Linking Heterogeneous Input Spaces with Pivots for Multi-Task Learning

Jingrui He * Yan Liu† Qiang Yang‡

Abstract

Most existing works on multi-task learning (MTL) assume the same input space for different tasks. In this paper, we address a general setting where different tasks have heterogeneous input spaces. This setting has a lot of potential applications, yet it poses new algorithmic challenges - how can we link seemingly uncorrelated tasks to mutually boost their learning performance?

Our key observation is that in many real applications, there might exist some correspondence among the inputs of different tasks, which is referred to as pivots. For such applications, we first propose a learning scheme for multiple tasks and analyze its generalization performance. Then we focus on the problems where only a limited number of the pivots are available, and propose a general framework to leverage the pivot information. The idea is to map the heterogeneous input spaces to a common space, and construct a single prediction model in this space for all the tasks. We further propose an effective optimization algorithm to find both the mappings and the prediction model. Experimental results demonstrate its effectiveness, especially with very limited number of pivots.

1 Introduction

Multi-task learning (MTL) has been proposed to address the problem of labeled data scarcity in each single task, and the goal is to leverage the label information from all the related tasks to build a better model for each task. Multi-task learning has been widely applied in many real problems, such as sentiment classification in different domains [17], remote sensing [20], cross-lingual classification [23], face recognition [27], etc. In recent years, multi-task learning has attracted extensive research attention from both the application and the algorithm sides (see Section 2 for a brief review). Most existing works assume that different tasks share the *same* input space. In other words, the examples from different tasks are represented by vectors in the same space.

Here, we address a general setting where different tasks have *heterogeneous* input spaces. For example, in cross-lingual classification [23], the first task might be classifying a set of English documents whose input space consists of English vocabulary, and the second task might be classifying a set of German documents whose input space consists of German vocabulary. Another example is simultaneous classification of documents and images [29]. Here the first task is document classification whose input space consists of the document vocabulary, and the second task is image classification whose input space consists of the 'image vocabulary' (e.g., clusters of image features extracted from different regions). Yet another example is cross-domain sentiment classification [22], where different tasks correspond to the classification of documents in different domains with different vocabularies. In all these examples, the two tasks have different input spaces - we refer to this problem setting as *MUSH* (Multi-Task Learning with Heterogeneous Input Spaces). Compared with the existing works on MTL in the same input space, *MUSH* has received much less attention up until now. Yet it is of great interest from the application perspective, since it brings the possibility of jointly learning multiple *arbitrary* tasks so that they could benefit from each other.

MUSH also poses new challenges from the algorithm perspective since multiple tasks seem to be *totally* uncorrelated with each other if their examples are in different input spaces. Here our key observation is that in many real applications, there might exist some correspondence among certain input dimensions of different tasks. In the example of cross-lingual classification, there is a natural correspondence between the words from two different languages (e.g., 'good' in English means 'gut' in German); in the example of document-image classification, some words can be naturally 'translated' into some image regions, and vice versa [28]; in the example of cross-domain sentiment classification, some domain-specific words are linked via the same sentiment polarity. The correspondence across different input spaces provides an important connection among multiple tasks. In this paper, such correspondence is represented by *pivots*, which consist of tuples of input dimensions from multiple tasks bearing the same correspondence.

The major contributions of this paper are three-fold. First of all, when all the pivot information is available,

*Stevens Institute of Technology
†University of Southern California
‡Hong Kong University of Science and Technology

we propose the following learning scheme for MTL: we first map the examples from heterogeneous input spaces of multiple tasks to the same pivot space, and then construct a single prediction model in the pivot space for all the tasks. Second, we analyze this learning scheme in terms of the generalization error of the prediction model for each task. Finally, based on this scheme, we propose a general framework, which, given a limited number of pivots, is able to learn the mappings from the input spaces to a common space and the prediction model in this space. In this framework, to make use of the pivot information, we enforce the input dimensions mapped to the same pivot be projected to the common space in a similar way.

The rest of the paper is organized as follows. In Section 2, we briefly review the related work. The learning scheme for *MUSH* with pivots is presented and analyzed in Section 3. Based on this scheme, in Section 4, we introduce the general framework for *MUSH* with a limited number of pivots. Then we discuss a special case of this framework together with the optimization algorithm in Section 5. In Section 6, we show some experimental results, which demonstrate the effectiveness of our proposed algorithm. Finally, we conclude the paper in Section 7.

2 Related Work

In multi-task learning, task relationship is usually modeled in the following two ways. One is to construct prediction functions in a common (derived) feature space. Here the common feature space can be either the latent space [19, 7, 2, 1], or the augmented feature space [18, 4]. The other is to assume shared function classes in the original feature space and similar parameters [14, 6, 20, 12]. Our proposed model is closer to the first way of modeling task relatedness in that we also map the examples to the common space. However, there are two major differences. One is that we focus on the general setting where different tasks have different input spaces, and the mappings for different tasks in terms of the transformation matrices are not the same. The other is that, for the first time, we make use of the pivot information to guide the calculation of the transformation matrices.

Heterogeneous multi-task learning is one special type of multi-task learning, where the dimensions of the inputs or even the input space are different across different tasks [26, 9]. Several models have been proposed to solve the problem, mostly motivated by cross-lingual applications [8, 25, 23, 29, 15, 16, 10]. For example, the TLRisk algorithm uses a language model to link the class labels to the inputs in the source spaces, which in turn is translated to the inputs in the target spaces [8]; the co-training-based algorithm makes use of Google translate to generate pivots for each word in each language, creates multiple language-dependent views, and then applies the co-training algorithm [5] to construct the classifiers [25]; the CL-SCL algorithm chooses from a large number of pivots the ones that satisfy the support condition, and maps the examples from different tasks to a common space according to the correlation of all the inputs to the selected pivots [23]; the feature augmentation method first transforms the data of heterogeneous input space into a common subspace to measure their similarities and then develops different feature mapping functions for each task to augment the transformed data [10].

As we can see, some existing algorithms require a large number of pivots (or dictionaries) to achieve good results [25, 23], which might be infeasible to obtain in many practical applications; on the other hand, if we apply the multi-view based method [16] in our setting, the common view would correspond to the pivots, and the task-specific views would correspond to the remaining input dimensions. However, the view consistency assumption may not hold if we only have a limited number of pivots, which will impair the performance of this method. In this paper we aim to address these issues and improve the performance in each task with only a limited number of pivots. As a result, the major technical challenge is how to effectively explore the connections between tasks.

3 Learning Scheme for *MUSH* with Pivots

In this section, we propose the learning scheme for *MUSH* with pivots, followed by the analysis regarding its generalization performance.

