A Survey on Multi-Task Learning

Yu Zhang and Qiang Yang

Abstract—Multi-Task Learning (MTL) is a learning paradigm in machine learning and its aim is to leverage useful information contained in multiple related tasks to help improve the generalization performance of all the tasks. In this paper, we give a survey for MTL. First, we classify different MTL algorithms into several categories, including feature learning approach, low-rank approach, task clustering approach, task relation learning approach, and decomposition approach, and then discuss the characteristics of each approach. In order to improve the performance of learning tasks further, MTL can be combined with other learning paradigms including semi-supervised learning, active learning, unsupervised learning, reinforcement learning, multi-view learning and graphical models. When the number of tasks is large or the data dimensionality is high, batch MTL models are difficult to handle this situation and online, parallel and distributed MTL models as well as dimensionality reduction and feature hashing are reviewed to reveal their computational and storage advantages. Many real-world applications use MTL to boost their performance and we review representative works. Finally, we present theoretical analyses and discuss several future directions for MTL.

Index Terms—Multi-Task Learning, Machine Learning, Artificial Intelligence

1 Introduction

Machine learning techniques usually require a large number of training samples to learn an accurate learner. For example, deep learning models, which build on neural networks, usually need millions of labeled samples to train neural networks with tens or even hundreds of layers which contain a huge number of model parameters. However, in some applications such as medical image analysis, this requirement cannot be fulfilled since (labeled) samples are hard to collect. In this case, limited training samples are not enough to learn shallow models, let alone deep models. For this data insufficient problem, Multi-Task Learning (MTL) [1] is a good solution when there are multiple related tasks each of which has limited training samples.

In MTL, there are multiple learning tasks each of which can be a general learning task such as supervised tasks (e.g., classification or regression problems), unsupervised tasks (e.g., clustering problems), semi-supervised tasks, reinforcement learning tasks, multi-view learning tasks or graphical models. Among these learning tasks, all of them or at least a subset of them are assumed to be related to each other. In this case, it is found that learning these tasks jointly can lead to much performance improvement compared with learning them individually. This observation leads to the birth of MTL. Hence MTL aims to improve the generalization performance of multiple tasks when they are related.

MTL is inspired by human learning activities where people often apply the knowledge learned from previous tasks to help learn a new task. For example, for a person who learns to ride the bicycle and tricycle together, the experience in learning to ride a bicycle can be utilized in riding a tricycle and vice versa. Similar to human learning, it is useful for multiple learning tasks to be learned jointly since the knowledge contained in a task can be leveraged by other tasks.

The setting of MTL is similar to that of transfer learning [2] but also they have significant difference. In MTL, there is no distinction among different tasks and the objective is to improve the performance of all the tasks. However, in transfer learning which is to improve the performance of a target task with the help of source tasks, the target task plays a more important role than source tasks. Hence, MTL treats all the tasks equally but in transfer learning the target task attracts most attentions among all the tasks. In [3], [4], [5], a new MTL setting called asymmetric multi-task learning is investigated and this setting considers a different scenario where a new task is arrived when multiple tasks have been learned jointly via some MTL method. A simple solution is to learn the old and new tasks together from scratch but it is computationally demanding. Instead the asymmetric multi-task learning only learns the new task with the help of old tasks and hence the core problem is how to transfer the knowledge contained in the old tasks to the new task. In this sense, this setting is more similar to transfer learning than to MTL.

In this paper, we give a survey on MTL. After giving a definition for MTL, we classify different MTL algorithms into several categories: feature learning approach which can be further categorized into feature transformation and feature selection approaches, low-rank approach, task clustering approach, task relation learning approach, and decomposition approach. We discuss the characteristics of each approach. MTL can be combined with other learning paradigms to further improve the performance of learning tasks and hence we discuss the combinations of MTL with other learning paradigms including semi-supervised learning, active learning, unsupervised learning, reinforcement learning, multi-view learning and graphical models. When the number of tasks is large, the number of training data in all the tasks can be very large, which makes the online and parallel computation of MTL models necessary. In this case, the training data of different tasks could locate in different machines and hence distributed MTL models are a good solution. Moreover, dimensionality reduction and feature hashing are vital tools to reduce the data dimension when facing high-dimensional data in MTL. Hence, we review those techniques that are helpful when handling big data in multiple tasks. As a general learning paradigm, MTL has many applications in various areas and here we briefly review its applications in computer vision, bioinformatics, health informatics, speech, natural language processing, web applications and ubiquitous computing. Besides algorithmic development and real-

- *Y. Zhang and Q. Yang are with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology.*
E-mail: {yuzhangcse,qyang}@cse.ust.hk.

world applications of MTL, we review theoretical analyses and discuss several future directions for MTL.

The remainder of this paper is organized as follows. Section 2 introduces several categories of MTL models. In Section 3, the combinations of MTL with other learning paradigms are reviewed. Section 4 overviews online, parallel, and distributed MTL models as well as dimensionality reduction and feature hashing. Section 5 presents the applications of MTL in various areas. Section 6 gives an overview on theoretical analyses and finally we make conclusions in Section 7 with some discussions on future directions in MTL.[1]

2 MTL Models

In order to fully characterize MTL, we first give the definition of MTL.

Definition 1. (**Multi-Task Learning**) Given m learning tasks $\{\mathcal{T}_i\}_{i=1}^{m}$ where all the tasks or a subset of them are related, *multi-task learning* aims to help improve the learning of a model for $\mathcal{T}_i$ by using the knowledge contained in all or some of the m tasks.

Based on the definition of MTL, we focus on supervised learning tasks in this section since most MTL studies fall in this setting. For other types of tasks, we review them in the next section. In the setting of supervised learning tasks, usually a task $\mathcal{T}_i$ is accompanied by a training dataset $\mathcal{D}_i$ consisting of n_i training samples, i.e., $\mathcal{D}_i = \{\mathbf{x}_j^i, y_j^i\}_{j=1}^{n_i}$, where $\mathbf{x}_j^i \in \mathbb{R}^{d_i}$ is the jth training instance in $\mathcal{T}_i$ and y_j^i is its label. We denote by $\mathbf{X}^i$ the training data matrix for $\mathcal{T}_i$, i.e., $\mathbf{X}^i = (\mathbf{x}_1^i, \ldots, \mathbf{x}_{n_i}^i)$. When different tasks share the same training data samples, i.e., $\mathbf{X}^i = \mathbf{X}^j$ for $i \neq j$, MTL reduces to multi-label learning or multi-output regression. Here we consider a general setting for MTL that at least two out of all the $\mathbf{X}^i$'s are different or a more general setting that all the $\mathbf{X}^i$'s are different from each other. When different tasks lie in the same feature space implying that d_i equals d_j for any $i \neq j$, this setting is the homogeneous-feature MTL, and otherwise it corresponds to heterogeneous-feature MTL. Without special explanation, the default MTL setting is the homogeneous-feature MTL. Here we need to distinguish the heterogeneous-feature MTL from the heterogeneous MTL. In [7], the heterogeneous MTL is considered to consist of different types of supervised tasks including classification and regression problems, and here we generalize it to a more general setting that the heterogeneous MTL consists of tasks with different types including supervised learning, unsupervised learning, semi-supervised learning, reinforcement learning, multi-view learning and graphical models. The opposite to the heterogeneous MTL is the homogeneous MTL which consist of tasks with only one type. In a word, the homogeneous and heterogeneous MTL differ in the type of learning tasks while the homogeneous-feature MTL is different from the heterogeneous-feature MTL in terms of the original feature representations. Similarly, without special explanation, the default MTL setting is the homogeneous MTL.

