ECAI 2020
G.D. Giacomo et al. (Eds.)

doi:10.3233/FAIA200216

On Partial Multi-Task Learning

Yi He[1] and Baijun Wu[1] and Di Wu[2] and Xindong Wu[3]

Abstract. Multi-Task Learning (MTL) has shown its effectiveness in real applications where many related tasks could be handled together. Existing MTL methods make predictions for multiple tasks based on the data examples of the corresponding tasks. However, the data examples of some tasks are expensive or time-consuming to collect in practice, which reduces the applicability of MTL. For example, a patient may be asked to provide her microtome test reports and MRI images for illness diagnosis in MTL-based system [37, 40]. It would be valuable if MTL can predict the abnormalities for such medical tests by feeding with some easy-to-collect data examples from other related tests instead of directly collecting data examples from them. We term such a new paradigm as multi-task learning from partial examples.

The challenges of partial multi-task learning are twofold. First, the data examples from different tasks may be represented in different feature spaces. Second, the data examples could be incomplete for predicting the labels of all tasks. To overcome these challenges, we in this paper propose a novel algorithm, named Generative Learning with Partial Multi-Tasks (GPMT). The key idea of GPMT is to discover a shared latent feature space that *harmonizes* disparate feature information of multiple tasks. Given a partial example, the information contained in its missing feature representations is recovered by projecting it onto the latent space. A learner trained on the latent space then enjoys complete information included in the original features and the recovered missing features, and thus can predict the labels for the partial examples. Our theoretical analysis shows that the GPMT guarantees a performance gain comparing with training an individual learner for each task. Extensive experiments demonstrate the superiority of GPMT on both synthetic and real datasets.

1 Introduction

To improve learning performance, Multi-Task Learning (MTL) solves multiple related tasks jointly, such that the knowledge learned from one task could be leveraged by others [5]. MTL has shown success in many real applications. For example, in medical service, physicians usually ask a patient to take several tests before making the final diagnosis. To aid this process, MTL is introduced to learn from various related physical testing results to predict the abnormality from each test. Then, based on the predicted abnormalities, an expert system [9, 22] could be used to diagnose the patient.

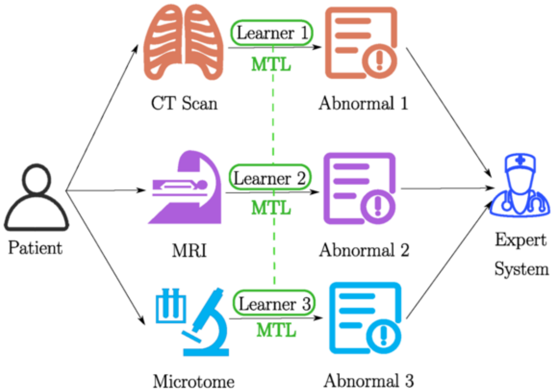

Figure 1: A typical MTL-based diagnosis system, in which the learners that predict the abnormalities from the physical test results are trained with MTL techniques.

Existing MTL approaches require data examples from all related tasks to make an accurate prediction [2]. As an example shown in Figure 1, to help diagnose if a patient has lung cancer or not, MTL is fed with different kinds of laboratory results (*e.g.* the images or reports of CT scan, MRI, and microtome test) as input. However, some physical tests, such as MRI and microtome, could be expensive and time-consuming, reducing the applicability of MTL in such scenarios. The question then is: instead of obtaining the complete data examples, can we make predications based on partial information (*i.e.* the data for some tasks are not presented) in MTL?

To address this issue, we in this paper propose the *Generative Learning with Partial Multi-Tasks* (GPMT) algorithm, which can handle and leverage the partial information effectively. The proposed algorithm is motivated on a key observation: since the tasks in MTL are related and share commonalities, there exists relatedness among the features across tasks. Following this spirit, GPMT learns a universal feature space so as to uniformly project data examples, came from different tasks and represented in different feature spaces, onto it. Specifically, GPMT employs a generative graph to capture the feature relatedness. The vertices of the graph denote the features in the universal feature space, and the out-edge weights of a vertex represent the relatedness between the corresponding feature and the others. As a result, such a graph can be used to map the original feature space of any data example to the universal feature space.

Notably, if we simply regenerate the missing features based on the learned generative graph, the constructed universal space may be very sparse in practice. Revisit the medical example: hundreds of physical tests can be provided, while only a few of them are necessary when diagnosing a patient. In this case, the features from the unnecessary tests are apparently less useful comparing with those from the necessary tests. Hence, the values of the useless features regen-

[1] School of Computing and Informatics, University of Louisiana at Lafayette, USA. Email: {yi.he1, bj.wu}@louisiana.edu

[2] Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, China. Email: wudi@cigit.ac.cn

[3] Mininglamp Academy of Sciences, Mininglamp Technology, Beijing, China; Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Hefei, China. Email: wuxindong@mininglamp.com

erated in the universal feature space are small or even zeroes, resulting in a sparse representation. Such a sparse representation induces high dimensionality and, therefore, deteriorates the informativeness and representativeness of the feature representations in the universal feature space. A learner trained directly on the universal feature space suffers from performance degradation, a phenomena known as "curse of dimensionality". We address this problem by embedding the reconstructed universal feature space into a low-dimensional shared latent space. The embedded latent space should preserve useful feature information in the universal feature space. To this end, we leverage the supervised information to guide the embedding process, such that the less the suffered training loss, the more informative the embedded latent space.

Specific contributions in this paper:

1. We explore a new problem, named Partial Multi-Task Learning (PMTL), which aims to predict data examples with partial feature information, rather than the conventional MTL methods that require a full feature information from all data examples.
2. A novel GPMT algorithm is proposed to solve the PMTL problem, with its performance bound being analyzed.
3. Extensive experiments on 9 benchmark datasets and 1 real-world dataset have been carried out to demonstrate the superiority of our proposal.

The rest of this paper is organized as follows. Section 2 discusses related work. Section 3 covers the preliminaries. Section 4 scrutinizes the building blocks of the proposed approach. Section 5 and Section 6 report the theoretical and experimental results, respectively. We conclude the work in Section 7. Due to the page limitation, we put the detailed derivations and proofs in supplementary material[4].

2 Related Work

Our work is related with multi-task learning, multi-view learning, and multi-label learning methods. This section discusses the relationships and differences with the existing literatures.

Multi-Task Learning (MTL). The pioneer works by [2, 15] shows that solving multiple tasks jointly can reduce the hypothesis space of training learners and improve generalization performances, compared with solving each task solely. From the data perspective, [24, 34, 42] propose to learn an augmented feature representation space which conveys a higher level of discriminant power than the original features. From the learner perspective, [1, 14] suggest that the feature relatedness among multiple tasks leads to redundancy in model parameters. To get rid of the redundant information, the learners trained on different tasks are merged and approximated in a low-rank latent space. These works train an individual learner on each task with respect to the fact that both the feature space and label semantics of different tasks can be disparate. Thus, they cannot predict a specific label for the data examples that are not represented in the corresponding task.

Multi-View Learning (MVL). The literatures in this class aim to improve the model performance in a task (view) of interest by leveraging information from multiple auxiliary tasks (views). Comparing with MTL, on the one hand, MVL pays more attention on the task of interest, rather than MTL which treats all tasks equally. On the other hand, MVL strictly requires the feature spaces of different tasks are non-identical. As such, from a technical viewpoint, MVL can be deemed as a special case of MTL, where the tasks are heterogeneous yet share the same semantic label as the prediction target. Prior works [20, 26, 28, 32, 36] envision a consensus pattern matrix that is extracted from the multiple data matrices of different views. Thus, the models trained on auxiliary views can be leveraged via the bridge of the extracted pattern matrix. Moreover, [8, 16, 21] propose to directly learn asymmetric mappings between the view of interest and each of the auxiliary views, such that the model trained on auxiliary views can be applied to the view of interest by simply projecting the data samples represented in the view of interest onto the auxiliary views. Unfortunately, all these methods require a full information of all views, which cannot be satisfied in our setting. A recent work by [23] lifted such requirement and allows missing data representations in some views, however, it focuses on clustering instead of supervised learning, where both the technical challenges and solutions are different from our work.

