $q_2(\boldsymbol{\theta}) = q_2(\boldsymbol{\theta}|\mathbf{m}, \mathbf{V})$ has the form of a multivariate normal with mean $\mathbf{m}$ and covariance matrix $\mathbf{V}$. We need to find the best set of variational parameters $\boldsymbol{\phi}$, $\mathbf{m}$ and $\mathbf{V}$ such that the following lower bound is maximized, which is equivalent to minimize the KL divergence between $q_1(\mathbf{s})q_2(\boldsymbol{\theta})$ and $p(\boldsymbol{\theta}, \mathbf{s}|\Omega, \mathbf{X}, \mathbf{t}, \mathbf{y})$:

$$\begin{aligned}&\log p(\mathbf{y}|\Omega, \mathbf{X}, \mathbf{t}) \\ \geq& \sum_{\mathbf{s}} q_1(\mathbf{s}|\boldsymbol{\phi}) \mathbb{E}_{q_2}[\log p(\mathbf{y}, \boldsymbol{\theta}, \mathbf{s}|\Omega, \mathbf{X}, \mathbf{t})] + H(\mathbf{s}) + H(\boldsymbol{\theta}) \\ =& \sum_{h=1}^{H} \phi_h \left\{\mathbb{E}[\log p(z = h|\Omega, \mathbf{t}) + \mathbb{E}[\log p(\boldsymbol{\theta}|\boldsymbol{\lambda}_h, \mathbf{\Psi})]\right\} \\ &+ \mathbb{E}[\log p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{X})] + H(\mathbf{s}) + H(\boldsymbol{\theta})\end{aligned}$$

where $H(\boldsymbol{\theta}) = -\int q_2(\boldsymbol{\theta}) \log q_2(\boldsymbol{\theta}) d\boldsymbol{\theta}$ is the entropy of $\boldsymbol{\theta}$, $H(\mathbf{s}) = -\sum \phi_h \log \phi_h$ is the entropy of $\mathbf{s}$, and the expected values of the first two terms on the right hand side are

$$\begin{aligned}\mathbb{E}\left[p(\boldsymbol{\theta}|\boldsymbol{\lambda}_{\cdot h}, \boldsymbol{\Psi})\right] &= -\frac{1}{2}\log|2\pi\boldsymbol{\Psi}| - \frac{1}{2}\mathsf{Tr}\left[\boldsymbol{\Psi}^{-1}\mathbf{V}\right] \\ &\quad -\frac{1}{2}(\mathbf{m}-\boldsymbol{\lambda}_h)^T\boldsymbol{\Psi}^{-1}(\mathbf{m}-\boldsymbol{\lambda}_h) \\ \mathbb{E}\left[\log p(\mathbf{s}=h|\Omega,\mathbf{t})\right] &= \log\frac{\exp(\boldsymbol{\gamma}_h^T\mathbf{t})}{\sum_{h'}\exp(\boldsymbol{\gamma}_{h'}^T\mathbf{t})}.\end{aligned}$$

However, the derivation is not complete yet because we cannot derive a closed-form representation for the term $\mathbb{E}[\log p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{X})]$, where $p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{X}) = \prod_i p(y_i|\boldsymbol{\theta}, \mathbf{x}_i)$ is a product of logistic likelihood functions. So we resort to another variational technique proposed in [Jaakkola and Jordan, 1997] to compute its lower bound as a function of $\mathbf{m}$ and $\mathbf{V}$ by introducing a new variational parameters ξ_i for the i^{th} example for the given task:

$$\begin{aligned}\mathbb{E}[\log p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{X})] &\geq \sum_{i=1}^{n}\left(\log g(\xi_i) + \frac{y_i\mathbf{m}^T\mathbf{x}_i - \xi_i}{2}\right. \\ &\quad \left. + h(\xi_i)\left(\mathbf{x}_i^T(\mathbf{V}+\mathbf{m}\mathbf{m}^T)\mathbf{x}_i - \xi_i^2\right)\right)\end{aligned}$$

where $h(t) = (1/2 - g(t))/(2t)$, $g(t)$ is the logistic function and n is the number of data for the task. By substituting the additional lower bound back to Eqn(3), we are able to optimize the lower bound of the log-likelihood function with respect to $\mathbf{V}$, $\mathbf{m}$, $\boldsymbol{\phi}$ to obtain the following E-step:

$$\xi_i = [\mathbf{x}_i^T(\mathbf{V}+\mathbf{m}\mathbf{m}^T)\mathbf{x}_i]^{1/2} \quad (3)$$

$$\mathbf{V} = \left(\boldsymbol{\Psi}^{-1} - 2\sum_{i=1}^{n} g(\xi_i)\mathbf{x}_i\mathbf{x}_i^T\right)^{-1} \quad (4)$$

$$\mathbf{m} = \mathbf{V}\left(\frac{1}{2}\sum_{i=1}^{n} y_i\mathbf{x}_i + \boldsymbol{\Psi}^{-1}\sum_{h=1}^{H}\phi_h\boldsymbol{\lambda}_h\right) \quad (5)$$

$$\phi_h \propto \exp\left(\boldsymbol{\gamma}_h^T\mathbf{t} - \frac{1}{2}(\mathbf{m}-\boldsymbol{\lambda}_h)^T\boldsymbol{\Psi}^{-1}(\mathbf{m}-\boldsymbol{\lambda}_h)\right) \quad (6)$$

These fixed point equations should be repeated over ξ_i's, $\mathbf{m}$, $\mathbf{V}$ and ϕ_h's until the lower bound is maximized. Upon convergence, we can use the resulting $q_1(\mathbf{s}|\boldsymbol{\phi})q_2(\boldsymbol{\theta}|\mathbf{m}, \mathbf{V})$ as a surrogate to the true posterior probability $p(\mathbf{s}, \boldsymbol{\theta}|\Omega, \mathbf{X}, \mathbf{t}, \mathbf{y})$.

M-step: Given the sufficient statistics obtained in the E-step, the M-step can be derived similarly by optimizing the lower bound of log-likelihood with respect to the model parameters, i.e., $\boldsymbol{\Gamma}, \boldsymbol{\Lambda}, \boldsymbol{\Psi}$:

$$\begin{aligned}\hat{\boldsymbol{\Gamma}} &= \arg\min_{\boldsymbol{\Gamma}}\left\{\sum_{k=1}^{K}\sum_{h=1}^{H}\phi_h^{(k)}\log\left[\frac{\exp\left(\boldsymbol{\gamma}_h^T\mathbf{t}^{(k)}\right)}{\sum_{h'}\exp\left(\boldsymbol{\gamma}_{h'}^T\mathbf{t}^{(k)}\right)}\right]\right\} \\ \hat{\boldsymbol{\Lambda}} &= \left[\frac{\sum_{k=1}^{K}\phi_1^{(k)}\mathbf{m}^{(k)}}{\sum_{k=1}^{K}\phi_1^{(k)}}, \ldots, \frac{\sum_{k=1}^{K}\phi_H^{(k)}\mathbf{m}^{(k)}}{\sum_{k=1}^{K}\phi_H^{(k)}}\right] \quad (7) \\ \hat{\boldsymbol{\Psi}} &= \frac{1}{K}\sum_{k=1}^{K}\left(\mathbf{V}^{(k)} + \sum_{h=1}^{H}\phi_h^{(k)}(\mathbf{m}^{(k)}-\boldsymbol{\lambda}_h)(\mathbf{m}^{(k)}-\boldsymbol{\lambda}_h)^T\right)\end{aligned}$$

In case we want to reduce the number of parameters we can assume that $\boldsymbol{\Psi}$ is diagonal with isotropic variance, e.g. $\boldsymbol{\Psi} = \tau^2\mathbf{I}$, and we have $\hat{\tau}^2 = \mathsf{Tr}(\hat{\boldsymbol{\Psi}})/F$. The EM learning process is summarized in Algorithm 1.

Algorithm 1 Empirical Bayes method for classification tasks

1. Initialize parameters $\boldsymbol{\Lambda}$, $\boldsymbol{\Gamma}$ and $\boldsymbol{\Psi}$.
2. **E-step:** For the k-th task ($k = 1, \ldots, K$):
   (a) update $\xi_i^{(k)}$ ($i = 1, \ldots, n_k$) based on Eqn(3)
   (b) update $\mathbf{V}^{(k)}$ and $\mathbf{m}^{(k)}$ based on Eqn(4) and Eqn(5)
   (c) update $\boldsymbol{\phi}^{(k)}$ based on Eqn(6)
   (d) continue (a)-(c) until convergence
3. **M-step:** Update parameters according to Eqn(7)
4. Continue steps 2-3 until convergence.