In order to characterize the relatedness in the definition of MTL, there are three issues to be addressed: when to share, what to share, and how to share.

The 'when to share' issue is to make choices between single-task and multi-task models for a multi-task problem. Currently such decision is made by human experts and there are few learning approaches to study it. A simple computational solution is to formulated such decision as a model selection problem and then use model selection techniques, e.g., cross validation, to make decisions, but this solution is usually computational heavy and may require much more training data. Another solution is to use multi-task models which can degenerate to the single-task counterparts, for example, problem (34) where the learning of different tasks can be decoupled when $\mathbf{\Sigma}$ becomes diagonal. In this case, we can let the training data determine the form of $\mathbf{\Sigma}$ to make an implicit choice.

'What to share' needs to determine the form through which knowledge sharing among all the tasks could occur. Usually, there are three forms for 'what to share', including feature, instance and parameter. Feature-based MTL aims to learn common features among different tasks as a way to share knowledge. Instance-based MTL wants to identify useful data instances in a task for other tasks and then shares knowledge via the identified instances. Parameter-based MTL uses model parameters (e.g., coefficients in linear models) in a task to help learn model parameters in other tasks in some ways, for example, the regularization. Existing MTL studies mainly focus on feature-based and parameter-based methods and few works belong to the instance-based method. A representative instance-based method is the multi-task distribution matching method proposed in [8], which first estimates density ratios between probabilities that each instance as well as its label belongs to both its own task and a mixture of all the tasks and then uses all the weighted training data from all the tasks based on the estimated density ratios to learn model parameters for each task. Since the studies on instance-based MTL are few, we mainly review feature-based and parameter-based MTL models.

After determining 'what to share', 'how to share' specifies concrete ways to share knowledge among tasks. In feature-based MTL, there is a primary approach: feature learning approach. The feature learning approach focuses on learning common feature representations for multiple tasks based on shallow or deep models, where the learned common feature representation can be a subset or a transformation of the original feature representation.

In parameter-based MTL, there are four main approaches: low-rank approach, task clustering approach, task relation learning approach, and decomposition approach. The low-rank approach interprets the relatedness of multiple tasks as the low rankness of the parameter matrix of these tasks. The task clustering approach assumes that all the tasks form a few clusters where tasks in a cluster are related to each other. The task relation learning approach aims to learn quantitative relations between tasks from data automatically. The decomposition approach decomposes the model parameters of all the tasks into two or more components, which are penalized by different regularizers.

In summary, there are mainly five approaches in the feature-based and parameter-based MTL. In the following sections, we review these approaches in a chronological order to reveal the relations and evolutions among different models in them.

2.1 Feature Learning Approach

Since tasks are related, it is intuitive to assume that different tasks share a common feature representation based on the original features. One reason to learn common feature representations instead of directly using the original ones is that the original representation may not have enough expressive power for multiple tasks. With the training data in all the tasks, a more powerful representation can

[1] For an introduction to MTL without technical details, please refer to [6].

be learned for all the tasks and this representation can bring the improvement on the performance.

Based on the relationship between the original feature representation and the learned one, we can further classify this category into two sub-categories. The first sub-category is the feature transformation approach where the learned representation is a linear or nonlinear transformation of the original representation and in this approach, each feature in the learned representation is different from original features. Different from this approach, the feature selection approach, the second sub-category, selects a subset of the original features as the learned representation and hence the learned representation is similar to the original one by eliminating useless features based on different criteria. In the following, we introduce these two approaches.

2.1.1 Feature Transformation Approach

The multi-layer feedforward neural network [1], which belongs to the feature transformation approach, is one of the earliest model for multi-task learning. To see how the multi-layer feedforward neural network is constructed for MTL, in Figure 1 we show an example with an input layer, a hidden layer, and an output layer. The input layer receives training instances from all the tasks and the output layer has m output units with one for each task. Here the outputs of the hidden layer can be viewed as the common feature representation learned for the m tasks and the transformation from the original representation to the learned one depends on the weights connecting the input and hidden layers as well as the activation function adopted in the hidden units. Hence, if the activation function in the hidden layer is linear, then the transformation is a linear function and otherwise it is nonlinear. Compared with multi-layer feedforward neural networks used for single-task learning, the difference in the network architecture lies in the output layers where in single-task learning, there is only one output unit while in MTL, there are m ones. In [9], the radial basis function network, which has only one hidden layer, is extended to MTL by greedily determining the structure of the hidden layer. Different from these neural network models, Silver et al. [10] propose a context-sensitive multi-task neural network which has only one output unit shared by different tasks but has a task-specific context as an additional input.

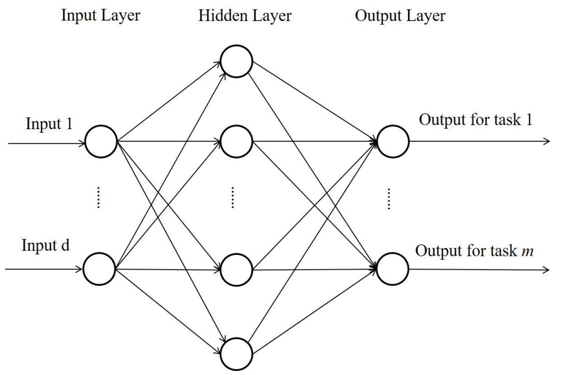

Fig. 1. An example for the multi-task feedforward neural network with an input layer, a hidden layer, and an output layer.

Different from multi-layer feedforward neural networks which are connectionist models, the multi-task feature learning (MTFL) method [11], [12] is formulated under the regularization framework with the objective function as

$$\min_{\mathbf{A},\mathbf{U},\mathbf{b}} \sum_{i=1}^{m}\sum_{j=1}^{n_i} l(y_j^i, (\mathbf{a}^i)^T\mathbf{U}^T\mathbf{x}_j^i + b_i) + \lambda\|\mathbf{A}\|_{2,1}^2 \text{ s.t. } \mathbf{U}\mathbf{U}^T = \mathbf{I}, \quad (1)$$

where $l(\cdot,\cdot)$ denotes a loss function such as the hinge loss or square loss, $\mathbf{b} = (b_1, \ldots, b_m)^T$ is a vector of offsets in all the tasks, $\mathbf{U} \in \mathbb{R}^{d\times d}$ is a square transformation matrix, $\mathbf{a}^i$, the ith column in $\mathbf{A}$, contains model parameters for the ith task after the transformation, the $\ell_{2,1}$ norm of a matrix $\mathbf{A}$ denoted by $\|\mathbf{A}\|_{2,1}$ equals the sum of the ℓ_2 norm of rows in $\mathbf{A}$, $\mathbf{I}$ denotes an identity matrix with an appropriate size, and λ is a positive regularization parameter. The first term in the objective function of problem (1) measures the empirical loss on the training sets of all the tasks, the second one is to enforce $\mathbf{A}$ to be row-sparse via the $\ell_{2,1}$ norm which is equivalent to selecting features after the transformation, and the constraint enforces $\mathbf{U}$ to be orthogonal. Different from the multi-layer feedforward neural network whose hidden representations may be redundant, the orthogonality of $\mathbf{U}$ can prevent the MTFL method from it. It is interesting to find out that problem (1) is equivalent to

$$\min_{\mathbf{W},\mathbf{D},\mathbf{b}} L(\mathbf{W},\mathbf{b}) + \lambda\text{tr}(\mathbf{W}^T\mathbf{D}^{-1}\mathbf{W}) \text{ s.t. } \mathbf{D} \succeq \mathbf{0},\ \text{tr}(\mathbf{D}) \leq 1, \quad (2)$$