Multi-Label Learning (MLL). As we consider the labels of different tasks convey disparate semantic meanings, our PMTL problem is also related with the MLL problem. The literatures in this class fall into three categories: First-order approaches [3,7,41] that decompose the MLL problem into a number of independent binary classification subproblems (one subproblem per label), and solve the subproblems independently; Second-order approaches [11, 13] that aim to capture the pairwise ranking relationship among pairs of labels; and High-order approaches [30,31,38] that are computationally more demanding and less scalable than the former ones yet possess stronger correlation-modeling capabilities via addressing connections among random subsets of labels. These methods assume that the data examples are generated from a single feature space, while in our setting, there exists multiple feature spaces. Indeed, it is difficult for a single feature space to capture the information required to label a large number of semantic meanings. Thus, addressing multi-label challenge in the PMTL problem is meaningful.

3 Preliminaries

We first summarize some notations used in this paper. Bold uppercase and lowercase characters are used for matrices (*e.g.*$\mathbf{A}$) and vectors (*e.g.*$\mathbf{a}$), respectively. For any matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, $\mathbf{A}_{i,j}$ denotes its $(i,j)^{\text{th}}$ entry, and for any vector $\mathbf{a} = [a_1, a_2, \ldots, a_n]^\top \in \mathbb{R}^n$, a_i denotes its i^{th} element. $\|\cdot\|_2$ denotes the ℓ_2-norm of a vector. For a non-square matrix $\mathbf{A}$, its Moore-Penrose pseudo-inverse is denoted by $\mathbf{A}^\dagger$.

3.1 The PMTL Problem

Given m tasks $\{\mathcal{T}_i\}_{i=1}^m$. Let $\mathcal{X}_i = \mathbb{R}^{d_i}$ and $\mathcal{Y} = \{y^{(1)}, \ldots, y^{(m)}\}$ denote the feature space of $\mathcal{T}_i$ and the label space of all tasks, respectively. Without loss of generality, we consider that the data examples from different tasks are represented in different feature spaces, and hence $\mathcal{X}_i \neq \mathcal{X}_j$ for any $i \neq j$.

Suppose each task $\mathcal{T}_i$ has n_i data examples and there are n data examples in total. Each data example is allowed to appear in an arbitrary number of tasks, and its labels in different tasks convey disparate semantic meanings. Denoted by $(\mathbf{x}_j, \mathbf{y}_j) = ([\mathbf{x}_j^{(1)}, \ldots, \mathbf{x}_j^{(k)}], [y_j^{(1)}, \ldots, y_j^{(m)}]) \in \mathcal{X}_1 \times \ldots \times \mathcal{X}_k \times \mathcal{Y}$ the j^{th} training example appearing in k tasks $\mathcal{T}_1, \ldots, \mathcal{T}_k$, where $1 \leq k < m$ and k varies for different data examples. Let $\mathbf{x}_j^{(i)} \in \mathbb{R}^{d_i}$ be a d_i-dimensional feature vector and $y_j^{(i)}$ be its groundtruth label, *i.e.* $y_j^{(i)} \in \{-1, +1\}$ for classification and $y_j^{(i)} \in \mathbb{R}$ for regression. In

[4] shorturl.at/hxLM6

other words, for each data example $\mathbf{x}_j$, we know its groundtruth labels in all m tasks yet can only observe its feature representations in partial k tasks. Our goal is to find an optimal hypothesis ψ, such that the empirical risk $\mathcal{L} = \frac{1}{n}\sum_{j=1}^{n} \ell\big(\mathbf{y}_j, \psi(\mathbf{x}_j)\big)$ is minimized, with ℓ being a predefined loss function, *e.g.* logistic loss, square loss, *etc*.

3.2 Multi-Task Generative Graphical Model

Let $\mathcal{U} := \mathcal{X}_1 \cup \ldots \cup \mathcal{X}_m \in \mathbb{R}^{d_1+\ldots+d_k+\ldots+d_m}$ denote a *universal feature space* that unions the feature spaces of all m tasks. For each given partial example, a generative graph is to embed a mapping $\phi : \mathbb{R}^{d_1+\ldots+d_k} \mapsto \mathcal{U}$, completing its feature representations in all m tasks from the partially observed k tasks. Denoted by $\mathcal{G}$ the graph whose vertices represent the features in $\mathcal{U}$ and weights on edges represent the relatedness between pairs of features. Specifically, for a vertex v, a vector containing the weights of its all out-edges is denoted as θ_v. The graph $\mathcal{G}$ thus can be represented by a matrix $\Theta = [\theta_1, \ldots, \theta_{|\mathcal{U}|}]^\top \in \mathbb{R}^{|\mathcal{U}|\times|\mathcal{U}|}$.

We define $\mathbf{u}_j := [\mathbf{x}_j, \tilde{\mathbf{x}}_j]^\top \in \mathcal{U}$ as a desired completion of of the j^{th} partial example's feature representation, where $\tilde{\mathbf{x}}_j \in \mathbb{R}^{d_{k+1}+\ldots+d_m}$ denotes its missing feature representation in the $(m-k)$ tasks $\mathcal{T}_{k+1}, \ldots, \mathcal{T}_m$. Learning $\mathcal{G}$ is to maximize the following log-likelihood:

$$\mathcal{Q} = \sum_{j=1}^{n} \log \mathbb{P}(\mathbf{u}_j|\mathbf{x}_j, \Theta), \tag{1}$$

where the features in $\mathcal{U}$ are approximated independently by given each task's representation $\mathbf{x}_j^{(i)}$. As such, we have:

$$\mathbb{P}(\mathbf{u}_j|\mathbf{x}_j, \Theta) = \prod_{i=1}^{k} \mathbb{P}(\mathbf{u}_j|\mathbf{x}_j^{(i)}, \Phi^{(i)}), \tag{2}$$

where we define $\Phi^{(i)} := [\theta_1, \ldots, \theta_{d_i}]^\top \in \mathbb{R}^{d_i\times|\mathcal{U}|}$ for simplicity.

In this work, we estimate the data generating process in $\mathcal{U}$ with a mixture of exponential Gaussian distributions [12]. Specifically, we let each Gaussian be contributed from a task where $\mathbf{x}_j$ appears. This makes an intuitive sense, since (i) the tasks in which $\mathbf{x}_j$ is missing contribute zero information for approximating $\mathbf{u}_j$; and (ii) the informativenesses of different tasks are non-identical, and the more informative tasks should play more significant roles in the generative modelling. The distribution density function in Eq. (2) is defined as:

$$\mathbb{P}(\mathbf{u}_j|\mathbf{x}_j^{(i)}, \Phi^{(i)}) \propto \exp\big(-\tfrac{\delta_i}{2}(\mathbf{u}_j - \mathbb{E}(\mathbf{u}_j))^\top \Sigma^{-1}(\mathbf{u}_j - \mathbb{E}(\mathbf{u}_j))\big), \tag{3}$$

where $\mathbb{E}(\mathbf{u}_j)$ is approximated based on ϕ given $\Phi^{(i)}$ and $\mathbf{x}_j^{(i)}$, and Σ is a constant semi-positive definite matrix. Denoted by δ_i the impact of i^{th} Gaussian contributed by the i^{th} task $\mathcal{T}_i$ and $\delta_i \in [0, 1]$. To train the graphical model, the iterative sampling techniques such as MCMC [19] can be readily applied, which yet may yield expensive training cost. We in Section 4.3 discuss our training strategy which differs from the sampling techniques and is more efficient.