3.1 Learning scheme Suppose that we have T tasks. Each task has a set of labeled examples $(\mathbf{x}_i^t, y_i^t)$, $(i = 1, \ldots, n^t,\ t = 1, \ldots, T)$, where $\mathbf{x}_i^t \in \mathbb{R}^{d^t}$ denotes the i^{th} example in the t^{th} task, d^t is the dimensionality of the input space for the t^{th} task, y_i^t is the output for input $\mathbf{x}_i^t$ ($y_i^t \in \{1, -1\}$ for classification problems, or $y_i^t \in \mathbb{R}$ for regression problems), and n^t is the number of labeled examples in the t^{th} task, which is usually small in multi-task learning. Notice that each task has its unique input space, and the number of input dimensions may be different across multiple tasks. For example, in cross-lingual classification, the English documents in Task 1 are described by a set of English words, which is different from the set of German words used to describe German documents in Task 2. Furthermore, let $\mathcal{D}^t$ denote the probability distribution of $\mathbf{x}_i^t$ for the t^{th} task. Here, our goal is to leverage the labeled examples from all the tasks to build a good prediction model for each task.

In this paper, we assume that the inputs for different tasks are connected via P pivots, which span the pivot space $\mathbb{R}^P$. In other words, for a certain task, each input dimension can be mapped to a single pivot, and the pivots are shared by multiple tasks. Therefore, in our previous example, the English word 'good' and the German word 'gut' map to the same pivot. On the other hand, multiple input dimensions of a single task may be mapped to the same pivot. For example,

in sentiment classification problems, both the English word 'good' and 'nice' are mapped to the same pivot. Therefore, for the t^{th} task, each example $\mathbf{x}_i^t$ is mapped to $\tilde{\mathbf{x}}_i^t \in \mathbb{R}^P$ $(i = 1, \ldots, n^t)$ such that the value for a pivot is equal to the sum of all the input dimensions mapped to the same pivot. To be specific, if the elements in $\mathbf{x}_i^t$ that correspond to 'good' and 'nice' are 0.7 and 0.4 respectively, then the element in $\tilde{\mathbf{x}}_i^t$ that corresponds to their pivot is 1.1. In this way, we have the induced distribution $\tilde{\mathcal{D}}^t$ for the t^{th} task with probability density function $f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}})$ for $\tilde{\mathbf{x}} \in \mathbb{R}^P$. We also assume that given $\tilde{\mathbf{x}}$, the probability distribution of the label $\tilde{\mathcal{D}}(y)$ is shared by all the tasks.

Based on the above notation, we propose the following learning scheme for *MUSH* with pivots. We first map the examples from all the tasks to the same pivot space, and then construct a single prediction model in this space. In the first step, we use different mappings for different tasks to accommodate the heterogeneous input spaces; and in the second step, we pool all the examples in the pivot space together to build a good prediction model. As we will show in the next subsection, if all the tasks have similar induced distribution in the pivot space, the prediction model will have good generalization performance for each task.

Throughout this paper, we use normal lower-case letters to denote scalers or functions, normal upper-case letters to denote constants, bold-face lower-case letters to denote vectors, and bold-face upper-case letters to denote matrices. Furthermore, in most cases, the superscript of a letter refers to the task index, and the subscript refers to the example index or input dimension index.

3.2 Generalization performance Next, we analyze the proposed learning scheme in terms of the generalization performance of the prediction model in the pivot space. To this end, we first compare two distributions $\tilde{\mathcal{D}}^s$ for the s^{th} task and $\tilde{\mathcal{D}}^t$ for the t^{th} task $(s, t = 1, \ldots, T)$ in the pivot space via the L1 distance, which is defined as

$$d_{L1}(\tilde{\mathcal{D}}^s, \tilde{\mathcal{D}}^t) = 2 \sup_{B \in \mathcal{B}} |\text{Pr}_{\tilde{\mathcal{D}}^s}[B] - \text{Pr}_{\tilde{\mathcal{D}}^t}[B]|$$

where $\mathcal{B}$ denotes the set of measurable subsets under $\tilde{\mathcal{D}}^s$ and $\tilde{\mathcal{D}}^t$. In other words, if two tasks are similar, i.e., they have similar distributions in the pivot space, then their L1 distance should be small. In particular, $d_{L1}(\tilde{\mathcal{D}}^t, \tilde{\mathcal{D}}^t) = 0$.

In this subsection, we focus on classification problems. Let $\mathcal{H}$ denote a hypothesis space of VC-dimension C in the pivot space. Given a classifier $h \in \mathcal{H}$, we define its prediction error with respect to the distribution $\tilde{\mathcal{D}}^t(\tilde{\mathbf{x}})$ for the t^{th} task and $\tilde{\mathcal{D}}(y)$ as follows.

$$\epsilon^t(h) = \text{E}_{\tilde{\mathbf{x}} \sim \tilde{\mathcal{D}}^t, y \sim \tilde{\mathcal{D}}}[I(y \neq h(\tilde{\mathbf{x}}))] = \text{Pr}_{\tilde{\mathbf{x}} \sim \tilde{\mathcal{D}}^t, y \sim \tilde{\mathcal{D}}}[y \neq h(\tilde{\mathbf{x}})]$$

where $I(\cdot)$ is the indicator function. We also define the empirical error in the pivot space as follows.

$$\hat{\epsilon}(h) = \sum_{t=1}^{T} \sum_{i=1}^{n^t} I(y_i^t \neq h(\tilde{\mathbf{x}}_i^t)) / (\sum_{t=1}^{T} n^t)$$

The following theorem builds the connection between $\epsilon^t(h)$ and $\hat{\epsilon}(h)$.

THEOREM 3.1. *For the t^{th} task, with probability at least $1 - \delta$, for every $h \in \mathcal{H}$:*

(3.1)

$$\epsilon^t(h) \leq \hat{\epsilon}(h) + \sqrt{\frac{4}{n}(C \log \frac{2en}{C} + \log \frac{4}{\delta})} + \sum_{s=1}^{T} \frac{n^s}{2n} d_{L1}(\tilde{\mathcal{D}}^s, \tilde{\mathcal{D}}^t)$$

where $n = \sum_{s=1}^{T} n^s$, and e is the Euler number.