where $L(\mathbf{W},\mathbf{b}) = \sum_{i=1}^{m}\sum_{j=1}^{n_i} l(y_j^i, (\mathbf{w}^i)^T\mathbf{x}_j^i + b_i)$ denotes the total training loss, $\text{tr}(\cdot)$ denotes the trace of a square matrix, $\mathbf{w}^i = \mathbf{U}\mathbf{a}^i$ is the model parameter for $\mathcal{T}_i$, $\mathbf{W} = (\mathbf{w}^1, \ldots, \mathbf{w}^m)$, $\mathbf{0}$ denotes a zero vector or matrix with an appropriate size, $\mathbf{M}^{-1}$ for any square matrix $\mathbf{M}$ denotes its inverse when it is nonsingular or otherwise its pseudo inverse, and $\mathbf{B} \succeq \mathbf{C}$ means that $\mathbf{B} - \mathbf{C}$ is positive semidefinite. Based on this formulation, we can see that the MTFL method is to learn the feature covariance $\mathbf{D}$ for all the tasks, which will be interpreted in Section 2.8 from a probabilistic perspective. Given $\mathbf{D}$, the learning of different tasks can be decoupled and this can facilitate the parallel computing. When given $\mathbf{W}$, $\mathbf{D}$ has an analytical solution as $\mathbf{D} = (\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}/\text{tr}\left((\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}\right)$ and by plugging this solution into problem (2), we can see that the regularizer on $\mathbf{W}$ is the squared trace norm. Then Argyriou et al. [13] extend problem (2) to a general formulation where the second term in the objective function becomes $\lambda\text{tr}(\mathbf{W}^T f(\mathbf{D})\mathbf{W})$ with $f(\mathbf{D})$ operating on the spectrum of $\mathbf{D}$ and discuss the condition on $f(\cdot)$ to make the whole problem convex.

Similar to the MTFL method, the multi-task sparse coding method [14] is to learn a linear transformation on features with the objective function formulated as

$$\min_{\mathbf{A},\mathbf{U},\mathbf{b}} L(\mathbf{U}\mathbf{A},\mathbf{b}) \text{ s.t. } \|\mathbf{a}^i\|_1 \leq \lambda\ \forall i \in [m], \|\mathbf{u}^j\|_2 \leq 1\ \forall j \in [D], \quad (3)$$

where $\mathbf{u}^j$ is the jth column in $\mathbf{U}$, $[a]$ for an integer a denotes a set of integers from 1 to a, $\|\cdot\|_1$ denotes the ℓ_1 norm of a vector or matrix and equals the sum of the absolute value of its entries, and $\|\cdot\|_2$ denotes the ℓ_2 norm of a vector. Here the transformation $\mathbf{U} \in \mathbb{R}^{d\times D}$ is also called the dictionary in sparse coding and shared by all the tasks. Compared with the MTFL method where $\mathbf{U}$ in problem (1) is a $d \times d$ orthogonal matrix, $\mathbf{U}$ in problem (3) is overcomplete, which implies that D is larger than d, with each column having a bounded ℓ_2 norm. Another difference is that in problem (1) $\mathbf{A}$ is enforced to be row-sparse but in problem (3) it is only sparse via the first constraint. With a similar idea to the multi-task sparse coding method, Zhu et al. [15] propose a multi-task infinite support vector machine via the Indian buffet process and the difference is that in [15] the dictionary is sparse and model parameters are non-sparse. In [16], the spike and slab prior is used to learn sparse model parameters for multi-output regression problems where transformed features are induced by Gaussian processes and shared by different outputs.

Recently deep learning becomes popular due to its capacity to learn nonlinear features in many applications and deep models have been used as basic models in MTL. Different from the aforementioned models which are shallow since there is only one level of the feature transformation, there are some deep MTL models which can have many layers of the feature transformations. For example, similar to the multi-task neural network shown in Fig. 1, many deep MTL methods [17], [18], [19], [20], [21] assume that different tasks share the first several hidden layers and then have task-specific parameters in the subsequent layers. Different from these deep MTL methods, a more advanced way is to use a learning-based approach to determine the inputs of hidden layers in different tasks, e.g., the cross-stitch network proposed in [22] to learn task relations in terms of the hidden feature representation, which is a bit similar to the task relation learning approach introduced later. Specifically, given two tasks A and B with an identical network architecture, $x_A^{i,j}$ ($x_B^{i,j}$) denotes the hidden feature outputted by the jth unit of the ith hidden layer for task A (B). Then we can define the cross-stitch operation on $x_A^{i,j}$ and $x_B^{i,j}$ as $\begin{pmatrix} \tilde{x}_A^{i,j} \\ \tilde{x}_B^{i,j} \end{pmatrix} = \begin{pmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{pmatrix} \begin{pmatrix} x_A^{i,j} \\ x_B^{i,j} \end{pmatrix}$, where $\tilde{x}_A^{i,j}$ and $\tilde{x}_B^{i,j}$ are new hidden features after learning the two tasks jointly. When both α_{AB} and α_{BA} equal 0, training the two networks jointly is equivalent to training them independently. The network architecture of the cross-stitch network is shown in Fig. 2. Here matrix $\boldsymbol{\alpha} = \begin{pmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{pmatrix}$ encodes the task relations between the two tasks and it can be learned via the backpropagation method. Different from the task relation learning approach whose task relations are defined based on the model parameters, $\boldsymbol{\alpha}$ is based on hidden features. Moreover, the adversarial learning is applied to learn common features for MTL in [23].

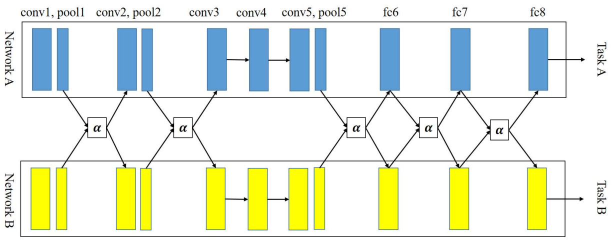

Fig. 2. The architecture for the cross-stitch network.

2.1.2 Feature Selection Approach

One way to do feature selection in MTL is to use the group sparsity based on the $\ell_{p,q}$ norm denoted by $\|\mathbf{W}\|_{p,q}$, which is equal to $\|(\|\mathbf{w}_1\|_p, \ldots, \|\mathbf{w}_d\|_p)\|_q$, where $\mathbf{w}_i$ denotes the ith row of $\mathbf{W}$ and $\|\cdot\|_p$ denotes the ℓ_p norm of a vector. Obozinski et al. [24], [25] are among the first to study the multi-task feature selection (MTFS) problem based on the $\ell_{2,1}$ norm with the objective function formulated as

$$\min_{\mathbf{W},\mathbf{b}} \; L(\mathbf{W},\mathbf{b}) + \lambda\|\mathbf{W}\|_{2,1}. \tag{4}$$

The regularizer on $\mathbf{W}$ in problem (4) is to enforce $\mathbf{W}$ to be row-sparse, which in turn helps select important features. In [24], [25], a path-following algorithm is proposed to solve problem (4) and then Liu et al. [26] employ an optimal first-order optimization method to solve it. Compared with problem (1), we can see that problem (4) is similar to the MTFL method without learning the transformation $\mathbf{U}$. Lee et al. [27] propose a weighted $\ell_{2,1}$ norm for multi-task feature selection where the weights can be learned as well and problem (4) is extended in [28] to a general case where feature groups can overlap with each other. In order to make problem (4) more robust to outliers, a square-root loss function is investigated in [29]. Moreover, in order to make speedup, a safe screening method is proposed in [30] to filter out useless features corresponding to zero rows in $\mathbf{W}$ before optimizing problem (4).