4 Our Approach

In this section, we scrutinize the building blocks of our GPMT approach, and discuss the intuitions behind its design. Section 4.1 details the discovery of the latent feature space, and Section 4.2 elaborates the method to train a unified learner. We combine the latent space discovery, the learner training process, and the generative graph learning process into one optimization problem, and propose an efficient strategy to solve it in Section 4.3.

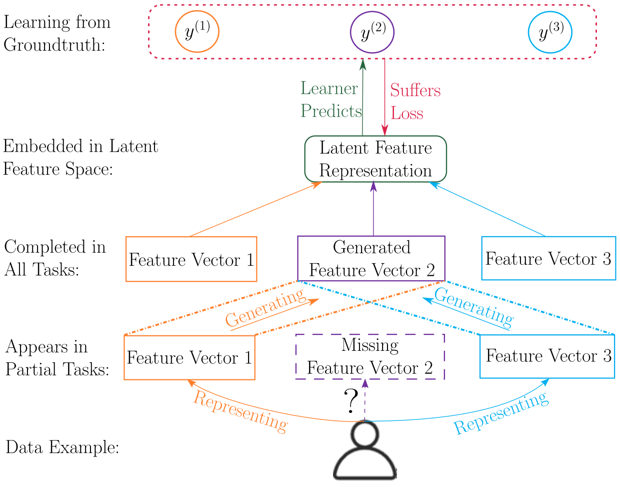

Figure 2: Learning protocol of our GPMT approach. Each partial example that appears in several tasks is represented with corresponding feature vectors. For the tasks they do not appear, the missing feature representations are generated from the observed feature vectors. Then, the observed and generated feature vectors are together embedded into a latent feature space. The learner is trained on the embedded latent space based on the discrepancy between the predictions and the groundtruth labels.

4.1 Latent Space Discovery under Supervision

Since the reconstructed space may be sparse in practice, we discover a shared latent feature space with low dimension to preserve useful feature information in the universal feature space. As shown in Figure 2, the wellness of the discovered latent space directly depends on the reconstructed universal feature space. However, if the data examples only appear in few tasks or if the tasks that they appear lie in less-informative feature spaces, the missing feature representations are likely to be generated with *noises*. These noises will persist and escalate in the later latent space discovery, and negatively affect the accurateness of the trained learner.

To address this issue, we exploit the plentiful supervised information conveyed by the the groundtruth labels to guide the discovery of the latent space. Specifically, the learner is trained on the latent feature representations – the less noises the latent space contains, the better the learning performance it can achieve. This idea leads to the development of the optimization problem as follows.

$$\begin{aligned}\mathop{\arg\min}_{\{\mathbf{w}_i\}_{i=1}^{m}, \mathcal{K}} \sum_{i=1}^{m}\sum_{j=1}^{n} \ell\big(y_j^{(i)}, \mathbf{w}_i^\top \mathcal{K}_{\text{emb}}(\mathbf{u}_j)\big) \\ + \frac{\lambda_1}{2}\|\mathbf{u}_j - \mathcal{K}_{\text{rec}}\big(\mathcal{K}_{\text{emb}}(\mathbf{u}_j)\big)\|_2^2,\end{aligned} \tag{4}$$

where $\{\mathbf{w}_i\}_{i=1}^{m}$ denotes the learner consisting of a set of linear classifiers *w.r.t.* all m tasks, in which all $\mathbf{w}_i \in \mathbb{R}^z$ with z being the dimension of the latent space. The function set $\mathcal{K} = \{\mathcal{K}_{\text{emb}}, \mathcal{K}_{\text{rec}}\}$ comprises (i) the embedding function $\mathcal{K}_{\text{emb}} : \mathcal{U} \mapsto \mathbb{R}^z$ which extracts the latent space from the universal feature space; and (ii) the recovering function $\mathcal{K}_{\text{rec}} : \mathbb{R}^z \mapsto \mathcal{U}$ which recovers the universal feature space from the latent space. Such a design draws insights from autoencoders [18], where the discovered latent space is enforced to contain sufficient information that can accurately recover the universal feature space. In this work, for the convenience of theoretical analysis and without loss of generality, we restrict our interest in searching $\mathcal{K}$ in the family of linear models. In practice, one may implement

$\mathcal{K}$ with deep generative networks [29] so as to model data with very complicated underlying distributions.

Eq. (4) couples the discovery of latent space with the training of learner. The intuition behind Eq. (4) is that the learner, trained on the latent space if well-constructed, should make minimum prediction errors over all tasks. The loss function ℓ is convex and Lipschitz in its first argument. For different types of tasks, ℓ can be defined differently – for example, logistic loss for classification and square loss for regression.

4.2 Learning Unified Learner via Respecting Task Correlations

The learner in Eq. (4) is defined as a set of independent classifiers, in which the i^{th} classifier learns from one specific label in $\mathcal{T}_i$. This strategy may yield high accuracy, yet does not scale to sizeable databases. In practice, a sizeable database (*e.g.* webpage categorization) can easily contain millions of data examples each of which is annotated with thousands of different labels. Training one classifier for each label needs to scan the whole database many times, incurring high computational and storage cost.

To improve the learning efficiency, we wish to train a unified learner that can handle multiple tasks simultaneously. However, the increased number of target labels tends to augment the searching space of the optimal learner in a combinatorial fashion, leading to poor convergence rate. Therefore, it is necessary to constrain the growth of the searching space before the start of training a unified learner.

In this work, we consider posing the constraint by respecting the correlation information among tasks from two channels – label semantics and feature spaces. Revisit the medical diagnosis example in Figure 1. First, the abnormalities (labels) predicted from the physical test results are highly correlated such as lung lesions (from CT scan) and pulmonary nodules (from MRI). Enforcing the learner to compromise such correlated label semantics can intuitively reduce its searching space.

Second, two patients who have been detected as having similar abnormalities usually suffer from similar symptoms (*e.g.* pulmonary pain or bleed) and hence their representations in the latent space should be close to each other, and be faraway to those who have disparate abnormalities. Based on these two observations, the feature-induced dictionary encoding (FIDE) [39] is employed to respect the correlation information, which not only captures the label correlations but also uses the label correlations to regularize the discovered latent space. Details follow.

Denoted by $\mathbf{M} \in \mathbb{R}^{n \times n}$ the dictionary that encodes the sparse reconstruction relationships among data examples in the label space. Each entry $\mathbf{M}_{i,j}$ denotes the influence of $\mathbf{y}_i$ in reconstructing the predicted labels of the j^{th} data example, and $\mathbf{M}_{i,j} \neq \mathbf{M}_{j,i}$ in most cases. The reconstruction error is defined as:

$$\mathcal{E}_{\text{label}} = \sum_{j=1}^{n} \frac{1}{2} \|\hat{\mathbf{y}}_j - \sum_{\substack{i=1 \\ j \neq i}}^{m} \mathbf{y}_i \mathbf{M}_{i,j}\|_2^2, \tag{5}$$

where $\hat{\mathbf{y}}_j = \mathbf{W}^\top \mathcal{K}_{\text{emb}}(\mathbf{u}_j)$ is the predicted labels of the j^{th} data example for all m tasks with $\mathbf{W} \in \mathbb{R}^{z \times m}$ being the unified learner that predicts all m labels simultaneously.