5 Experiments: Information Retrieval

In this section, we examine the effectiveness of the proposed model for an information retrieval task on large-scale video collections. We compared the proposed TL-TF model with several baseline models, including a single task learning (STL) model by directly estimating the likelihood in Eqn (1), a transfer learning model with a single cluster (TL-SC) and a TL model with multiple clusters that does not utilize any task features(TL-MC). To compare in a fair manner, we also evaluate another multi-cluster TL model that combines task features with the set of data features without using a task-dependent gating function in Eqn(2) (TL-Comb).

5.1 Experimental Setting

We evaluated our learning algorithms using the video collections officially provided by the TREC video retrieval evaluation(TRECVID) 2002-2005 [Smeaton and Over, 2003]. Video shots are used as the retrieval unit. For each query, average precision is adopted as a measure of retrieval effectiveness. In more details, let R be the number of true relevant documents in a set of size S. At any given index j let R_j be the number of relevant documents in the top j documents. Let $I_j = 1$ if the j^{th} document is relevant and 0 otherwise. The average precision (AP) is then defined as $\frac{1}{R}\sum_{j=1}^{S}\frac{R_j}{j} * I_j$. Mean average precision (Mean-AP) is the mean of average precision for all queries in the collection.

We split the four video collections, i.e., TREC'02-'05, into a development set to learn model parameters, and a search set to evaluate retrieval performance. For each search set, NIST officially provided 25 query topics, including both text description and image/video examples, together with their relevance judgment. On the other hand, the four development collections are combined into a single training corpus. Several human annotators collaboratively defined 88 query topics – which are different from the testing queries – and collected

Data Set	t02	t03	t04	t05	dev
Query Num	25	25	24	24	88
Doc Num	24263	75850	48818	77979	124097

Table 1: Labels of the video collections and statistics. $t**$ indicate the TRECVID testing sets, where the embedded number is the year, and dev is the development set.

Data	Method	Mean-AP	P30	P100	Person	SObj	GObj	Sport	Other
t02	STL	0.114(+0%)	0.135	0.081	0.199	0.205	0.074	0.007	0.019
	TL-SC(H=1)	0.119(+4%)	0.123	0.081	0.290	0.183	0.062	0.017	0.018
	TL-MC(H=6)	0.123(+8%)	0.132	0.083	0.293	0.206	0.059	0.009	0.020
	TL-Comb(H=6)	0.124(+9%)	0.132	0.084	0.293	0.206	0.059	0.009	0.021
	TL-TF(H=6)	0.136(+19%)	0.138	0.082	0.387	0.208	0.049	0.006	0.026
t03	STL	0.150(+0%)	0.212	0.136	0.251	0.293	0.095	0.101	0.012
	TL-SC(H=1)	0.177(+18%)	0.224	0.134	0.366	0.314	0.103	0.070	0.012
	TL-MC(H=6)*	0.192(+28%)	0.228	0.140	0.428	0.350	0.081	0.102	0.013
	TL-Comb(H=6)*	0.190(+27%)	0.227	0.139	0.429	0.350	0.080	0.100	0.013
	TL-TF(H=6)*	0.203(+35%)	0.243	0.136	0.460	0.345	0.096	0.109	0.014
t04	STL	0.079(+0%)	0.177	0.116	0.144	0.063	0.034	0.108	0.051
	TL-SC(H=1)	0.089(+12%)	0.186	0.114	0.178	0.067	0.037	0.085	0.061
	TL-MC(H=6)	0.093(+17%)	0.185	0.110	0.185	0.036	0.038	0.100	0.035
	TL-Comb(H=6)	0.094(+17%)	0.186	0.111	0.186	0.037	0.037	0.099	0.035
	TL-TF(H=6)*	0.109(+39%)	0.190	0.123	0.252	0.069	0.045	0.111	0.047
t05	STL	0.095(+0%)	0.238	0.205	0.164	0.029	0.090	0.271	0.017
	TL-SC(H=1)	0.095(+0%)	0.242	0.198	0.159	0.021	0.091	0.200	0.019
	TL-MC(H=6)	0.097(+2%)	0.240	0.195	0.139	0.021	0.075	0.293	0.016
	TL-Comb(H=6)	0.099(+4%)	0.242	0.194	0.141	0.020	0.077	0.294	0.016
	TL-TF(H=6)	0.118(+21%)	0.243	0.207	0.180	0.047	0.072	0.329	0.015

Table 2: Comparison of retrieval performance between STL, TL-SC, TL-MC, TL-Comb and TL-TF on multiple testing sets. All parameters are learned from the development set with the 88 training queries. * means statistical significance over STL with *p-value* < 0.01 (sign tests).

their relevance judgment on the development set. Table 1 lists the labels of video collections and their query/document statistics.