Liu et al. [31] propose to use the $\ell_{\infty,1}$ norm to select features with the objective function formulated as

$$\min_{\mathbf{W},\mathbf{b}} \; L(\mathbf{W},\mathbf{b}) + \lambda\|\mathbf{W}\|_{\infty,1}. \tag{5}$$

A block coordinate descent method is proposed to solve problem (5). In general, we can use the $\ell_{p,q}$ norm to select features for MTL and the objective function is formulated as

$$\min_{\mathbf{W},\mathbf{b}} \; L(\mathbf{W},\mathbf{b}) + \lambda\|\mathbf{W}\|_{p,q}. \tag{6}$$

In order to keep the convexity of problem (6), it is required that $p > 1$ and $q \geq 1$. For the optimization of problem (6), Vogt and Roth [32] propose an active set algorithm to solve the $\ell_{p,1}$ norm regularization efficiently for arbitrary p.

In order to attain a more sparse subset of features, Gong et al. [33] propose a capped-$\ell_{p,1}$ penalty for multi-task feature selection where $p = 1$ or 2 and the objective function is formulated as

$$\min_{\mathbf{W},\mathbf{b}} \; L(\mathbf{W},\mathbf{b}) + \lambda \sum_{i=1}^{d} \min(\|\mathbf{w}_i\|_p, \theta), \tag{7}$$

where $\mathbf{w}_i$ denotes the ith row of $\mathbf{W}$. With the given threshold θ, the capped-$\ell_{p,1}$ penalty (i.e., the second term in problem (7)) focuses on rows with smaller ℓ_p norms than θ, which is more likely to be sparse. When θ becomes large enough, the second term in problem (7) becomes $\|\mathbf{W}\|_{p,1}$ and hence problem (7) degenerates to problem (4) or (5) when p equals 2 or ∞.

Lozano and Swirszcz [34] propose a multi-level Lasso for MTL where the (j,i)th entry in the parameter matrix $\mathbf{W}$ is defined as $w_{ji} = \theta_j \hat{w}_{ji}$. When θ_j is equal to 0, w_{ji} becomes 0 for $i \in [m]$ and hence the jth feature is not selected by the model. In this sense, θ_j controls the global sparsity for the jth feature among the m tasks. Moreover, when $\hat{w}_{ji}$ becomes 0, w_{ji} is also 0 for i only, implying that the jth feature is not useful for task $\mathcal{T}_i$, and so $\hat{w}_{ji}$ is a local indicator for the sparsity in task $\mathcal{T}_j$. Based on these observations, θ_j and $\hat{w}_{ji}$ are expected to be sparse, leading to the objective function formulated as

$$\min_{\boldsymbol{\theta},\hat{\mathbf{W}},\mathbf{b}} L(\mathbf{W},\mathbf{b}) + \lambda_1\|\boldsymbol{\theta}\|_1 + \lambda_2\|\hat{\mathbf{W}}\|_1 \text{ s.t. } w_{ji} = \theta_j \hat{w}_{ji}, \theta_j \geq 0, \tag{8}$$

where $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_d)^T$, $\hat{\mathbf{W}} = (\hat{\mathbf{w}}^1, \ldots, \hat{\mathbf{w}}^m)$, and the nonnegative constraint on θ_j is to keep the model identifiability. It has been proven in [34] that problem (8) leads to a regularizer $\sum_{j=1}^d \sqrt{\|\mathbf{w}_j\|_1}$, the square root of the $\ell_{1,\frac{1}{2}}$ norm regularization. Moreover, Wang et al. [35] extend problem (8) to a general situation where the regularizer becomes $\lambda_1 \sum_{i=1}^m \|\hat{\mathbf{w}}^i\|_p^p + \lambda_2\|\boldsymbol{\theta}\|_q^q$. By utilizing a priori information describing the task relations in a hierarchical structure, Han et al. [36] propose a multi-component product based decomposition for w_{ij} where the number of components in the decomposition can be arbitrary instead of only 2 in [34], [35]. Similar to [34], Jebara [37], [38] proposes to learn a binary indicator vector to do multi-task feature selection based on the maximum entropy discrimination formalism.

Similar to [36] where a priori information is given to describe task relations in a hierarchical/tree structure, Kim and Xing [39] utilize the given tree structure to design a regularizer on $\mathbf{W}$ as $f(\mathbf{W}) = \sum_{i=1}^{d} \sum_{v \in V} \lambda_v \|\mathbf{w}_{i,G_v}\|_2$, where V denotes the set of nodes in the given tree structure, G_v denotes the set of leaf nodes (i.e., tasks) in a sub-tree rooted at node v, and $\mathbf{w}_{i,G_v}$ denotes a subvector of the ith row of $\mathbf{W}$ indexed by G_v. This regularizer not only enforces each row of $\mathbf{W}$ to be sparse as the $\ell_{2,1}$ norm did in problem (4), but also induces sparsity in subsets of each row in $\mathbf{W}$ based on the tree structure.

Different from conventional multi-task feature selection methods which assume that different tasks share a set of original features, Zhou et al. [40] consider a different scenario where useful features in different tasks have no overlapping. In order to achieve this, an exclusive Lasso model is proposed with the objective function formulated as $\min_{\mathbf{W},\mathbf{b}} \ L(\mathbf{W}, \mathbf{b}) + \lambda \|\mathbf{W}\|_{1,2}^2$, where the regularizer is the squared $\ell_{1,2}$ norm on $\mathbf{W}$.

Another way to select common features for MTL is to use sparse priors to design probabilistic or Bayesian models. For $\ell_{p,1}$-regularized multi-task feature selection, Zhang et al. [41] propose a probabilistic interpretation where the $\ell_{p,1}$ regularizer corresponds to a prior: $w_{ji} \sim \mathcal{GN}(0, \rho_j, p)$, where $\mathcal{GN}(\cdot, \cdot, \cdot)$ denotes the generalized normal distribution. Based on this interpretation, Zhang et al. [41] further propose a probabilistic framework for multi-task feature selection, in which task relations and outlier tasks can be identified, based on the matrix-variate generalized normal prior.

In [42], a generalized horseshoe prior is proposed to do feature selection for MTL as:

$$\mathbb{P}(\mathbf{w}^i) = \int \prod_{j=1}^{d} \mathcal{N}(w_{ji}|0, \frac{u_{ji}}{v_{ji}}) \mathcal{N}(\mathbf{u}^i|0, \rho^2\mathbf{C}) \mathcal{N}(\mathbf{v}^i|0, \gamma^2\mathbf{C}) \mathrm{d}\mathbf{u}^i \mathrm{d}\mathbf{v}^i,$$

where $\mathcal{N}(\mathbf{m}, \boldsymbol{\sigma})$ denotes a univariate or multivariate normal distribution with $\mathbf{m}$ as the mean and $\boldsymbol{\sigma}$ as the variance or covariance matrix, u_{ji} and v_{ji} are the jth entries in $\mathbf{u}^i$ and $\mathbf{v}^i$, respectively, and ρ, γ are hyperparameters. Here $\mathbf{C}$ shared by all the tasks denotes the feature correlation matrix to be learned from data and it encodes an assumption that different tasks share identical feature correlations. When $\mathbf{C}$ becomes an identity matrix which means that features are independent, this prior degenerates to the horseshoe prior which can induce sparse estimations.