The intuition behind why minimizing Eq. (5) can reduce the searching space of learner is that, if two labels are correlated, the number of data examples that support such correlation (*i.e.* the patients who have both lung lesions and pulmonary nodules) should be larger than those that do not. Given a new data example, if its representation in the latent space is closer to those supporting data examples, the learner, if it gives disparate predictions on the correlated labels, will incur large reconstruction error in Eq. (5). For the simplicity of notations, we define $\dot{\mathbf{Y}}_j := \{\mathbf{y}_1, \ldots, \mathbf{y}_n\} \setminus \mathbf{y}_j \in \mathbb{R}^{m \times (n-1)}$ and $\mathbf{m}_j := \{\mathbf{M}_{1,j}, \ldots, \mathbf{M}_{n,j}\} \setminus \mathbf{M}_{j,j} \in \mathbb{R}^{n-1}$. Eq. (5) can be rewritten as:

$$\mathcal{E}_{\text{label}} = \sum_{j=1}^{n} \frac{1}{2} \|\hat{\mathbf{y}}_j - \dot{\mathbf{Y}}_j \mathbf{m}_j\|_2^2. \tag{6}$$

The dictionary $\mathbf{M}$ is then used to regularize the discovered latent space. Similar to Eq. (6), we define the reconstruction error in the latent space as:

$$\mathcal{E}_{\text{latent}} = \sum_{j=1}^{n} \frac{1}{2} \|\mathcal{K}_{\text{emb}}(\mathbf{u}_j) - \dot{\mathbf{K}}_j \mathbf{m}_j\|_2^2, \tag{7}$$

where $\dot{\mathbf{K}}_j := \{\mathcal{K}_{\text{emb}}(\mathbf{u}_1), \ldots, \mathcal{K}_{\text{emb}}(\mathbf{u}_n)\} \setminus \mathcal{K}_{\text{emb}}(\mathbf{u}_j) \in \mathbb{R}^{z \times (n-1)}$. Eq. (7) enforces that, in the latent space, the data examples that have similar labels are represented closely to each other while those have disparate labels are separated.

4.3 An Efficient Optimization Strategy

Our solution to the PMTL problem, by incorporating Eq. (4), Eq. (6), and Eq. (7), boils down to the following tri-objective optimization problem:

$$\begin{aligned} \underset{\mathbf{W}, \mathcal{K}, \mathbf{M}}{\arg\min} \sum_{j=1}^{n} \ell\big(\mathbf{y}_j, \mathbf{W}^\top \mathcal{K}_{\text{emb}}(\mathbf{u}_j)\big) + \frac{\lambda_1}{2} \|\mathbf{u}_j - \mathcal{K}_{\text{rec}}\big(\mathcal{K}_{\text{emb}}(\mathbf{u}_j)\big)\|_2^2 \\ + \lambda_2(\mathcal{E}_{\text{label}} + \mathcal{E}_{\text{latent}}), \end{aligned} \tag{8}$$

where the optimal solution does not have a closed form. Hence, one may consider to use iterative methods to seek the parameters of $\mathbf{W}$, $\mathcal{K}$, and $\mathbf{M}$ that deliver a saddle point of Eq. (8). However, this strategy suffers from the expensiveness of obtaining $\mathbf{u}_j$ at each iteration in which the graphical model is re-trained with expensive sampling techniques such as MCMC [19]. On the other hand, the decoupling of the graph learning and the other learning procedures ignores the plentiful supervised information, and thus may degrade the accurateness of the obtained $\mathbf{u}_j$.

To overcome the above two issues, we propose to embed the graph learning process into the optimization of Eq. (8), such that the learning efficiency can be improved, and in turn the supervised information can be leveraged to help learn a better generative graph. The optimization problem defined in Eq. (8) can be equivalently reformulated as follows (for details refer to Section 1 of supplementary material):

$$\begin{aligned} \underset{\mathbf{W}, \mathcal{Z}, \mathbf{M}}{\arg\min} \sum_{j=1}^{n} \ell\big(\mathbf{y}_j, \mathbf{W}^\top \mathcal{Z}_{\text{F}}^\top \mathbf{x}_j\big) + \frac{\lambda_1}{2} \|\mathbf{x}_j - \mathcal{Z}_{\text{B}}^\top \mathcal{Z}_{\text{F}}^\top \mathbf{x}_j\|_2^2 \\ + \frac{\lambda_2}{2} \big(\|\mathbf{W}^\top \mathcal{Z}_{\text{F}}^\top \mathbf{x}_j - \dot{\mathbf{Y}}_j \mathbf{m}_j\|_2^2 + \|\mathcal{Z}_{\text{F}}^\top (\mathbf{x}_j - \dot{\mathbf{X}}_j \mathbf{m}_j)\|_2^2\big), \end{aligned} \tag{9}$$

where $\dot{\mathbf{X}}_j := \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \setminus \mathbf{x}_j \in \mathbb{R}^{(d_1 + \ldots + d_k) \times (n-1)}$, and $\mathcal{Z} = \{\mathcal{Z}_{\text{F}}, \mathcal{Z}_{\text{B}}\}$ comprising two mapping functions, namely, $\mathcal{Z}_{\text{F}}$ (forward) : $\mathbb{R}^{d_1 + \ldots + d_k} \mapsto \mathbb{R}^z$ and $\mathcal{Z}_{\text{B}}$ (backward) : $\mathbb{R}^z \mapsto \mathbb{R}^{d_1 + \ldots + d_k}$. We have

$$\begin{aligned} &\mathcal{Z}_{\text{F}}^\top \mathbf{x}_j = \mathcal{K}_{\text{emb}}\big(\operatorname{diag}(\delta) \Theta^\top \mathbf{x}_j\big) = \sum_{i=1}^{k} \mathcal{K}_{\text{emb}}\big(\delta_i \Phi^{(i)\top} \mathbf{x}_j^{(i)}\big) \in \mathbb{R}^z, \\ &\mathcal{Z}_{\text{B}}^\top \mathcal{Z}_{\text{F}}^\top \mathbf{x}_j = \operatorname{diag}(\delta)^\dagger \mathcal{K}_{\text{rec}}(\mathcal{Z}_{\text{F}}^\top \mathbf{x}_j) \in \mathbb{R}^{d_1 + \ldots + d_k}. \end{aligned}$$

Notably, given trained $\mathbf{W}$ and $\mathcal{Z}$, the predicted labels for a partial example $\mathbf{x}_j$ can be directly calculated as $\hat{\mathbf{y}}_j = \mathbf{W}^\top \mathcal{Z}_\mathrm{F}^\top \mathbf{x}_j$, with $\mathcal{Z}_\mathrm{F}^\top \mathbf{x}_j$ being the feature representation of $\mathbf{x}_j$ in the discovered latent space. This means that the universal feature space does not need to be truly visited in making predictions, which greatly reduces the number of operations involved in matrix multiplications. Further, we define $\mathbf{W}^\top \mathcal{Z}_\mathrm{F}^{(i)\top} \mathbf{x}_j^{(i)} := \mathbf{W}^\top \mathcal{K}_\mathrm{emb}(\Phi^{(i)\top} \mathbf{x}_j^{(i)})$ as the predictions made based on $\mathbf{x}_j^{(i)}$, the feature representation of $\mathbf{x}_j$ in $\mathcal{T}_i$ only. We have $\hat{\mathbf{y}}_j = \sum_{i=1}^{k} \delta_i \mathbf{W}^\top \mathcal{Z}_\mathrm{F}^{(i)\top} \mathbf{x}_j^{(i)}$, where δ_i determines the impact of the task $\mathcal{T}_i$ in making predictions. This makes an intuitive sense since the feature spaces that different tasks lie in have different levels of informativeness. Let $L_T^{(i)} = \frac{1}{n_i} \sum_{t=1}^{T} \sum_{j=1}^{n_i} \ell\big(\mathbf{y}_j, \mathbf{W}^\top \mathcal{Z}_\mathrm{F}^{(i)\top} \mathbf{x}_j^{(i)}\big)$ denote the average cumulative loss suffered by making predictions based on $\mathcal{T}_i$ solely over T iterations, we update δ_i at the $T + 1^{\text{th}}$ iteration as:

$$\delta_i = \exp\big(-\tau L_T^{(i)}\big) / \sum_{i=1}^{k} \exp\big(-\tau L_T^{(i)}\big) \tag{10}$$

where τ is a tuned parameter and its value assignment is discussed in Section 5. If the task which lies in a less informative feature space that incurs larger $L_T^{(i)}$, then its corresponding δ_i is negatively rewarded by Eq. (10) in an exponential fashion.