As building blocks for information retrieval, we generated a number of ranking features on each video document, including 14 high-level semantic features learned from development data (face, anchor, commercial, studio, graphics, weather, sports, outdoor, person, crowd, road, car, building, motion), and 5 uni-modal retrieval experts (text retrieval, face recognition, image-based retrieval based on color, texture and edge histograms) [Yan *et al.*, 2004]. For the transfer learning algorithms, the covariance matrix $\mathbf{\Psi}$ is initialized to an identity matrix. The parameters of $\boldsymbol{\gamma}_h$ and $\boldsymbol{\lambda}_h$ are initialized to random values uniformly sampled from $[0, 1]$. The EM algorithm stops when the relative change of the log-likelihood is less than 10^{-5}.

We also designed the following 10 binary task features for the TL-TF model, which indicate if the query topics contain (1) specific person names; (2) specific object names; (3) more than two noun phrases; (4) words related to people/crowd; (5) words related to sports; (6) words related to vehicle; (7) words related to motion; (8) similar image examples; (9) image examples with faces; (10) if the text retrieval module finds more than 100 documents. All these task features can be automatically detected from the query description through manually defined rules plus natural language processing and image processing techniques [Yan *et al.*, 2004].

5.2 Retrieval Results

We present the information retrieval results on all the testing sets using the proposed models and parameters learned from the development collection. The number of task clusters can be estimated by optimizing the regularized log-likelihood with an additional BIC term. Based on development data, we estimated that the optimal number of task mixtures is 6.

Table 2 provided a detailed comparison between TL-TF with 6 query clusters (TL-TF, H=6) and several baseline algorithms including single task learning using logistic regression (STL), single-cluster TL (TL-SC, H=1), multi-cluster TL with 6 clusters(TL-MC, H=6), and TL-Comb with 6 clusters (TL-Comb, H=6)[3]. All the results are reported in terms of the mean average precisions (Mean-APs) up to 1000 documents, precision at top 30 and 100 documents. We also grouped the queries in each collection and reported their Mean-APs in five different categories, i.e., named person, special object, general object, sports and general queries.

We can observe from the results that TL-MC is usually superior to the STL and TL-SC algorithms, because user queries are typically organized into a limited number of clusters in the query feature space. However without task features, TL-MC needs to assume all new queries share the same parameter distribution, which limits the flexibility of transfer learning to produce more accurate results. TL-Comb, which incorporates task features as additional data features, offers slight gains over TL-MC, but the improvement is limited because it cannot dynamically change the clustering assignment when the queries vary. In contrast to TL-MC, TL-TF is able to leverage more powers from TL by routing new queries to the correct task clusters. On average, it brings a roughly 4% absolute improvement (or 30% relative improvement) over STL in terms of mean average precision. As it results, the difference between TL-TF and STL becomes statistically significant in two out of four collections. By comparing Mean-APs with respect to each query type, we find that TL-TF benefits most from the Person and Special Object type queries, as well as the Sports type in $t05$. This is because these query types have

[3] All our baseline results are among the top performance in each year's TRECVID evaluation

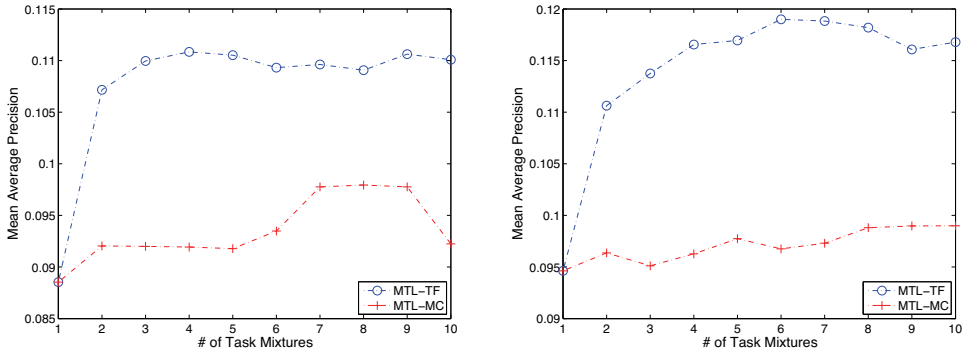

Figure 1: Comparison of TL-TF and TL-MC vs. the number of task clusters on two datasets. Left: t04, Right: t05