Hernández-Lobato et al. [43] propose a probabilistic model based on the horseshoe prior as:

$$\begin{aligned} \mathbb{P}(w_{ji}) \quad = \quad & \left[\pi(w_{ji})^{\eta_{ji}} \delta_0^{1-\eta_{ji}}\right]^{z_j} \left[\pi(w_{ji})^{\tau_{ji}} \delta_0^{1-\tau_{ji}}\right]^{\omega_i(1-z_j)} \\ & \left[\pi(w_{ji})^{\gamma_j} \delta_0^{1-\gamma_j}\right]^{(1-\omega_i)(1-z_j)}, \end{aligned} \tag{9}$$

where δ_0 is the probability mass function at zero and $\pi(\cdot)$ denotes the density function of non-zero coefficients. In Eq. (9), z_j indicates whether feature j is an outlier ($z_j = 1$) or not ($z_j = 0$) and ω_i indicates whether task $\mathcal{T}_i$ is an outlier ($\omega_i = 1$) or not ($\omega_i = 0$). Moreover, η_{ji} and τ_{ji} indicate whether feature j is relevant for the prediction in $\mathcal{T}_i$ ($\eta_{ji}, \tau_{ji} = 1$) or not ($\eta_{ji}, \tau_{ji} = 0$), and γ_j indicates whether the non-outlier feature j is relevant ($\gamma_j = 1$) for the prediction or not ($\gamma_j = 0$) in all non-outlier tasks. Based on the above definitions, the three terms in the right-hand side of Eq. (9) specify probability density functions of w_{ji} based on different situations of features and tasks. So this model can also handle outlier tasks but in a way different from [41].

2.1.3 *Comparison between Two Sub-categories*

The two sub-categories have different characteristics where the feature transformation approach learns a transformation of the original features as the new representation but the feature selection approach selects a subset of the original features as the new representation for all the tasks. Based on the characteristics of those two approaches, the feature selection approach can be viewed as a special case of the feature transformation approach when the transformation matrix is a diagonal $0/1$ matrix where the diagonal entries with value 1 correspond to the selected features. From this perspective, the feature transformation approach usually can fit the training data better than the feature selection approach since it has more capacity and hence if there is no overfitting when using the feature transformation approach, its generalization performance will have a certain probability to be better than that of the feature selection approach. On the other hand, by selecting a subset of the original features as the new representation, the feature selection approach has a better interpretability. In a word, if an application needs better performance, the feature transformation approach is more preferred and if the application needs some decision support, the feature selection approach may be the first choice.

2.2 Low-Rank Approach

The relatedness among multiple tasks can imply the low-rank of $\mathbf{W}$, leading to the low-rank approach.

Ando and Zhang [44] assume that the model parameters of different tasks share a low-rank subspace in part and more specifically, $\mathbf{w}^i$ takes the following form as

$$\mathbf{w}^i = \mathbf{u}^i + \boldsymbol{\Theta}^T \mathbf{v}^i. \tag{10}$$

Here $\boldsymbol{\Theta} \in \mathbb{R}^{h \times d}$ is the shared low-rank subspace by multiple tasks where $h < d$. Then we can write in a matrix form as $\mathbf{W} = \mathbf{U} + \boldsymbol{\Theta}^T \mathbf{V}$. Based on the form of $\mathbf{W}$, the objective function proposed in [44] is formulated as

$$\min_{\mathbf{U},\mathbf{V},\boldsymbol{\Theta},\mathbf{b}} L(\mathbf{U} + \boldsymbol{\Theta}^T \mathbf{V}, \mathbf{b}) + \lambda \|\mathbf{U}\|_F^2 \ \text{ s.t. } \boldsymbol{\Theta}\boldsymbol{\Theta}^T = \mathbf{I}, \tag{11}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. The orthonormal constraint on $\boldsymbol{\Theta}$ in problem (11) makes the subspace non-redundant. When λ is large enough, the optimal $\mathbf{U}$ can become a zero matrix and hence problem (11) is very similar to problem (1) except that there is no regularization on $\mathbf{V}$ in problem (11) and that $\boldsymbol{\Theta}$ has a smaller number of rows than columns. Chen et al. [45] generalize problem (11) as

$$\begin{aligned} \min_{\mathbf{U},\mathbf{V},\boldsymbol{\Theta},\mathbf{b}} \quad & L(\mathbf{U} + \boldsymbol{\Theta}^T \mathbf{V}, \mathbf{b}) + \lambda_1 \|\mathbf{U}\|_F^2 + \lambda_2 \|\mathbf{W}\|_F^2 \\ \text{s.t.} \quad & \boldsymbol{\Theta}\boldsymbol{\Theta}^T = \mathbf{I}. \end{aligned} \tag{12}$$

When setting λ_2 to be 0, problem (12) reduces to problem (11). Even though problem (12) is non-convex, with some convex relaxation technique, it can be relaxed to the following convex problem as

$$\min_{\mathbf{W},\mathbf{b},\mathbf{M}} L(\mathbf{W}, \mathbf{b}) + \lambda \mathrm{tr}\Big(\mathbf{W}^T \left(\mathbf{M} + \eta \mathbf{I}\right)^{-1} \mathbf{W}\Big) \text{ s.t. } \begin{matrix} \mathrm{tr}(\mathbf{M}) = h \\ \mathbf{0} \preceq \mathbf{M} \preceq \mathbf{I} \end{matrix}, \tag{13}$$

where $\eta = \lambda_2/\lambda_1$ and $\lambda = \lambda_1 \eta(\eta + 1)$. One advantage of problem (13) over problem (12) is that the global optimum of the convex problem (13) is much easier to be obtained than that of the non-convex problem (12). Compare with the alternative objective function (2) in the MTFL method, problem (13) has a similar formulation where $\mathbf{M}$ models the feature covariance for all the tasks. Problem (11) is extended in [46] to a general case where different $\mathbf{w}^i$'s lie in a manifold instead of a subspace. Moreover, in [47], [48], a latent variable model is proposed for $\mathbf{W}$ with the same decomposition as Eq. (10) and it can provide a framework for MTL by modeling more cases than problem (11) such as task clustering, sharing sparse representation, duplicate tasks and evolving tasks.

It is well known that using the trace norm as a regularizer can make a matrix have low rank and hence this regularization is suitable for MTL. Specifically, an objective function with the trace norm regularization is proposed in [49] as

$$\min_{\mathbf{W},\mathbf{b}} \; L(\mathbf{W},\mathbf{b}) + \lambda\|\mathbf{W}\|_{S(1)}, \tag{14}$$

where $\mu_i(\mathbf{W})$ denotes the ith smallest singular value of $\mathbf{W}$ and $\|\mathbf{W}\|_{S(1)} = \sum_{i=1}^{\min(m,d)} \mu_i(\mathbf{W})$ denotes the trace norm of matrix $\mathbf{W}$. Based on the trace norm, Han and Zhang [50] propose a variant called the capped trace regularizer with the objective function formulated as

$$\min_{\mathbf{W},\mathbf{b}} L(\mathbf{W},\mathbf{b}) + \lambda \sum_{i=1}^{\min(m,d)} \min(\mu_i(\mathbf{W}),\theta). \tag{15}$$

With the use of the threshold θ, the capped trace regularizer only penalizes small singular values of $\mathbf{W}$, which is related to the determination of the rank of $\mathbf{W}$. When θ is large enough, the capped trace regularizer will become the trace norm and hence in this situation, problem (15) reduces to problem (14). Moreover, a spectral k-support norm is proposed in [51] as an improvement over the trace norm regularization.

The trace norm regularization has been extended to regularize model parameters in deep learning models. Specifically, the weights in the last several fully connected layers of deep feedforward neural networks can be viewed as the parameters of different learners of all the tasks. In this view, the weights connecting two consecutive layers for one task can be organized in a matrix and hence the weights of all the tasks can form a tensor. Based on such tensor representations, several tensor trace norms that are based on the matrix trace norm [49], are used in [52] as regularizers to identify the low-rank structure of the parameter tensor.