In this work, we adapt the *Blockwise Gradient Descent* (BGD) solvers [25, 27] to optimize Eq. (9). Following the common steps of BGD, we (i) decompose Eq. (9) into two optimization subproblems in which one subproblem is *w.r.t.* $\mathbf{W}$ and $\mathcal{Z}$ and the other is *w.r.t.* $\mathbf{M}$; and (ii) solve Eq. (9) by alternating between the two subproblems, minimizing over one while keeping the other one fixed. For simplicity, let $\mathcal{F}$ denote the main function in Eq. (9), and ℓ is implemented as the square loss, namely, $\ell(\mathbf{y}_j, \hat{\mathbf{y}}_j) = (1/2)\|\mathbf{y}_j - \hat{\mathbf{y}}_j\|_2^2$. The partial derivatives of $\mathcal{F}$ *w.r.t.* $\mathbf{W}$, $\mathcal{Z}$, and $\mathbf{M}$ are as follows:

$$\nabla_\mathbf{W} \mathcal{F} = -\mathcal{Z}_\mathrm{F}^\top \mathbf{x}_j \big(\mathbf{y}_j - (1-\lambda_2)\hat{\mathbf{y}}_j - \lambda_2 \dot{\mathbf{Y}}_j \mathbf{m}_j\big)^\top, \tag{11}$$

$$\begin{aligned}\nabla_{\mathcal{Z}_\mathrm{F}} \mathcal{F} = &-\mathbf{x}_j \big(\mathbf{y}_j - (1-\lambda_2)\hat{\mathbf{y}}_j - \lambda_2 \dot{\mathbf{Y}}_j \mathbf{m}_j\big)^\top \mathbf{W}^\top \\ &- \lambda_1 \mathbf{x}_j \big(\mathbf{x}_j - \mathcal{Z}_\mathrm{B}^\top \mathcal{Z}_\mathrm{F}^\top \mathbf{x}_j\big)^\top \mathcal{Z}_\mathrm{B}^\top \\ &+ \lambda_2 (\mathbf{x}_j - \dot{\mathbf{X}}_j \mathbf{m}_j)(\mathbf{x}_j - \dot{\mathbf{X}}_j \mathbf{m}_j)^\top \mathcal{Z}_\mathrm{F},\end{aligned} \tag{12}$$

$$\nabla_{\mathcal{Z}_\mathrm{B}} \mathcal{F} = -\lambda_1 \mathcal{Z}_\mathrm{F}^\top \mathbf{x}_j (\mathbf{x}_j - \mathcal{Z}_\mathrm{B}^\top \mathcal{Z}_\mathrm{F}^\top \mathbf{x}_j)^\top, \tag{13}$$

$$\nabla_\mathbf{M} \mathcal{F} = -\lambda_2 \big(\dot{\mathbf{Y}}_j^\top (\hat{\mathbf{y}}_j - \dot{\mathbf{Y}}_j \mathbf{m}_j) + \dot{\mathbf{X}}_j^\top \mathcal{Z}_\mathrm{F} \mathcal{Z}_\mathrm{F}^\top (\mathbf{x}_j - \dot{\mathbf{X}}_j \mathbf{m}_j)\big). \tag{14}$$

We summarize the main steps of our approach in Algorithm 1. As a common setting in the gradient-based optimization methods [6], the step size η is set as a varied rate which yields aggressive updates at the initial iterations and then judicious updates at the later iterations. In this way, our algorithm enjoys a faster convergence rate than that uses a fixed step size. Moreover, in step 8, we only update $\mathbf{m}_j$, a partial column vector of $\mathbf{M}$ that corresponds to the given input $\mathbf{x}_j$, rather than updating the entire dictionary $\mathbf{M}$.

5 Theoretical Analysis

In this section, we derive the performance bound of GPMT algorithm, aiming to answer the following question:

Q1 *Given partial examples, can performance gain be expected via learning from multi-tasks jointly than from each task individually?*

For the sake of soundness, the proofs of this section are provided in Section 2 of supplementary material.

Algorithm 1: The GPMT algorithm

```
Input :
  1: Partial examples for training: ${(\mathbf{x}_j, \mathbf{y}_j) | j = 1, 2, \ldots, n}$;
  2: Nonnegative parameters $\lambda_1$ and $\lambda_2$;
  3: Maximal number of iterations $T$;
1  Initialize $\mathbf{W} \in \mathbb{R}^{z \times m}$, $\mathbf{M} \in \mathbb{R}^{n \times n}$, $\mathcal{Z} = \{\mathcal{Z}_\mathrm{F}, \mathcal{Z}_\mathrm{B}\}$, and
     $t \leftarrow 0$;
2  repeat
3      $t \leftarrow t + 1$ ;
4      for $j = 1, \ldots, n$ do
5          Receive data example $(\mathbf{x}_j, \mathbf{y}_j)$;
6          Initialize a varied step $\eta \leftarrow \sqrt{1/(t * n + j)}$;
7          Optimize $\mathbf{W}$ and $\mathcal{Z}$ by fixing $\mathbf{M}$ via Eq. (11),
            Eq. (12), and Eq. (13):
              $\mathbf{W} \leftarrow \mathbf{W} - \eta \nabla_\mathbf{W} \mathcal{F}$
              $\mathcal{Z}_\mathrm{F} \leftarrow \mathcal{Z}_\mathrm{F} - \eta \nabla_{\mathcal{Z}_\mathrm{F}} \mathcal{F}$
              $\mathcal{Z}_\mathrm{B} \leftarrow \mathcal{Z}_\mathrm{B} - \eta \nabla_{\mathcal{Z}_\mathrm{B}} \mathcal{F}$
8          Optimize $\mathbf{M}$ by fixing $\mathbf{W}$ and $\mathcal{Z}$ using Eq. (14) as:
              $\mathbf{M} \leftarrow \mathbf{M} - \eta \nabla_\mathbf{M} \mathcal{F}$
9      Update $\delta_i$ for each $\mathcal{T}_i$ using Eq. (10) with $\tau = \sqrt{8 \ln m / T}$;
10 until convergence or $t$ exceeds $T$;
11 return $\mathbf{W}$ and $\mathcal{Z}_\mathrm{F}$;
```

Theorem 1. *Denoted by $L_T^{GPMT} = \frac{1}{n} \sum_{t=1}^{T} \sum_{j=1}^{n} \ell(\mathbf{y}_j, \hat{\mathbf{y}}_j)$ the average cumulative loss of GPMT suffered by learning from n partial examples across m tasks over T iterations. Assume the loss function ℓ is convex in its first argument. L_T^{GPMT} with parameter $\tau = \sqrt{8 \ln m / T}$ satisfies*

$$L_T^{GPMT} \leq \min\{L_T^{(i)}\}_{i=1}^{m} + \sqrt{(\ln m/2)T}. \tag{15}$$

Let $\Delta = \sqrt{(\ln m/2)T}$, which is log-bounded by the number of tasks and sub-linearly bounded by the number of training iterations. Theorem 1 suggests that L_T^{GPMT} is comparable to the minimum of $L_T^{(1)}, \ldots, L_T^{(m)}$ which is the average cumulative loss suffered by making predictions on the task that lies in the most informative feature space among all m tasks. Furthermore, we have the following theorem.