Proof. In the pivot space, if we consider all the examples from multiple tasks as coming from a single distribution $\sum_{s=1}^{T} \frac{n^s}{n} \tilde{\mathcal{D}}^s$, then the prediction error $\epsilon(h)$ of h with respect to this distribution can be written as $\epsilon(h) = \sum_{s=1}^{T} \frac{n^s}{n} \epsilon^s(h)$. Therefore the prediction error of h with respect to the distribution $\tilde{\mathcal{D}}^t$ for the t^{th} task can be bounded as follows.

$$\begin{aligned}
\epsilon^t(h) &= \epsilon(h) + \sum_{s=1}^{T} \frac{n^s}{n} (\epsilon^t(h) - \epsilon^s(h)) \\
&= \epsilon(h) + \sum_{s=1}^{T} \frac{n^s}{n} \int_{\tilde{\mathbf{x}}} \text{Pr}_{y \sim \tilde{D}}[y \neq h(\tilde{\mathbf{x}})](f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}}) - f_{\tilde{\mathcal{D}}^s}(\tilde{\mathbf{x}})) d\tilde{\mathbf{x}} \\
&= \epsilon(h) + \sum_{s=1}^{T} \frac{n^s}{n} \cdot \\
&\{ \int_{f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}}) \geq f_{\tilde{\mathcal{D}}^s}(\tilde{\mathbf{x}})} \text{Pr}_{y \sim \tilde{D}}[y \neq h(\tilde{\mathbf{x}})](f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}}) - f_{\tilde{\mathcal{D}}^s}(\tilde{\mathbf{x}})) d\tilde{\mathbf{x}} \\
&- \int_{f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}}) < f_{\tilde{\mathcal{D}}^s}(\tilde{\mathbf{x}})} \text{Pr}_{y \sim \tilde{D}}[y \neq h(\tilde{\mathbf{x}})](f_{\tilde{\mathcal{D}}^s}(\tilde{\mathbf{x}}) - f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}})) d\tilde{\mathbf{x}} \} \\
&\leq \epsilon(h) + \sum_{s=1}^{T} \frac{n^s}{n} \cdot \\
&\int_{f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}}) \geq f_{\tilde{\mathcal{D}}^s}(\tilde{\mathbf{x}})} \text{Pr}_{y \sim \tilde{D}}[y \neq h(\tilde{\mathbf{x}})](f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}}) - f_{\tilde{\mathcal{D}}^s}(\tilde{\mathbf{x}})) d\tilde{\mathbf{x}} \\
&\leq \epsilon(h) + \sum_{s=1}^{T} \frac{n^s}{n} \int_{f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}}) \geq f_{\tilde{\mathcal{D}}^s}(\tilde{\mathbf{x}})} (f_{\tilde{\mathcal{D}}^t}(\tilde{\mathbf{x}}) - f_{\tilde{\mathcal{D}}^s}(\tilde{\mathbf{x}})) d\tilde{\mathbf{x}} \\
&\leq \epsilon(h) + \sum_{s=1}^{T} \frac{n^s}{2n} d_{L1}(\tilde{\mathcal{D}}^s, \tilde{\mathcal{D}}^t)
\end{aligned}$$

On the other hand, by applying Vapnik-Chervonenkis theory [24], we can bound $\epsilon(h)$ by its empirical estimate $\hat{\epsilon}(h)$. Combined with the fact that the VC-dimension of $\mathcal{H}$ is C, we can show that with probability at least $1 - \delta$,

$$\epsilon(h) \leq \hat{\epsilon}(h) + \sqrt{\frac{4}{n}(C \log \frac{2en}{C} + \log \frac{4}{\delta})}$$

Putting everything together, we complete the proof.■

Notice that in Equation (1), we have three terms on the right hand side. The first two terms are obtained as if all the examples come from the same task. The third term is unique

for multi-task learning, and it measures how related/similar different tasks are. On one hand, if all the tasks have the same distribution in the pivot space, then the third term is equal to 0, and Equation (1) is reduced to the bound for single task learning; on the other hand, if all the tasks are completely different in the sense that the L1 distance between any pair of them reaches the maximum value 1, then the third term is equal to 1, which means that multi-task learning does not help in such cases.

According to Theorem 3.1, comparing MTL with Single-Task Learning (STL), the main advantage of MTL is in the fact that it leverages all n labeled examples from multiple tasks to build the prediction model, as opposed to only using n^t labeled examples from the t^{th} task. Therefore, MTL is a effective way to solve the problem of label scarcity.

4 General Framework

The learning scheme proposed in Section 3 is based on the assumption that we know the mapping from each input space to the pivot space. Notice that in many real applications, the mappings may not be available to us or may be expensive to obtain. For example, in cross-lingual classification, given documents in several languages, we may only know the correspondence of a small number of words from different vocabularies. In this case, our goal is to leverage the limited pivot information to help us find the mappings from the input space to a common space (not necessarily the pivot space), as well as the prediction model in this space.

In this section, we assume that we are given p pivots, $p \ll P$. Each pivot is a T-tuple of input dimensions, one dimension from each task, although our proposed framework can be easily generalized to the cases where multiple input dimensions of a single task map to the same pivot. For the ease of explanation, we rearrange the inputs in each task so that the first p dimensions in each task are always contained in the pivots.

In our framework, we assume that examples from multiple tasks can be mapped into a d-dimensional common space, where a single prediction model can be used to predict the outputs of examples from all the tasks. Notice that ideally, d should be equal to P, and the common space is the pivot space. However, due to the lack of the pivot information, we need to learn the mappings from the data. Therefore, the common space is not necessarily the same as the pivot space. The major difference between this assumption and the one used in [2] is: in our case, the transformation matrices for different tasks are different, but the prediction model in the common space is shared by all the tasks; whereas in [2], the transformation matrix is the same for all the tasks, but the prediction models vary among the tasks. In applications such as cross-lingual classification and document-image classification, our assumption is more appropriate. To see this, simply notice that due to the different input spaces of different tasks, we can not use a single transformation matrix for all the tasks. Furthermore, we assume that the input dimensions corresponding to the same pivot have similar prediction power of the output. Therefore, their projection matrices to the common space should be similar. This is motivated by the mappings from the input space to the pivot space introduced in the last section when all P pivots are available: the input dimensions corresponding to the same pivot are mapped to this pivot with weight 1.

Based on the above assumptions, we first use the task-specific transformation matrix $\mathbf{U}^t \in \mathbb{R}^{d\times d^t}$ to map the examples in the t^{th} task to the d-dimensional common space, $d \leq d^t$, $(t = 1, \ldots, T)$. Here, we require that the rows of $\mathbf{U}^t$ be orthonormal. Notice that since the first p input dimensions in each task are contained in the pivots, $\mathbf{U}^t(:, 1 : p)$, the first p columns of $\mathbf{U}^t$, correspond to the projection matrix of these dimensions to the common space. Due to the definition of pivots such that input dimensions contained in the same pivot have similar contribution to the output, $\mathbf{U}^t(:, 1 : p)$ should all be close to a certain matrix $\mathbf{V} \in \mathbb{R}^{d\times p}$. In other words, similar as in [13], the projection matrices for the dimensions contained in pivots are regularized via their mean. Then we construct a linear predictor in the common space with coefficient vector $\mathbf{a} \in \mathbb{R}^d$ based on the labeled examples from all the tasks. To be specific, we propose the following objective function in Equation (2).