2.3 Task Clustering Approach

The task clustering approach assumes that different tasks form several clusters each of which consists of similar tasks. As indicated by its name, this approach has a close connection to clustering algorithms and it can be viewed an extension of clustering algorithms to the task level while the conventional clustering algorithms are on the data level.

Thrun and Sullivan [53] propose the first task clustering algorithm by using a weighted nearest neighbor classifier for each task, where the initial weights to define the weighted Euclidean distance are learned by minimizing pairwise within-class distances and maximizing pairwise between-class distances simultaneously within each task. Then they define a task transfer matrix $\mathbf{A}$ whose (i,j)th entry a_{ij} records the generalization accuracy obtained for task $\mathcal{T}_i$ by using task $\mathcal{T}_j$'s distance metric via the cross validation. Based on $\mathbf{A}$, m tasks can be grouped into r clusters $\{\mathcal{C}_i\}_{i=1}^r$ by maximizing $\sum_{t=1}^r \frac{1}{|\mathcal{C}_t|}\sum_{i,j\in\mathcal{C}_t} a_{ij}$, where $|\cdot|$ denotes the cardinality of a set. After obtaining the cluster structure among all the tasks, the training data of tasks in a cluster will be pooled together to learn the final weighted nearest neighbor classifier. This approach has been extended to an iterative process in [54] in a way similar to k-means clustering.

Bakker and Heskes [55] propose a multi-task Bayesian neural network model with the network structure similar to Fig. 1 where input-to-hidden weights are shared by all the tasks but hidden-to-output weights are task-specific. By defining $\mathbf{w}^i$ as the vector of hidden-to-output weights for task $\mathcal{T}_i$, the multi-task Bayesian neural network assigns a mixture of Gaussian prior to it: $\mathbf{w}^i \sim \sum_{j=1}^r \pi_j \mathcal{N}(\mathbf{m}_j, \mathbf{\Sigma}_j)$, where π_j, $\mathbf{m}_j$ and $\mathbf{\Sigma}_j$ specify the prior, the mean and the covariance in the jth cluster. For tasks in a cluster, they will share a Gaussian distribution. When r equals 1, this model degenerates to a case where model parameters of different tasks share a prior, which is similar to several Bayesian MTL models such as [56], [57], [58] that are based on Gaussian processes and t processes.

Xue et al. [3] deploy the Dirichlet process to do clustering on task level. Specifically, it defines the prior on $\mathbf{w}^i$ as

$$\mathbf{w}^i \sim G, \; G \sim \mathcal{DP}(\alpha, G_0) \; \forall i \in [m],$$

where $\mathcal{DP}(\alpha, G_0)$ denotes a Dirichlet process with α as a positive scaling parameter and G_0 a base distribution. To see the clustering effect, by integrating out G, the conditional distribution of $\mathbf{w}^i$, given model parameters of other tasks $\mathbf{W}_{-i} = \{\cdots, \mathbf{w}^{i-1}, \mathbf{w}^{i+1}, \cdots\}$, is

$$\mathbb{P}(\mathbf{w}^i|\mathbf{W}_{-i},\alpha,G_0) = \frac{\alpha}{m-1+\alpha}G_0 + \frac{1}{m-1+\alpha}\sum_{j=1,j\neq i}^m \delta_{\mathbf{w}^j},$$

where $\delta_{\mathbf{w}^j}$ denotes the distribution concentrated at a single point $\mathbf{w}^j$. So $\mathbf{w}^i$ can be equal to either $\mathbf{w}^j$ ($j \neq i$) with probability $\frac{1}{m-1+\alpha}$, which corresponds to the case that those two tasks lie in the same cluster, or a new sample from G_0 with probability $\frac{\alpha}{m-1+\alpha}$, which is the case that task $\mathcal{T}_i$ forms a new task cluster. When α is large, the chance to form a new task cluster is large and so α will affect the number of task clusters. This model is extended in [59], [60] to a case where different tasks in a task cluster share useful features via a matrix stick-breaking process and a beta-Bernoulli hierarchical prior, respectively, and in [61] where each task is a compressive sensing task. Moreover, a nested Dirichlet process is proposed in [62], [63] to use Dirichlet processes to learn both task clusters and the state structure of an infinite hidden Markov model, which handles sequential data in each task. In [64], $\mathbf{w}^i$ is decomposed as $\mathbf{w}^i = \mathbf{u}^i + \mathbf{\Theta}_i^T\mathbf{v}^i$ similar to Eq. (10), where $\mathbf{u}^i$ and $\mathbf{\Theta}_i$ are sampled according to a Dirichlet process.

Different from [3], [55], Jacob et al. [65] aim to learn task clusters under the regularization framework by considering three orthogonal aspects, including a global penalty to measure on average how large the parameters, a measure of between-cluster variance to quantify the distance among different clusters, and a measure of within-cluster variance to quantify the compactness of task clusters. By combining these three aspects and adopting some convex relaxation technique, the convex objective function is formulated as

$$\begin{aligned}\min_{\mathbf{W},\mathbf{b},\mathbf{\Sigma}} \; & L(\mathbf{W},\mathbf{b}) + \lambda \mathrm{tr}(\mathbf{W}\mathbf{1}\mathbf{1}^T\mathbf{W}^T) + \mathrm{tr}(\tilde{\mathbf{W}}\mathbf{\Sigma}^{-1}\tilde{\mathbf{W}}^T) \\ \text{s.t.} \; & \tilde{\mathbf{W}} = \mathbf{W}\mathbf{\Pi}, \; \alpha\mathbf{I} \preceq \mathbf{\Sigma} \preceq \beta\mathbf{I}, \; \mathrm{tr}(\mathbf{\Sigma}) = \gamma,\end{aligned} \tag{16}$$

where $\mathbf{\Pi}$ denotes the $m \times m$ centering matrix, $\mathbf{1}$ denotes a column vector of all ones with its size depending on the context, and α, β, γ are hyperparameters.

Kang et al. [66] extend the MTFL method [11], [12], which treats all the tasks as a whole cluster, to the case with multiple task clusters and aim to minimize the squared trace norm in each cluster. A diagonal matrix, $\mathbf{Q}_i \in \mathbb{R}^{m\times m}$, is defined as a cluster indicator matrix for the ith cluster. The jth diagonal entry of $\mathbf{Q}_i$ is equal to 1 if task $\mathcal{T}_j$ lies in the ith cluster and otherwise 0. Since each task can belong to only one cluster, it is easy to see that $\sum_{i=1}^r \mathbf{Q}_i = \mathbf{I}$. Based on these considerations, the objective function is formulated as

$$\min_{\mathbf{W},\mathbf{b},\{\mathbf{Q}_i\}} L(\mathbf{W},\mathbf{b}) + \lambda \sum_{i=1}^r \|\mathbf{W}\mathbf{Q}_i\|_{S(1)}^2 \;\text{s.t.}\; \begin{array}{c}\mathbf{Q}_i \in \{0,1\}^{m\times m} \\ \sum_{i=1}^r \mathbf{Q}_i = \mathbf{I}\end{array}.$$

When r equals 1, this method reduces to the MTFL method.