their information needs clearly defined and thus they are able to be improved by making better use of the training data.

To further analyze the performance of the proposed approaches, Figure 1 compares the TL-TF and TL-MC learning curves on two testing sets with the number of query clusters growing from 1 to 10. As we can see, TL-TF produces noticeably better results than TL-MC when the task clusters is larger than 2. The mean average precision of TL-TF increase dramatically using more task clusters especially when the cluster number is under four. For example, in $t05$ the mean average precision is boosted from 9.5% with one cluster to 12% with four clusters. This clearly shows the benefits of using task features to model complex task distributions compared with its non-task-feature counterpart.

6 Conclusion

In this paper, we propose a probabilistic transfer learning model called TL-TF, of which the basic idea is to use task-level features to control the task mixture selection in a hierarchical Bayesian model. In this model, each task predictor is modeled as a linear combination of a set of basic classifiers, and the task relatedness is explained by some hidden sources. We use task features to determine the distribution parameters of these hidden sources via a multi-class logistic regression model. As a result, each task predictor is able to choose the specific task mixture component more accurately based on the characteristics described by task features. To estimate the parameters of TL-TF, we also develop an empirical Bayes method based on variational approximation techniques. Experimental results on several information retrieval collections show that the proposed model is much more effective than the other transfer learning methods that either do not utilize task features, or simply treat them as data features.

References

[Ando and Zhang, 2004] R. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Technical Report RC23462, IBM Research*, 45, 2004.

[Bakker and Heskes, 2003] B. Bakker and T. Heskes. Task clustering and gating for bayesian multitask learning. *J. Mach. Learn. Res.*, 4:83–99, 2003.

[Bonilla *et al.*, 2007] E. V. Bonilla, F. V. Agakov, and C. K. I. Williams. Kernel multi-task learning using task-specific features. In *Proceedings of the 11th AISTATS*, 2007.

[Breiman and Friedman, 1997] L. Breiman and J.H. Friedman. Predicting multivariate responses in multiple linear regression. *J. Royal. Statist. Soc B.*, 59(1):3–54, 1997.

[Caruana, 1997] R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[Evgeniou *et al.*, 2005] T. Evgeniou, C. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

[Hastie *et al.*, 2001] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer-Verlag, first edition, 2001.

[Heskes, 2000] Tom Heskes. Empirical bayes for learning to learn. In *Proc. 17th International Conf. on Machine Learning*, pages 367–374. Morgan Kaufmann, San Francisco, CA, 2000.

[Jaakkola and Jordan, 1997] T. Jaakkola and M. Jordan. A variational approach to bayesian logistic regression models and their extensions. In *Proceedings of 6th International Worksop on AI and Statistics*, 1997.

[Smeaton and Over, 2003] A.F. Smeaton and P. Over. TRECVID: Benchmarking the effectiveness of information retrieval tasks on digital video. In *Proc. of the Intl. Conf. on Image and Video Retrieval*, 2003.

[Thrun and Pratt, 1998] S. Thrun and L. Pratt. *Learning to Learn*. Kluwer Academic Publishers, 1998.

[Xue *et al.*, 2007] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *J. Mach. Learn. Res.*, 8:35–63, 2007.

[Yan *et al.*, 2004] R. Yan, J. Yang, and A. G. Hauptmann. Learning query-class dependent weights in automatic video retrieval. In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 548–555, 2004.

[Yu *et al.*, 2005] K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In *Proceedings of 22nd International Conference on Machine Learning (ICML)*, 2005.

[Zhang *et al.*, 2005] J. Zhang, Z. Ghahramani, and Y. Yang. Learning multiple related tasks using latent independent component analysis. In *Neural Information Processing Systems 18*, 2005.