$$
\begin{aligned}
\min\ & f_d(\mathbf{U}^t, \mathbf{V}, \mathbf{a}) = \sum_{t=1}^{T}\Big(\frac{1}{n^t}\sum_{i=1}^{n^t} L(y_i^t, \mathbf{a}'\mathbf{U}^t\mathbf{x}_i^t)\Big) \\
& + \sum_{t=1}^{T}(\gamma^t\|\mathbf{U}^t(:, 1:p) - \mathbf{V}\|_F^2) + \gamma\|\mathbf{a}\|^2 \qquad (4.2)\\
& s.t.\ \mathbf{U}^t \cdot (\mathbf{U}^t)' = \mathbf{I}_{d\times d},\ (t = 1, \ldots, T)
\end{aligned}
$$

where $\mathbf{a}'$ denotes the transpose of $\mathbf{a}$, $L(\cdot, \cdot)$ denotes the prediction loss, $\|\cdot\|_F$ denotes matrix Frobenius norm, $\|\cdot\|$ denotes vector L2 norm, $\mathbf{I}_{d\times d}$ is a $d \times d$ identity matrix, γ^t and γ are positive parameters that balance among different terms in this objective function.

In the objective function of Equation (2), the first term measures the average prediction loss of all the tasks, the second term measures the difference between the projection matrices for the dimensions contained in pivots and the matrix $\mathbf{V}$, which encourages similar projections for the dimensions in the same pivot, and the last term penalizes the L2 norm of the linear classifier in the common space. When $\gamma^t = 0$ $(t = 1, \ldots, T)$, Equation (2) imposes weak relatedness among multiple tasks, since it only requires the L2 norm of all the coefficient vectors to be the same, and the pivot information is not used. On the other hand, when $\gamma^t \to \infty$, $(t = 1, \ldots, T)$, the projection matrices for the dimensions contained in pivots are all forced to be $\mathbf{V}$. We will discuss this special case in Section 5. The

constraints in Equation (2) require that the transformation matrices $\mathbf{U}^t$ be orthonormal. This is to avoid the non-identifiability problem. For each task, the matrices that satisfy this constraint constitute the Stiefel manifold [11].

With respect to the generalization performance of this prediction model, we have the following lemma.

LEMMA 4.1. *For the t^{th} task, with probability at least $1-\delta$, for every $h \in \mathcal{H}$:*

$$\epsilon^t(h) \leq \hat{\epsilon}(h) + \sqrt{\frac{4}{n}((d+1)\log\frac{2en}{d+1} + \log\frac{4}{\delta})} + \sum_{s=1}^{T}\frac{n^s}{2n}d_{L1}(\tilde{\mathcal{D}}^s, \tilde{\mathcal{D}}^t)$$

Proof. Theorem 3.1 combined with the fact that the VC-dimension of linear classifiers in the d-dimensional common space is $d+1$ proves this lemma.

The objective function in Equation (2) is not jointly convex in $\mathbf{U}^t$, $\mathbf{U}$, and $\mathbf{a}$. However, as long as $L(\cdot,\cdot)$ is convex with respect to the second argument, the objective function is convex with respect to each of its variables. Therefore, we could apply block coordinate descent to iteratively update $\mathbf{U}^t$, $\mathbf{U}$ and $\mathbf{a}$, which is guaranteed to converge to a local optimum [21]. Furthermore, in order to update the orthonormal transformation matrices, we could apply Newton or conjugate gradient methods on the Stiefel manifold [11]. For the special case of the optimization framework to be discussed in the next section, we will introduce a simple solution via gradient descent.

Before ending this section, we would like to point out that the minimum of the objective function f_d depends on the dimensionality d of the common space. The following theorem shows that it is non-decreasing with respect to d.

THEOREM 4.1. *For any pair of positive integers d_1 and d_2, such that $d_2 > d_1$, $\min\ f_{d_2} \geq \min\ f_{d_1}$.*

Proof. Without loss of generality, assume that $d_2 = d_1 + 1$. When the dimensionality of the common space is d_2, let $\mathbf{U}_2^t$ denote the transformation matrix of size $d_2 \times d^t$ for the t^{th} task ($t = 1, \ldots, T$). Therefore, $\mathbf{U}_2^t \cdot (\mathbf{U}_2^t)' = I_{d_2 \times d_2}$. Let $\mathbf{a}_2$ denote any coefficient vector of length d_2.

On the other hand, when the dimensionality of the common space is d_1, we construct the set of orthonormal transformation matrices $\mathbf{U}_1^t \in d_1 \times d^t$ ($t = 1, \ldots, T$) as follows. Let the first $d_1 - 1$ rows of $\mathbf{U}_1^t$ be the same as $\mathbf{U}_2^t$, and the last row of $\mathbf{U}_1^t$ be $\frac{\mathbf{a}_2(d_1)\mathbf{U}_2^t(d_1,:)+\mathbf{a}_2(d_2)\mathbf{U}_2^t(d_2:)}{\sqrt{(\mathbf{a}_2(d_1))^2+(\mathbf{a}_2(d_2))^2}}$, where $\mathbf{a}_2(d_1)$ is the $d_1{}^{\text{th}}$ element in $\mathbf{a}_2$, and $\mathbf{U}_2^t(d_1,:)$ denotes the $d_1{}^{\text{th}}$ row in $\mathbf{U}_2^t$. It is easy to verify that $\mathbf{U}_1^t$ is orthonormal. The coefficient vector $\mathbf{a}_1 \in \mathbb{R}^{d_1}$ is constructed in a similar way by setting $\mathbf{a}_1(i) = \mathbf{a}_2(i)$ $(i = 1, \ldots, d_1 - 1)$, and $\mathbf{a}_1(d_1) = \sqrt{(\mathbf{a}_2(d_1))^2 + (\mathbf{a}_2(d_2))^2}$.

Notice that in Equation (2), the minimum value of the objective function is obtained when $\mathbf{V} = \frac{1}{n^t}\sum_{t=1}^{T}\mathbf{U}^t(:, 1:p)$. Let $\mathbf{V}_i = \frac{1}{n^t}\sum_{t=1}^{T}\mathbf{U}_i^t(:, 1:p)$ $(i = 1, 2)$. It is easy to see that the first $d_1 - 1$ rows of $\mathbf{V_1}$ and $\mathbf{V_2}$ are the same, and the last row of $\mathbf{V}_1$ is equal to $\frac{\mathbf{a}_2(d_1)\mathbf{V}_2(d_1,:)+\mathbf{a}_2(d_2)\mathbf{V}_2(d_2,:)}{\sqrt{(\mathbf{a}_2(d_1))^2+(\mathbf{a}_2(d_2))^2}}$.