Theorem 2. *Suppose the feature space of $\mathcal{T}_b$ is more informative than those of the other tasks, satisfying $L_T^{(b)} < L_T^{(i)}$, for any $b \neq i$. Over T iterations, L_T^{GPMT} is bounded as:*

$$L_T^{GPMT} < L_T^{(b)} + C, \tag{16}$$

where C is a constant, and $C \ll \Delta$.

Theorem 1 and 2 validate that our learning algorithm possesses a nice property as follows.

Corollary 1. *The learning performance is improved by making use of multiple tasks.*

Proof. On the one hand, for the tasks lie in more informative feature spaces (*e.g.* the most informative task $\mathcal{T}_b$), Theorem 2 tells that L_T^{GPMT} is comparable to $L_T^{(b)}$ and is tightly bounded to a small constant. On the other hand, for those tasks lie in less informative feature spaces, it is obvious that other tasks that have more informative feature spaces are helpful. Furthermore, if the feature space of a task

$\mathcal{T}_a$ is less informative than that of $\mathcal{T}_b$ to a certain degree, satisfying $L_T^{(a)} - L_T^{(b)} > \Delta$, it is easy to verify that $L_T^{\text{GPMT}} < L_T^{(a)}$. To conclude, given partial examples, the learner trained on multi-tasks achieves better performance than that trained on an individual task, *i.e.* **Q1** is answered. □

6 Experiments

In this section, we first introduce the experimental setups in Section 6.1, and then present and extrapolate the experimental results in Section 6.2.

6.1 General Setup

6.1.1 Dataset

We perform the experiments on 10 datasets consisting of 9 synthetic datasets and 1 real dataset. Table 1 summarizes the statistics of the studied datasets, and the details follow.

Synthetic Datasets are prepared by following the same idea in [17]. We select 9 UCI datasets [10] that spanned a broad range of domains, including *audio*, *image*, *etc*., whose scales vary from 502 to 16,105 and dimensions vary from 68 to 1449. These datasets only have one feature space at first. We map the original datasets via two random Gaussian matrices, so that we have data represented in three tasks. Since the data are complete in all tasks, we randomly remove 50% data examples for each task while making sure that each data example appears in at least one task.

Real Dataset is collected by ourselves during the research project called the Communities that Care Youth Survey (CCYS), which is administered and funded by the Louisiana government. The data are collected from 79,988 students from 6th to 12th grades via questionnaires provided by 8 independent agencies (therefore, 8 tasks). The questionnaires assess the students' exposure to different sets of social factors (*e.g*. family, neighborhood, school, *etc*.), and collect in total 355 features. Our goal is to predict the students's involvements in 8 different behaviors, including academic performance, mental health, substance abuse, *etc*. The students are not obligated to finish all questionnaires, and therefore, the partial examples refer to the students who only answer a few questions, *i.e*. those students who arbitrarily appear or absent in the tasks. By calculation, the average missing ratio is about 60% (in other datasets the ratio is 50% as fixed).

6.1.2 Compared Methods

We compare the proposed GPMT with three state-of-the-art multi-task learning algorithms:

MTMR trains an individual learner on each task, and regularizes the trained learners by projecting them onto a low-dimensional manifold [1].

MTDA allows the tasks lie in different feature spaces and learns multiple mappings, each of which independently maps a task space to a shared latent space [42].

HEGS selects a subset of data examples from multiple tasks which have similar distributions, and then trains a unified learner on the selected data subset [33].

Moreover, to conduct the ablation study, we propose two variants of GPMT, named **GPMT-U**(niversal) and **GPMT-I**(ndependent), respectively. Their differences from GPMT are: (i) GPMT-U does not further embed the universal feature space into a lower-dimensioned space; instead, a unified learner is directly trained on the universal space. (ii) GPMT-I does not exploit the label correlations and simply learns independent classifiers to handle multiple labels.

Table 1: Characteristics of the studied datasets.

Dataset	#examples	#dim. 1*	#dim. 2*	#dim. 3*	Domain
CAL500	502	68	49	70	audio
emotions	593	72	66	81	audio
genbase	662	1, 186	1, 023	1, 352	biology
medical	978	1, 449	1, 161	1, 620	text
Enron	1, 702	1, 001	974	1, 111	text
yeast	2, 417	103	85	113	biology
Slashdot	3, 782	1, 079	981	1, 220	text
Corel5k	5, 000	499	352	507	image
delicious	16, 105	500	400	600	text
CCYS	79, 988	355 (overall in 8 tasks)			education

* The dimensions of the original, the first mapped, and the second mapped feature spaces, respectively.

6.1.3 Evaluation Metrics and Parameters

We evaluate the experimental performances using *example-based accuracy* and *one-error*, both of which are widely used in literature. Specifically, the example-based accuracy calibrates the proportion of the correctly predicted labels to the overall labels for each instance:

$$\text{Exam-Acc} = \frac{1}{n} \sum_{j=1}^{n} \frac{|\mathbf{y}_j \cap \hat{\mathbf{y}}_j|}{|\mathbf{y}_j|}, \tag{17}$$

and the one-error measures the frequency of the instances whose most confidently predicted label is not in the groundtruth label set:

$$\text{One-Error} = \frac{1}{n} \sum_{j=1}^{n} [\![\arg\max_{\hat{y} \in \hat{\mathbf{y}}_j} \text{rank}(\hat{y}) \notin \mathbf{y}_j]\!], \tag{18}$$

where $\text{rank}(\cdot)$ denote the function that ranks the predicted labels by their confidence. In this work, we use *max-margin* principle to quantify the prediction confidences.

For our approach, we use grid search to find the optimal parameter set based on the one-error for each dataset. In particular, λ_1 and λ_2 are selected from $\{.01, .03, .05, .1, .3, .5\}$ and $\{.001, .005, .01, .05, .1, .5\}$, respectively. We set $z = 500$ for genbase, medical, Enron, and Slashdot; and $z = 100$ for the other datasets. For the compared methods, parameters are set as suggested in the corresponding literatures.

6.2 Results

Table 2 reports the detailed results of performance comparison, from which we investigate the following two questions.

Q2 *Does the proposed GPMT approach outperform the state-of-the-art methods?*

GPMT shows a significant superiority over the compared methods on most datasets. It is worth noting that GPMT simply adopts a linear model as the implementation of the unified learner, which means that the performance gain can be expected by implementing a more powerful classifier such as SVM or multi-layer perceptron. The

Table 2: Comparison of GPMT with the state-of-the-arts under two evaluation metrices (Mean ± Standard Deviation). For each evaluation metric, ↑ indicates "the larger the better" while ↓ indicates "the smaller the better". The best performances are bold. ● indicates GPMT has a statistically significant better performance than the compared methods (hypothesis supported by *paired t-tests* at 95% significance level).