Han and Zhang [67] devise a structurally sparse regularizer to cluster tasks with the objective function as

$$\min_{\mathbf{W},\mathbf{b}} L(\mathbf{W},\mathbf{b}) + \lambda \sum_{j>i} \|\mathbf{w}^i - \mathbf{w}^j\|_2. \tag{17}$$

Problem (17) is a special case of the method proposed in [67] with only one level of task clusters. The regularizer on $\mathbf{W}$ enforces any pair of columns in $\mathbf{W}$ to have a chance to be identical and after solving problem (17), the cluster structure can be discovered by comparing columns in $\mathbf{W}$. One advantage of this structurally sparse regularizer is that the convex problem (17) can automatically determine the number of task clusters.

Barzilai and Crammer [68] propose a task clustering method by defining $\mathbf{W}$ as $\mathbf{W} = \mathbf{FG}$ where $\mathbf{F} \in \mathbb{R}^{d\times r}$ and $\mathbf{G} \in \{0,1\}^{r\times m}$. With an assumption that each task belongs to only one cluster, the objective function is formulated as

$$\min_{\mathbf{F},\mathbf{G},\mathbf{b}} L(\mathbf{FG},\mathbf{b}) + \lambda\|\mathbf{F}\|_F^2 \text{ s.t. } \mathbf{G} \in \{0,1\}^{r\times m},\ \|\mathbf{g}^i\|_2 = 1\ \forall i \in [m],$$

where $\mathbf{g}^i$ denotes the ith column of $\mathbf{G}$. When using the hinge loss or logistic loss, this non-convex problem can be relaxed to a min-max problem, which has a global optimum, by utilizing the dual problem with respect to $\mathbf{W}$ and $\mathbf{b}$ and discarding some non-convex constraints.

Zhou and Zhao [69] aim to cluster tasks by identifying representative tasks which are a subset of the given m tasks. If task $\mathcal{T}_i$ is selected by task $\mathcal{T}_j$ as a representative task, then it is expected that model parameters for $\mathcal{T}_j$ are similar to those of $\mathcal{T}_i$. z_{ij} is defined as the probability that task $\mathcal{T}_j$ selects task $\mathcal{T}_i$ as its representative task. Then based on $\{z_{ij}\}$, the objective function is formulated as

$$\min_{\mathbf{W},\mathbf{b},\mathbf{Z}} L(\mathbf{W},\mathbf{b}) + \lambda_1\|\mathbf{W}\|_F^2 + \lambda_2 \sum_{i=1}^{m}\sum_{j=1}^{m} z_{ij}\|\mathbf{w}^i - \mathbf{w}^j\|_2^2 + \lambda_3\|\mathbf{Z}\|_{2,1}$$
$$\text{s.t. } \mathbf{Z} \geq \mathbf{0},\ \mathbf{Z}^T\mathbf{1} = \mathbf{1}. \tag{18}$$

The third term in the objective function of problem (18) enforces the closeness of each pair of tasks based on $\mathbf{Z}$ and the last term employs the $\ell_{2,1}$ norm to enforce the row sparsity of $\mathbf{Z}$ which implies that the number of representative tasks is limited. The constraints in problem (18) guarantees that entries in $\mathbf{Z}$ define valid probabilities. Problem (18) is related to problem (17) since the regularizer in problem (17) can be reformulated as $2\sum_{j>i}\|\mathbf{w}^i - \mathbf{w}^j\|_2 = \min_{\hat{\mathbf{Z}}\geq\mathbf{0}} \sum_{j>i}\left(\hat{z}_{ij}\|\mathbf{w}^i - \mathbf{w}^j\|_2^2 + \frac{1}{\hat{z}_{ij}}\right)$, where both the regularizer and constraint on $\hat{\mathbf{Z}}$ are different from those on $\mathbf{Z}$ in problem (18).

Previous studies assume that each task can belong to only one task cluster and this assumption seems too restrictive. In [70], a GO-MTL method relaxes this assumption by allowing a task to belong to more than one cluster and defines a decomposition of $\mathbf{W}$ similar to [68] as $\mathbf{W} = \mathbf{LS}$ where $\mathbf{L} \in \mathbb{R}^{d\times r}$ denotes the latent basis with $r < m$ and $\mathbf{S} \in \mathbb{R}^{r\times m}$ contains linear combination coefficients for all the tasks. $\mathbf{S}$ is assumed to be sparse since each task is generated from only a few columns in the latent basis or equivalently belongs to a small number of clusters. The objective function is formulated as

$$\min_{\mathbf{L},\mathbf{S},\mathbf{b}} L(\mathbf{LS},\mathbf{b}) + \lambda_1\|\mathbf{S}\|_1 + \lambda_2\|\mathbf{L}\|_F^2. \tag{19}$$

Compared with the objective function of multi-task sparse coding, i.e., problem (3), we can see that when the regularization parameters take appropriate values, these two problems are almost equivalent except that in multi-task sparse coding, the dictionary $\mathbf{U}$ is overcomplete, implying that the number of columns in $\mathbf{U}$ is larger than that of rows, while here the number of columns in $\mathbf{S}$ is smaller than that of its rows. This method has been extended in [71] to decompose the parameter tensor in the fully connected layers of deep neural networks.

Among the aforementioned methods, the method in [53] first identifies the cluster structure and then learns the model parameters of all the tasks separately, which is not preferred since the cluster structure learned may be suboptimal for the model parameters, hence follow-up works learn model parameters and the cluster structure together. An important problem in clustering is to determine the number of clusters and this is also important for this approach. Out of the above methods, only methods in [3], [67] can automatically determine the number of task clusters, where the method in [3] depends on the capacity of the Dirichlet process while the method in [67] relies on the use of a structurally sparse regularizer. For the aforementioned methods, some of them belong to Bayesian learning, i.e., [3], [55], while the rest models are regularized models. Among these regularized methods, only the objective function proposed in [67] is convex while others are originally non-convex.

The task clustering approach is related to the low-rank approach. To see that, suppose that there are r task clusters where $r < m$ and all the tasks in a cluster share the same model parameters, making the parameter matrix $\mathbf{W}$ low-rank with the rank at most r. From the perspective of modeling, by setting $\mathbf{u}^i$ to be a zero vector in Eq. (10), we can see that the decomposition of $\mathbf{W}$ in [44] becomes similar to those in [68], [70], which in some aspect demonstrates the relations between these two approaches. Moreover, the equivalence between problems (13) and (16), two typical methods in the low-rank and task clustering approaches, has been proved in [72]. The task clustering approach can visualize the cluster structure, which is an advantage over the low-rank approach.

2.4 Task Relation Learning Approach

In MTL, tasks are related and the task relatedness can be quantitated via task similarity, task correlation, task covariance and so on. Here we use task relations to include all the quantitative relatedness.

In earlier studies on MTL, the task relations are assumed to be known as a priori information. In [73], [74], each task is assumed to be similar to any other task and so model parameters of each task will be enforced to approach the average model parameters of all the tasks. In [75], [76], [77], task similarities for each pair of tasks are given and these studies utilize the task similarities to design regularizers to guide the learning of multiple tasks in a principle that the more similar two tasks are, the closer the corresponding model parameters are expected to be. Moreover, given a tree structure describing relations among tasks in [78], model parameters of a task corresponding to a node in the tree is enforced to be similar to those of its parent node.