Based on the two sets of transformation matrices $\mathbf{U}_1^t$ and $\mathbf{U}_2^t$ ($t = 1, \ldots, T$), $\mathbf{V}_1$, $\mathbf{V}_2$, the coefficient vectors $\mathbf{a}_1$ and $\mathbf{a}_2$, we can compare $f_{d_1}(\mathbf{U}_1^t, \mathbf{V}_1, \mathbf{a}_1)$ and $f_{d_2}(\mathbf{U}_2^t, \mathbf{V}_2, \mathbf{a}_2)$. In Equation (2), the first term $\mathbf{a}_1'\mathbf{U}_1^t = \mathbf{a}_2'\mathbf{U}_2^t$ $(t = 1, \ldots, T)$, and the third term $\|\mathbf{a}_1\|^2 = \|\mathbf{a}_2\|^2$. Therefore, we only need to compare the second term. $\forall t = 1, \ldots, T$,

$$\begin{aligned}
&\|\mathbf{U}_2^t(:, 1:p) - \mathbf{V}_2\|_F^2 - \|\mathbf{U}_1^t(:, 1:p) - \mathbf{V}_1\|_F^2 \\
=&\|\mathbf{U}_2^t(d_1:d_2, 1:p) - \mathbf{V}_2(d_1:d_2, :)\|_F^2 \\
&- \|\mathbf{U}_1^t(d_1, 1:p) - \mathbf{V}_1(d_1, :)\|^2 \\
\geq&\|\mathbf{U}_2^t(d_1:d_2, 1:p) - \mathbf{V}_2(d_1:d_2, :)\|^2 \\
&- \|[\frac{\mathbf{a}_2(d_1)}{\sqrt{(\mathbf{a}_2(d_1))^2 + (\mathbf{a}_2(d_2))^2}}\ \frac{\mathbf{a}_2(d_2)}{\sqrt{(\mathbf{a}_2(d_1))^2 + (\mathbf{a}_2(d_2))^2}}]\cdot \\
&(\mathbf{U}_2^t(d_1:d_2, 1:p) - \mathbf{V}_2(d_1:d_2, :))\|^2 \\
\geq&0
\end{aligned}$$

Therefore, the minimum of $f_{d_2}(\mathbf{U}_2^t, \mathbf{V}_2, \mathbf{a}_2)$ is no smaller than the minimum of $f_{d_1}(\mathbf{U}_1^t, \mathbf{V}_1, \mathbf{a}_1)$.■

5 A Special Case of the Framework

In this section, we discuss a special case of the general framework proposed in Section 4 when $\gamma^t \to \infty$, $(t = 1, \ldots, T)$. In this case, $\mathbf{U}^t(:, 1:p) = \mathbf{V}$, $(t = 1, \ldots, T)$. That is, the projection matrices for the input dimensions contained in pivots are exactly the same across all the tasks. $\forall t = 1, \ldots, T$, let $\mathbf{U}^t = [\mathbf{V}\ \mathbf{V}^t]$, where $\mathbf{V}^t \in \mathbb{R}^{d \times (d^t - p)}$. The problem in Equation (2) can be written as follows.

(5.3)
$$\min\ g_d(\mathbf{V}^t, \mathbf{V}, \mathbf{a}) = \sum_{t=1}^{T}\sum_{i=1}^{n^t}\frac{L(y_i^t, \mathbf{a}'[\mathbf{V}\ \mathbf{V}^t]\mathbf{x}_i^t)}{n^t} + \gamma\|\mathbf{a}\|^2$$
$$s.t.\ \mathbf{V}\cdot(\mathbf{V})' + \mathbf{V}^t\cdot(\mathbf{V}^t)' = \mathbf{I}_{d\times d},\ (t = 1, \ldots, T)$$

In Theorem 4.1, we have shown that the minimum value of the objective function in Equation (2) depends on d, the dimensionality of the common space. Although Equation (3) is a special case of Equation (2), the following theorem shows that under certain conditions, the minimum of g_d is independent of d.

THEOREM 5.1. *If $d < \min(p, d^t - p)$, then Equation (3) is equivalent to the following optimization problem.*

(5.4)
$$\min\ g(\mathbf{u}^t, c) = \sum_{t=1}^{T}(\frac{1}{n^t}\sum_{i=1}^{n^t}L(y_i^t, (\mathbf{u}^t)'\mathbf{x}_i^t)) + \gamma c^2$$
$$s.t.\ \mathbf{u}^t(1:p) = \mathbf{v}$$
$$\|\mathbf{u}^t\| = c,\ (t = 1, \ldots, T)$$

where $\mathbf{u}^t \in \mathbb{R}^{d^t}$, $\mathbf{u}^t(1:p)$ denotes the first p elements of $\mathbf{u}^t$, and $\mathbf{v} \in \mathbb{R}^p$ is the common coefficient vector for the input dimensions contained in pivots.

Proof. First of all, it is easy to see when $d = 1$, the coefficient vector $\mathbf{a}$ is reduced to a scaler, and Equation (3) is equivalent to Equation (4). Next, we need to show that the minimum of g_d does not depend on d. According to Theorem 4.2, when $d > 1$, the minimum of g_d is no smaller than the minimum of g. We will prove that the minimum of g is also no smaller than the minimum of g_d. To this end, next we show that given any feasible solution to Equation (4), there always exists a feasible solution to Equation (3) such that their objective functions have the same value.

Let $\mathbf{u}^t = c[\mathbf{v}' \ (\mathbf{v}^t)']' \ (t = 1, \ldots, T)$ denote the set of feasible coefficient vectors for Equation (4), where $\mathbf{v}^t \in \mathbb{R}^{d^t - p}$, and $\|\mathbf{v}\|^2 + \|\mathbf{v}^t\|^2 = 1$. Let $\mathbf{V} = [\mathbf{v} \ \mathbf{W}']'$ and $\mathbf{V}^t = [\mathbf{v}^t \ (\mathbf{W}^t)']'$ denote a set of matrices constructed based on $\mathbf{v}$ and $\mathbf{v}^t$ where $\mathbf{W} \in \mathbb{R}^{(d-1)\times p}$ and $\mathbf{W}^t \in \mathbb{R}^{(d-1)\times(d^t-p)}$ $(t =, 1 \ldots, T)$. Next, we prove that $\mathbf{W}$ and $\mathbf{W}^t$ can be constructed in such a way that $\mathbf{V} \cdot (\mathbf{V})' + \mathbf{V}^t \cdot (\mathbf{V}^t)' = \mathbf{I}_{d\times d}$ $(t = 1, \ldots, T)$. In other words,