Dataset	Metric	HEGS	MTMR	MTDA	GPMT-U	GPMT-I	GPMT
CAL500	Exam-Acc ↑	.839 ± .001	.734 ± .003 ●	.775 ± .001	.815 ± .003	**.867 ± .001**	.823 ± .002
	One-Error ↓	.074 ± .001	.195 ± .004 ●	.113 ± .002 ●	.086 ± .001	**.041 ± .007**	.057 ± .001
emotions	Exam-Acc ↑	.619 ± .001 ●	.715 ± .005 ●	.762 ± .001 ●	.795 ± .006 ●	.817 ± .001	**.848 ± .001**
	One-Error ↓	.247 ± .003 ●	.155 ± .007 ●	.181 ± .001 ●	.139 ± .009 ●	.112 ± .002	**.098 ± .003**
genbase	Exam-Acc ↑	.536 ± .010 ●	.642 ± .005 ●	.668 ± .004 ●	.639 ± .003 ●	.687 ± .008 ●	**.755 ± .003**
	One-Error ↓	.290 ± .001 ●	.224 ± .004 ●	.202 ± .003 ●	.196 ± .002 ●	.201 ± .006 ●	**.107 ± .005**
medical	Exam-Acc ↑	.516 ± .009 ●	.722 ± .006	.697 ± .001 ●	.716 ± .002 ●	.683 ± .007 ●	**.763 ± .005**
	One-Error ↓	.279 ± .002 ●	.227 ± .009	.245 ± .002 ●	.214 ± .003	.263 ± .002 ●	**.197 ± .001**
Enron	Exam-Acc ↑	.503 ± .003 ●	.693 ± .005 ●	.761 ± .007	.712 ± .004 ●	.722 ± .003 ●	**.801 ± .004**
	One-Error ↓	.292 ± .017 ●	.201 ± .006 ●	.147 ± .001	.154 ± .006 ●	.129 ± .002	**.114 ± .002**
yeast	Exam-Acc ↑	.727 ± .002 ●	.738 ± .002 ●	.784 ± .002 ●	.793 ± .003	.762 ± .001 ●	**.827 ± .002**
	One-Error ↓	.188 ± .001 ●	.156 ± .003 ●	.119 ± .002 ●	.053 ± .012	.082 ± .003	**.046 ± .000**
Slashdot	Exam-Acc ↑	.486 ± .014 ●	.557 ± .001 ●	.620 ± .006 ●	.637 ± .017 ●	.682 ± .002	**.714 ± .001**
	One-Error ↓	.414 ± .007 ●	.329 ± .004 ●	.224 ± .003	.206 ± .001	.216 ± .009	**.191 ± .002**
Corel5k	Exam-Acc ↑	.616 ± .003 ●	.695 ± .009 ●	.733 ± .001 ●	.793 ± .002	**.820 ± .009**	.803 ± .001
	One-Error ↓	.303 ± .001 ●	.257 ± .006 ●	.172 ± .005 ●	.156 ± .003	**.091 ± .004**	.137 ± .007
delicious	Exam-Acc ↑	.543 ± .008 ●	.638 ± .005 ●	.611 ± .007 ●	.606 ± .003 ●	.612 ± .001 ●	**.718 ± .002**
	One-Error ↓	.289 ± .003 ●	.185 ± .012	.198 ± .003 ●	.202 ± .002 ●	.214 ± .002 ●	**.136 ± .001**
CCYS	Exam-Acc ↑	.573 ± .001 ●	.645 ± .004 ●	.708 ± .001 ●	.691 ± .008 ●	.613 ± .003 ●	**.782 ± .001**
	One-Error ↓	.333 ± .005 ●	.301 ± .002 ●	.132 ± .005	.264 ± .005 ●	.215 ± .002 ●	**.112 ± .006**
GPMT:	Exam-Acc ↑	9 / 0 / 1	9 / 1 / 0	8 / 2 / 0	7 / 3 / 0	6 / 2 / 2	—
	One-Error ↓	9 / 1 / 0	8 / 2 / 0	7 / 3 / 0	5 / 5 / 0	4 / 4 / 2	—

win/tie/loss counts of our approach versus the compared methods are summarized in the last row of Table 2.

Overall, we observe that GPMT significantly outperforms HEGS, MTMR, and MTDA by wining on most datasets. In particular, HEGS yields the worst prediction performances on most datasets, and is more sensitive to the dimension change of datasets than other algorithms. For instance, HEGS achieves a 51.0% example-based accuracy on four datasets with higher dimensions (*i.e.* genbase, medical, Enron, and Slashdot) on average while the number on the other six datasets with lower dimensions is 65.3% – a 28% performance degradation is suffered. The reason is that HEGS pads zeros to handle the missing data, in which the zeros may carry noisy information and skew the original data distributions. Such negative effect is escalated in datasets with high dimensions since, the higher the dimensionality, the larger the number of padded zeros. In this regard, GPMT performs more robustly on both high- and low-dimensional datasets by achieving 75.8% and 80% example-based accuracies, respectively – only 5.5% performance degradation is suffered.

The average performances of MTMR and MTDA are similar (*e.g.* example-based accuracies are 68% and 71%, respectively), which are better than HEGS yet worse than GPMT (78.3%). Both MTMR and MTDA have the corresponding mechanisms to build a latent space so as to capture the commonalities among different feature spaces. However, no matter mapping the classifiers onto manifold (MTMR) or individually embedding each feature space (MTDA), the correlations between pairs of tasks are not respected. On the contrary, GPMT constructs the latent feature space that harmonizes information from all tasks and exploits the label correlations, and thus achieves better learning performances.

Q3 *Does the unified learner trained on the latent space improve the learning performance?*

Comparing with its two variants, GPMT wins on less criteria (22 out of 40) and ties on more criteria (14 out of 40). GPMT-U performs better on the six low-dimensional datasets (ties 6 criteria comparing with GPMT) than on the four high-dimensional datasets (ties only 2 criterion comparing with GPMT). The reason is that GPMT-U directly learns from the universal feature space and the classifier suffers from high dimensionality, *e.g.* a $1186 + 1023 + 1352 = 3561$ dimensional classifier is trained on genbase. GPMT-I achieves the closest performance with GPMT among all compared methods – tying on 6 criteria and winning on 2 criteria in CAL500 and 2 criteria in Corel5k. The overall better performance of GPMT validates the effectiveness of building a unified learner with respect to the task correlations.

7 Conclusion

In this paper, we have explored a new problem in the context of multi-task learning – how to predict the semantic labels of multiple tasks when only partial data examples are presented. By utilizing the relatedness among features across tasks, we discover a shared latent feature space that harmonizes the information contained in various tasks. To ensure the correctness of the discovered latent space, we leverage the supervised information so that the data examples with the same labels are represented closely in the latent space while those with disparate labels are separated. The processes of space discovering and learner training are unified in a tri-objective optimization problem, and an efficient optimization strategy is then developed to solve it. Theoretical results showed that our approach leads to better learning performance with guarantees. We carried out extensive experiments and the results evidenced the effectiveness of our approach.

It is worth pointing out that the data examples of some tasks may contain noise in practice, however, our current method does not consider such noise. In the future, we plan to extend GPMT to handle noisy representations since they will negatively affect the accurateness of the recovery of missing information. One possible solution is to utilize the recent advances in compressed sensing [4, 35] to discover the manifold structure underlying the data examples in multiple tasks. The discovered manifold can then be used to filter the data examples that disobey it, such that the negative effects caused by noise could be mitigated.

Acknowledgements

We thank ECAI 2020 reviewers for their constructive feedback. This work is supported by the U.S. National Science Foundation (NSF) under grants IIS-1652107, IIS-1763620, and CCF-1750886.