$$
\begin{aligned}
&\mathbf{V} \cdot (\mathbf{V})' + \mathbf{V}^t \cdot (\mathbf{V}^t)' \\
&= \begin{bmatrix} \mathbf{v}' \\ \mathbf{W} \end{bmatrix} [\mathbf{v} \ \mathbf{W}'] + \begin{bmatrix} (\mathbf{v}^t)' \\ \mathbf{W}^t \end{bmatrix} \left[\mathbf{v}^t \ (\mathbf{W}^t)'\right] \\
&= \begin{bmatrix} \mathbf{v}'\mathbf{v} + (\mathbf{v}^t)'\mathbf{v}^t & \mathbf{v}'\mathbf{W}' + (\mathbf{v}^t)'(\mathbf{W}^t)' \\ \mathbf{W}\mathbf{v} + \mathbf{W}^t\mathbf{v}^t & \mathbf{W}\mathbf{W}' + \mathbf{W}^t(\mathbf{W}^t)' \end{bmatrix} \\
&= \begin{bmatrix} 1 & \mathbf{v}'\mathbf{W}' + (\mathbf{v}^t)'(\mathbf{W}^t)' \\ \mathbf{W}\mathbf{v} + \mathbf{W}^t\mathbf{v}^t & \mathbf{W}\mathbf{W}' + \mathbf{W}^t(\mathbf{W}^t)' \end{bmatrix} = \mathbf{I}_{d\times d}
\end{aligned}
$$

Therefore, $\mathbf{W}$ and $\mathbf{W}^t$ need to satisfy the following equations.

$$\mathbf{W}\mathbf{v} + \mathbf{W}^t\mathbf{v}^t = \mathbf{0} \tag{5}$$

$$\mathbf{W}(\mathbf{W})' + \mathbf{W}^t(\mathbf{W}^t)' = \mathbf{I}_{(d-1)\times(d-1)} \tag{6}$$

Let $\mathbf{A}\mathbf{\Lambda}\mathbf{A}'$ denote the eigen-decomposition of $\mathbf{W}\mathbf{W}'$, where $\mathbf{A} \in \mathbb{R}^{(d-1)\times(d-1)}$ is an orthogonal matrix, and $\mathbf{\Lambda} \in \mathbb{R}^{(d-1)\times(d-1)}$ is a diagonal matrix whose diagonal elements are non-negative. From Equation (6), it is easy to see that the eigen-decomposition of $\mathbf{W}^t(\mathbf{W}^t)'$ is $\mathbf{A}(\mathbf{I}_{(d-1)\times(d-1)} - \mathbf{\Lambda})\mathbf{A}'$, and the diagonal elements of $\mathbf{\Lambda}$ are no greater than 1. Therefore, the SVD decomposition of $\mathbf{W}$ can be written as $\mathbf{A}\sqrt{\mathbf{\Lambda}}\mathbf{B}$, where $\sqrt{\mathbf{\Lambda}}$ is a diagonal matrix whose diagonal elements are square root of $\mathbf{\Lambda}$, $\mathbf{B} \in \mathbb{R}^{(d-1)\times p}$, and $\mathbf{B}\mathbf{B}' = \mathbf{I}_{(d-1)\times(d-1)}$. Similarly, the SVD decomposition of $\mathbf{W}^t$ can be written as $\mathbf{A}\sqrt{\mathbf{I}_{(d-1)\times(d-1)} - \mathbf{\Lambda}}\mathbf{B}^t$, where $\mathbf{B}^t \in \mathbb{R}^{(d-1)\times(d^t-p)}$, and $\mathbf{B}^t(\mathbf{B}^t)' = \mathbf{I}_{(d-1)\times(d-1)}$. Therefore, Equation (5) becomes,

$$\sqrt{\mathbf{\Lambda}}\mathbf{B}\mathbf{v} + \sqrt{\mathbf{I}_{(d-1)\times(d-1)} - \mathbf{\Lambda}}\mathbf{B}^t\mathbf{v}^t = 0$$

Let $\mathbf{\Lambda} = \|\mathbf{v}^t\|^2\mathbf{I}_{(d-1)\times(d-1)}$, then $\mathbf{I}_{(d-1)\times(d-1)} - \mathbf{\Lambda} = \|\mathbf{v}\|^2\mathbf{I}_{(d-1)\times(d-1)}$, and the above equation is equivalent to

$$\mathbf{B}\frac{\mathbf{v}}{\|\mathbf{v}\|} + \mathbf{B}^t\frac{\mathbf{v}^t}{\|\mathbf{v}^t\|} = 0$$

Therefore, we can construct $\mathbf{B}$ and $\mathbf{B}^t$ as follows. Let the first row of $\mathbf{B}$ be $\frac{\mathbf{v}'}{\|\mathbf{v}\|}$, and the first row of $\mathbf{B}^t$ be $-\frac{(\mathbf{v}^t)'}{\|\mathbf{v}^t\|}$. Let the remaining rows of $\mathbf{B}$ ($\mathbf{B}^t$) be unit-length row vectors that are orthogonal to the first row as well as orthogonal among themselves, which is always achievable since $d < \min(p, d^t - p)$. Finally, by setting $\mathbf{a}(1) = c$ and $\mathbf{a}(i) = 0$ $(i = 2, \ldots, d)$, $g_d(\mathbf{V}^t, \mathbf{V}, \mathbf{a})$ is equal to $g(\mathbf{u}^t, c)$.

Putting everything together, we have shown that for any feasible solution to Equation (4), there always exists a feasible solution to Equation (3) with the same objective function value. Therefore, the minimum of g is no smaller than the minimum of g_d. Together with Theorem 4.2, we complete the proof.■

Based on Theorem 5.1, we can see that when the dimensionality of the common space d is smaller than both p and $d^t - p$, $(t = 1, \ldots, T)$, the value of d will not affect the optimal solution, and Equation (3) is equivalent to Equation (4), where we only need to minimize the objective function with respect to a set of equal-length vectors $\mathbf{u}^t$, whereas in Equation (3), we need to optimize with respect to a set of orthonormal matrices. Let $\mathbf{u}^t = [\mathbf{v}', (\mathbf{v}^t)']' \ (t = 1, \ldots, T)$. According to the constraints in Equation (4), $\mathbf{v}^t$ should have equal lengths.

Similar as before, the optimization problem in Equation (4) is not jointly convex with respect to $\mathbf{v}$ and $\mathbf{v}^t$. Here we propose *MUSH-p*, a gradient descent based method to find a local optimum, which is described in Algorithm 1.

Algorithm 1 *MUSH-p* algorithm

Input: $(\mathbf{x}_i^t, y_i^t)$, $(i = 1, \ldots, n^t, t = 1, \ldots, T)$, $L(\cdot, \cdot)$, p, γ, the number of iteration steps I

Output: $\mathbf{u}^t$, $(t = 1, \ldots, T)$

1: Initialize $\mathbf{v}$ and $\mathbf{v}^t$ to be zero vectors
2: **for** $i = 1$ to I **do**
3: Calculate the gradient $\nabla\mathbf{v} = \frac{\partial g}{\partial \mathbf{v}}$ and $\nabla\mathbf{v}^t = \frac{\partial g}{\partial \mathbf{v}^t}$, $(t = 1, \ldots, T)$
4: Perform line search to find the optimal step size c^t, $(t = 1, \ldots, T)$, such that $\|\mathbf{v}^t - c^t\nabla\mathbf{v}^t\|$ is a constant across different tasks
5: For $t = 1, \ldots, T$, update $\mathbf{v}^t \leftarrow \mathbf{v}^t - c^t\nabla\mathbf{v}^t$
6: Perform line search to find the optimal step size c^0, and update $\mathbf{v} \leftarrow \mathbf{v} - c^0\nabla\mathbf{v}$
7: **end for**
8: Generate $\mathbf{u}^t = [\mathbf{v}', (\mathbf{v}^t)']'$, $(t = 1, \ldots, T)$

Algorithm 1 works as follows. In Step 1, we initialize all the vectors to be zero. Then we iteratively update the vectors S times. To be specific, in Step 3, we calculate the gradient vectors $\nabla\mathbf{v}$ and $\nabla\mathbf{v}^t$; in Step 4, we perform line search to find the optimal step sizes while maintaining the equal-length constraint, which can be done by varying the step size for one task and calculating the step sizes for the

other tasks; in Step 5, we update $\mathbf{v}^t$; and in Step 6, we update the common vector $\mathbf{v}$ for the input dimensions contained in pivots. Finally, in Step 8, we obtain the weight vectors $\mathbf{u}^t$ by concatenating $\mathbf{v}$ and $\mathbf{v}^t$. During the test phase, we predict the output of $\mathbf{x}^t$ from the t^{th} task using the sign of $(\mathbf{u}^t)'\mathbf{x}^t$ for classification problems, or the value of $(\mathbf{u}^t)'\mathbf{x}^t$ for regression problems.

6 Experimental Results

In this section we evaluate the performance of the proposed algorithms. In particular, we would like to answer the following questions: (1) Does the performance of *MUSH-p* improve as more pivots are given? (2) Given different labeled set sizes, how does *MUSH-p* compare with alternative competitors?

6.1 Experimental set-up In this paper, we use the cross-lingual sentiment classification data set from [23]. It consists of 4 million product reviews from Amazon.{de|fr|co.jp}, which are written in three languages: German, French, and Japanese. Furthermore, the corpus is extended with English product reviews from [3], and dictionaries between English and the three languages are available. To assign class labels, a review with >3 (<3) stars is labeled as positive (negative), and other reviews are discarded. For each language, the labeled reviews are grouped into 3 categories: books, dvd, and music [23]. Based on this data set, we can create multi-task learning problems as follows. We pick a category, say books, and assign reviews from this category in 2 different languages to 2 tasks, say English and German[1]. The goal is to leverage the labeled reviews from both tasks to classify the unlabeled reviews.

Due to the scarcity of existing methods for *MUSH* which are able to utilize pivot information, we adapt the CL-SCL algorithm for transfer learning [23] to the multi-task learning setting for the sake of comparison. Notice that our proposed *MUSH-p* algorithm is supervised, whereas CL-SCL is semi-supervised. Therefore, in our experiments, in addition to the labeled training set, CL-SCL is also given the unlabeled test set in each task, and its performance on these test sets is reported for comparison. For this algorithm, we set the parameters the same way as in [23]. For a fair comparison, CL-SCL is provided with the same set of pivots as our proposed algorithms. Besides CL-SCL, we also compare with two simple baselines. The first one is single task learning, denoted as STL, where multiple tasks are learned separately without considering the pivot information. The second one is to project all the examples to the space spanned by the pivots, and construct a single classifier in this space to predict the examples from different tasks, which is denoted as Pivot-Space. For the proposed *MUSH-p* we use the negative log-likelihood in logistic regression as the loss function $L(\cdot,\cdot)$, and the parameters are chosen by cross-validation.

[1]Since we only have dictionaries between English and the other languages, one task always consists of reviews in English.

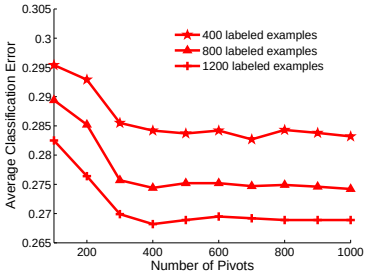
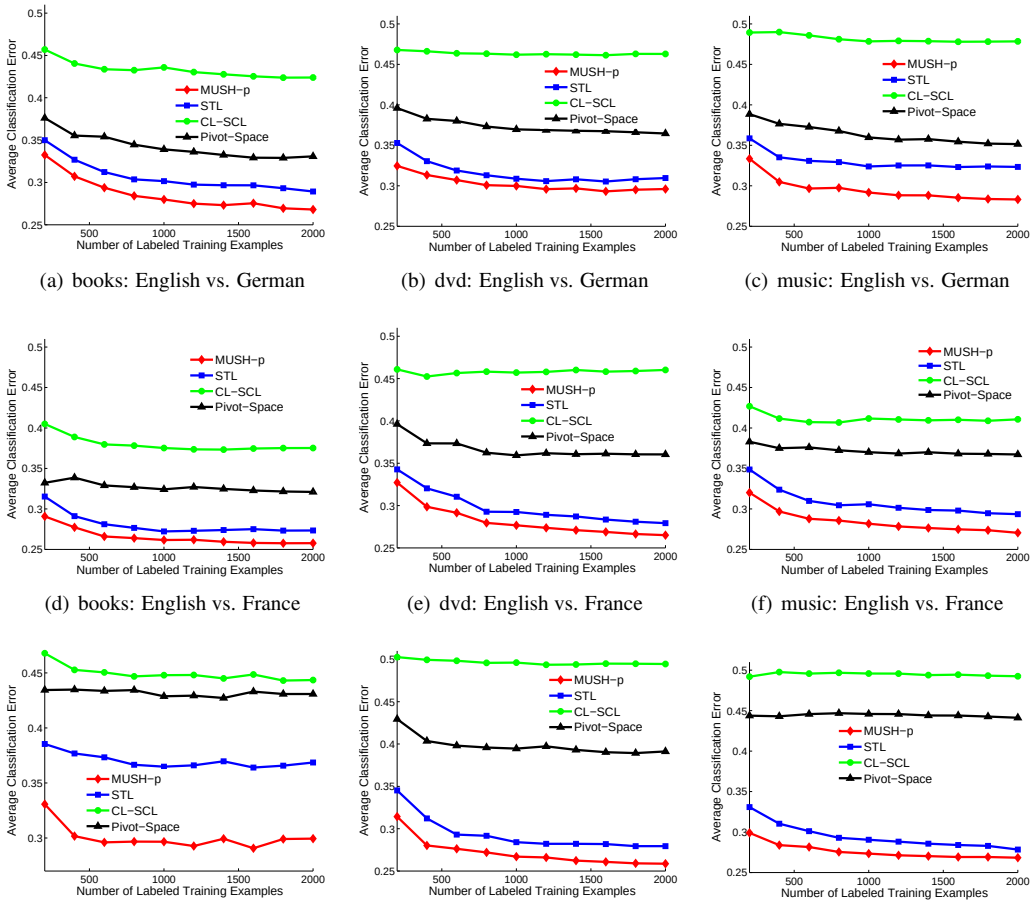


Figure 1: Performance study with various number of pivots.