REFERENCES

[1] Arvind Agarwal, Samuel Gerber, and Hal Daume, ‘Learning multiple tasks using manifold regularization’, in *NIPS*, pp. 46–54, (2010).

[2] Jonathan Baxter, ‘A model of inductive bias learning’, *Journal of artificial intelligence research*, **12**, 149–198, (2000).

[3] Matthew R Boutell, Jiebo Luo, Xipeng Shen, and Christopher M Brown, ‘Learning multi-label scene classification’, *Pattern recognition*, **37**(9), 1757–1771, (2004).

[4] Emmanuel J Candes and Yaniv Plan, ‘Matrix completion with noise’, *Proceedings of the IEEE*, **98**(6), 925–936, (2010).

[5] Rich Caruana, ‘Multitask learning’, *Machine learning*, **28**(1), 41–75, (1997).

[6] Nicolo Cesa-Bianchi and Gabor Lugosi, *Prediction, learning, and games*, Cambridge university press, 2006.

[7] Amanda Clare and Ross D King, ‘Knowledge discovery in multi-label phenotype data’, in *European Conference on Principles of Data Mining and Knowledge Discovery*, pp. 42–53. Springer, (2001).

[8] Wenyuan Dai, Yuqiang Chen, Gui-Rong Xue, Qiang Yang, and Yong Yu, ‘Translated learning: Transfer learning across different feature spaces’, in *NIPS*, pp. 353–360, (2009).

[9] Jean-Marc David, Jean-Paul Krivine, and Reid Simmons, *Second generation expert systems*, Springer Science & Business Media, 2012.

[10] Dheeru Dua and Efi Karra Taniskidou. UCI machine learning repository, 2017.

[11] André Elisseeff and Jason Weston, ‘A kernel method for multi-labelled classification’, in *NIPS*, pp. 681–687, (2002).

[12] Chih-Hai Fan, Jason L Speyer, and Christian R Jaensch, ‘Centralized and decentralized solutions of the linear-exponential-gaussian problem’, *IEEE Transactions on Automatic Control*, **39**(10), 1986–2003, (1994).

[13] Johannes Fürnkranz, Eyke Hüllermeier, Eneldo Loza Mencía, and Klaus Brinker, ‘Multilabel classification via calibrated label ranking’, *Machine learning*, **73**(2), 133–153, (2008).

[14] Lei Han and Yu Zhang, ‘Multi-stage multi-task learning with reduced rank’, in *AAAI*, (2016).

[15] Shaobo Han, Xuejun Liao, and Lawrence Carin, ‘Cross-domain multitask learning with latent probit models’, in *ICML*, (2012).

[16] Maayan Harel and Shie Mannor, ‘Learning from multiple outlooks’, in *ICML*, (2010).

[17] Yi He, Baijun Wu, Di Wu, Ege Beyazit, Sheng Chen, and Xindong Wu, ‘Online learning from capricious data streams: a generative approach’, in *IJCAI*, pp. 2491–2497, (2019).

[18] Geoffrey E Hinton and Ruslan R Salakhutdinov, ‘Reducing the dimensionality of data with neural networks’, *science*, **313**(5786), 504–507, (2006).

[19] Matthew D Hoffman, ‘Learning deep latent gaussian models with markov chain monte carlo’, in *ICML*, pp. 1510–1519, (2017).

[20] Chenping Hou, Changshui Zhang, Yi Wu, and Feiping Nie, ‘Multiple view semi-supervised dimensionality reduction’, *Pattern Recognition*, **43**(3), 720–730, (2010).

[21] Brian Kulis, Kate Saenko, and Trevor Darrell, ‘What you saw is not what you get: Domain adaptation using asymmetric kernel transforms’, in *CVPR*, pp. 1785–1792, (2011).

[22] Nada Lavrač, ‘Machine learning for data mining in medicine’, in *Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making*, pp. 47–62. Springer, (1999).

[23] Shao-Yuan Li, Yuan Jiang, and Zhi-Hua Zhou, ‘Partial multi-view clustering’, in *AAAI*, (2014).

[24] Xuejun Liao and Lawrence Carin, ‘Radial basis function network for multi-task learning’, in *NIPS*, pp. 792–802, (2006).

[25] Bao-Di Liu, Yu-Xiong Wang, Bin Shen, Yu-Jin Zhang, and Yan-Jiang Wang, ‘Blockwise coordinate descent schemes for sparse representation’, in *ICASSP*, pp. 5267–5271, (2014).

[26] Bo Long, Philip S Yu, and Zhongfei Zhang, ‘A general model for multiple view unsupervised learning’, in *SDM*, pp. 822–833. SIAM, (2008).

[27] Julie Nutini, Mark Schmidt, Issam Laradji, Michael Friedlander, and Hoyt Koepke, ‘Coordinate descent converges faster with the gauss-southwell rule than random selection’, in *ICML*, pp. 1632–1641, (2015).

[28] Peter Prettenhofer and Benno Stein, ‘Cross-language text classification using structural correspondence learning’, in *ACL*, pp. 1118–1127, (2010).

[29] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin, ‘Variational autoencoder for deep learning of images, labels and captions’, in *NIPS*, pp. 2352–2360, (2016).

[30] Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank, ‘Classifier chains for multi-label classification’, *Machine learning*, **85**(3), 333, (2011).

[31] Jesse Read, Bernhard Pfahringer, and Geoffrey Holmes, ‘Multi-label classification using ensembles of pruned sets’, in *ICDM*, pp. 995–1000, (2008).

[32] Xiaoxiao Shi, Qi Liu, Wei Fan, S Yu Philip, and Ruixin Zhu, ‘Transfer learning on heterogenous feature spaces via spectral transformation’, in *ICDM*, pp. 1049–1054, (2010).

[33] Xiaoxiao Shi, Qi Liu, Wei Fan, Qiang Yang, and Philip S Yu, ‘Predictive modeling with heterogeneous sources’, in *SDM*, pp. 814–825. SIAM, (2010).

[34] Daniel L Silver, Ryan Poirier, and Duane Currie, ‘Inductive transfer with context-sensitive neural networks’, *Machine Learning*, **73**(3), 313, (2008).

[35] Bart Vandereycken, ‘Low-rank matrix completion by riemannian optimization’, *SIAM Journal on Optimization*, **23**(2), 1214–1236, (2013).

[36] Chang Wang and Sridhar Mahadevan, ‘Heterogeneous domain adaptation using manifold alignment’, in *IJCAI*, (2011).

[37] Qian Xu and Qiang Yang, ‘A survey of transfer and multitask learning in bioinformatics’, *Journal of Computing Science and Engineering*, **5**(3), 257–268, (2011).

[38] Rong Yan, Jelena Tesic, and John R Smith, ‘Model-shared subspace boosting for multi-label classification’, in *SIGKDD*, pp. 834–843, (2007).

[39] Guoxian Yu, Xia Chen, Carlotta Domeniconi, Jun Wang, Zhao Li, Zili Zhang, and Xindong Wu, ‘Feature-induced partial multi-label learning’, in *ICDM*, pp. 1398–1403, (2018).

[40] Han Yuan, Ivan Paskov, Hristo Paskov, Alvaro J González, and Christina S Leslie, ‘Multitask learning improves prediction of cancer drug sensitivity’, *Scientific reports*, **6**, 31619, (2016).

[41] Min-Ling Zhang and Zhi-Hua Zhou, ‘Multi-label learning by instance differentiation’, in *AAAI*, volume 7, pp. 669–674, (2007).

[42] Yu Zhang and Dit-Yan Yeung, ‘Multi-task learning in heterogeneous feature spaces’, in *AAAI*, (2011).