6.2 Pivot study In this subsection, we answer the first question. To this end, we pick the reviews from the category of books written the English and German languages, vary the number of pivots provided to *MUSH-p*, and report the average classification error with 400, 800, and 1200 labeled examples respectively in Figure 1. From this figure, we can see that all 3 curves follow the same trend: in the beginning when we only have a small number of pivots, more pivot information helps improve the performance, and the performance is stabilized around 400 pivots, beyond which no improvement is observed with additional pivot information.

6.3 Comparison with alternative competitors In this subsection, we answer the second question. Here we show the comparison results on 9 multi-task learning problems in Figure 2. In these figures, the x-axis shows the total number of labeled training examples, and the y-axis is the average classification error on the test sets of the two tasks over 10 runs. The number of available pivots is fixed at 100. From these figures, we have the following observations. (1) The performance of *MUSH-p* is better than STL in most cases, which also uses logistic regression in the two tasks, but ignores the pivots. It shows that making use of the pivot information to build the connection among different tasks can indeed improve the performance. (2) The performance of *MUSH-p* is also better than Pivot-Space, which combines the label information from all the tasks, but ignores the non-pivot input dimensions. This is because when the number of pivots is small, constructing the classifier solely based on the pivots tends to lose much information. (3) The performance of CL-SCL is the worst of all the competitors, which can be explained as follows. In CL-SCL, the subspace of bilingual classifiers is approximated using the pivot predictors. A small number of pivots may negatively affect this approximation, which in turn affects its performance.



(a) books: English vs. German (b) dvd: English vs. German (c) music: English vs. German

(d) books: English vs. France (e) dvd: English vs. France (f) music: English vs. France

(g) books: English vs. Japanese (h) dvd: English vs. Japanese (i) music: English vs. Japanese

Figure 2: Comparison of average classification error with various labeled set sizes: *MUSH-p* is consistently better than the competitors.

7 Conclusion

In this paper, we study *MUSH*, a general setting in MTL where different tasks have heterogeneous input spaces. To this end, we first propose a learning scheme, which maps the examples from all the tasks to the pivot space, and then construct a single prediction model in this space. The performance of the prediction model is shown to depend on both the total number of examples, as well as the similarity among multiple tasks. Based on this scheme, we focus on the problems where only a limited number of pivots are available, and propose an optimization framework to find both the mappings and the prediction model. Finally, we propose the *MUSH-p* algorithm to solve a special case of this framework. Experimental results demonstrate the effectiveness of *MUSH-p*.

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*, 6:1817–1853, 2005.

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[3] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 2007.

[4] J. Blitzer, R. T. McDonald, and F. Pereira. Domain adaptation with structural correspondence learning. In *EMNLP*, pages 120–128, 2006.

[5] A. Blum and T. M. Mitchell. Combining labeled and unlabeled sata with co-training. In *COLT*, 1998.

[6] E. V. Bonilla, K. M. Chai, and C. K. I. Williams. Multi-task gaussian process prediction. In *NIPS*, 2007.

[7] J. Chen, L. Tang, J. Liu, and J. Ye. A convex formulation for learning shared structures from multiple tasks. In *ICML*, page 18, 2009.

[8] W. Dai, Y. Chen, G.-R. Xue, Q. Yang, and Y. Yu. Translated learning: Transfer learning across different feature spaces. In *NIPS*, pages 353–360, 2008.

[9] W. Dai, O. Jin, G.-R. Xue, Q. Yang, and Y. Yu. Eigentransfer: a unified framework for transfer learning. In *ICML*, page 25, 2009.

[10] L. Duan, D. Xu, and I. W. Tsang. Learning with augmented features for heterogeneous domain adaptation. In *ICML*, 2012.

[11] A. Edelman, T. s, A. Arias, Steven, and T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM J. Matrix Anal. Appl*, 20:303–353, 1998.

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

[13] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *KDD*, pages 109–117, 2004.

[14] P. Gong, J. Ye, and C. Zhang. Multi-stage multi-task feature learning. In *NIPS*, pages 1997–2005, 2012.

[15] M. Harel and S. Mannor. Learning from multiple outlooks. In *ICML*, pages 401–408, 2011.

[16] J. He and R. Lawrence. A graphbased framework for multi-task multi-view learning. In *ICML*, pages 25–32, 2011.

[17] H. D. III. Bayesian multitask learning with latent hierarchies. *CoRR*, abs/0907.0783, 2009.

[18] H. D. III. Frustratingly easy domain adaptation. *CoRR*, abs/0907.1815, 2009.

[19] B. Lin, S. Yang, C. Zhang, J. Ye, and X. He. Multi-task vector field learning. In *NIPS*, pages 296–304, 2012.

[20] Q. Liu, X. Liao, and L. Carin. Semi-supervised multitask learning. In *NIPS*, 2007.

[21] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, Massachusetts, 2nd edition edition, 1973.

[22] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen. Cross-domain sentiment classification via spectral feature alignment. In *WWW*, pages 751–760, 2010.

[23] P. Prettenhofer and B. Stein. Cross-language text classification using structural correspondence learning. In *ACL*, pages 1118–1127, 2010.

[24] V. Vapnik. *Statistical learning theory*. Wiley, 1998.

[25] X. Wan. Co-training for cross-lingual sentiment classification. In *ACL/AFNLP*, pages 235–243, 2009.

[26] Q. Yang, Y. Chen, G.-R. Xue, W. Dai, and Y. Yu. Heterogeneous transfer learning for image clustering via the socialweb. In *ACL/AFNLP*, pages 1–9, Suntec, Singapore, August 2009.

[27] Y. Zhang and J. Schneider. Learning multiple tasks with a sparse matrix-normal penalty. In *NIPS*, 2010.

[28] X. Zhu, A. B. Goldberg, M. Eldawy, C. R. Dyer, and B. Strock. A text-to-picture synthesis system for augmenting communication. In *AAAI*, pages 1590–, 2007.

[29] Y. Zhu, Y. Chen, Z. Lu, S. J. Pan, G.-R. Xue, Y. Yu, and Q. Yang. Heterogeneous transfer learning for image classification. In *AAAI*, 2011.