However, in most applications, task relations are not available. In this case, learning task relations from data automatically is a good option. Bonilla et al. [79] propose a multi-task Gaussian process (MTGP) by defining a prior on f_j^i, the functional value for $\mathbf{x}_j^i$, as $\mathbf{f} \sim \mathcal{N}(\mathbf{0},\mathbf{\Sigma})$, where $\mathbf{f} = (f_1^1,\ldots,f_{n_m}^m)^T$ contains the functional values for all the training data. $\mathbf{\Sigma}$, the covariance matrix, defines the covariance between f_j^i and f_q^p as $\sigma(f_j^i, f_q^p) = \omega_{ip}k(\mathbf{x}_j^i,\mathbf{x}_q^p)$, where $k(\cdot,\cdot)$ denotes a kernel function and ω_{ip} describes the covariance between tasks $\mathcal{T}_i$ and $\mathcal{T}_p$. In order to keep $\mathbf{\Sigma}$ positive definite, a matrix $\mathbf{\Omega}$ containing ω_{ip}

as its (i,p)th entry is also required to be positive definite, which makes $\mathbf{\Omega}$ the task covariance to describe the similarities between tasks. Then based on the Gaussian likelihood for labels given $\mathbf{f}$, the analytically marginal likelihood by integrating out $\mathbf{f}$ can be used to learn $\mathbf{\Omega}$ from data. In [80], the learning curve and generalization bound of the multi-task Gaussian process are studied. Since $\mathbf{\Omega}$ in MTGP has a point estimation which may lead to the overfitting, based on a proposed weight-space view of MTGP, Zhang and Yeung [81] propose a multi-task generalized t process by placing an inverse-Wishart prior on $\mathbf{\Omega}$ as $\mathbf{\Omega} \sim \mathcal{IW}(\nu, \mathbf{\Psi})$, where ν denotes the degree of freedom and $\mathbf{\Psi}$ is the base covariance for generating $\mathbf{\Omega}$. Since $\mathbf{\Psi}$ models the covariance between pairs of tasks, it can be determined based on the maximum mean discrepancy (MMD) [82].

Different from [79], [81] which are Bayesian models, Zhang and Yeung [4], [5] propose a regularized multi-task model called multi-task relationship learning (MTRL) by placing a matrix-variate normal prior on $\mathbf{W}$ as

$$\mathbf{W} \sim \mathcal{MN}(\mathbf{0}, \mathbf{I}, \mathbf{\Omega}), \tag{20}$$

where $\mathcal{MN}(\mathbf{M}, \mathbf{A}, \mathbf{B})$ denotes a matrix-variate normal distribution with $\mathbf{M}$ as the mean, $\mathbf{A}$ the row covariance, and $\mathbf{B}$ the column covariance. Based on this prior as well as some likelihood function, the objective function for the maximum a posterior solution is formulated as

$$\begin{aligned} \min_{\mathbf{W},\mathbf{b},\mathbf{\Omega}} \ & L(\mathbf{W},\mathbf{b}) + \lambda_1\|\mathbf{W}\|_F^2 + \lambda_2 \mathrm{tr}(\mathbf{W}\mathbf{\Omega}^{-1}\mathbf{W}^T) \\ \text{s.t.} \ & \mathbf{\Omega} \succ \mathbf{0}, \ \mathrm{tr}(\mathbf{\Omega}) \le 1, \end{aligned} \tag{21}$$

where the second term in the objective function is to penalize the complexity of $\mathbf{W}$, the last term is due to the matrix-variate normal prior, and the constraints control the complexity of the positive definite covariance matrix $\mathbf{\Omega}$. It has been proved in [4], [5] that problem (21) is jointly convex with respect to $\mathbf{W}$, $\mathbf{b}$, and $\mathbf{\Omega}$. Problem (21) has been extended to multi-task boosting [83] and multi-label learning [84] by learning label correlations. Problem (21) can also been interpreted from the perspective of reproducing kernel Hilbert spaces for vector-valued functions [85], [86], [87], [88]. Moreover, Problem (21) is extended to learn sparse task relations in [89] via the ℓ_1 regularization on $\mathbf{\Omega}$ when the number of tasks is large. A model similar to problem (21) is proposed in [90] via a matrix-variate normal prior on $\mathbf{W}$: $\mathbf{W} \sim \mathcal{MN}(\mathbf{0}, \mathbf{\Omega}_1, \mathbf{\Omega}_2)$, where the inverses of $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$ are assumed to be sparse. The MTRL model is extended in [91] to use the symmetric matrix-variate generalized hyperbolic distribution to learn block sparse structure in $\mathbf{W}$ and in [92] to use the matrix generalized inverse Gaussian prior to learn low-rank $\mathbf{\Omega}_1$ and $\mathbf{\Omega}_2$. Moreover, the MTRL model is generalized to the multi-task feature selection problem [41] by learning task relations via the matrix-variate generalized normal distribution. Since the prior defined in Eq. (20) implies that $\mathbf{W}^T\mathbf{W}$ follows $\mathcal{W}(\mathbf{0}, \mathbf{\Omega})$, where $\mathcal{W}(\cdot,\cdot)$ denotes a Wishart distribution, Zhang and Yeung [93] generalize it as

$$(\mathbf{W}^T\mathbf{W})^t \sim \mathcal{W}(\mathbf{0}, \mathbf{\Omega}), \tag{22}$$

where t is a positive integer to model high-order task relationships. Eq. (22) can induce a new prior, which is a generalization of the matrix-variate normal distribution, on $\mathbf{W}$ and based on this new prior, a new regularized method is devised to learn the task relations in [93]. As a special case of MTL, multi-output regression problems, where each output is treated as a task and all the tasks share the training data, are investigated in [92], [94], [95], [96] to not only learn the relations among different outputs/tasks in a way similar to problem (21) but also model the structure contained in noises via some matrix-variate priors. The MTRL method has been extended to deep neural networks in [97] by placing a tensor norm distribution as a prior on the parameter tensor in the fully connected layers.

Different from the aforementioned methods which investigate the use of global learning models in MTL, Zhang [98] aims to learn the task relations in local learning methods such as the k-nearest-neighbor (kNN) classifier by defining the learning function as a weighted voting of neighbors:

$$f(\mathbf{x}_j^i) = \sum_{(p,q)\in N_k(i,j)} \sigma_{ip} s(\mathbf{x}_j^i, \mathbf{x}_q^p) y_q^p, \tag{23}$$

where $N_k(i,j)$ denotes the set of task indices and instance indices for the k nearest neighbors of $\mathbf{x}_j^i$, i.e., $(p,q) \in N_k(i,j)$ meaning that $\mathbf{x}_q^p$ is one of the k nearest neighbors of $\mathbf{x}_j^i$, $s(\mathbf{x}_j^i, \mathbf{x}_q^p)$ defines the similarity between $\mathbf{x}_j^i$ and $\mathbf{x}_q^p$, and σ_{ip} represents the contribution of task $\mathcal{T}_p$ to $\mathcal{T}_i$ when $\mathcal{T}_p$ has some data points to be neighbors of a data point in $\mathcal{T}_i$. σ_{ip} can be viewed as the similarity from $\mathcal{T}_p$ to $\mathcal{T}_i$. When $\sigma_{ip} = 1$ for all i and p, Eq. (23) reduces to the decision function of the kNN classifier for all the tasks. Then the objective function to learn $\mathbf{\Sigma}$, which is a $m \times m$ matrix with σ_{ip} as its (i,p)th entry, can be formulated as

$$\begin{aligned} \min_{\mathbf{\Sigma}} \quad & \sum_{i=1}^m \sum_{j=1}^{n_i} l(y_j^i, f(\mathbf{x}_j^i)) + \frac{\lambda_1}{4}\|\mathbf{\Sigma} - \mathbf{\Sigma}^T\|_F^2 + \frac{\lambda_2}{2}\|\mathbf{\Sigma}\|_F^2 \\ \text{s.t.} \quad & \sigma_{ii} \ge 0 \ \forall i \in [m], -\sigma_{ii} \le \sigma_{ij} \le \sigma_{ii} \ \forall i \ne j. \end{aligned} \tag{24}$$

The first regularizer in problem (24) enforces $\mathbf{\Sigma}$ to be nearly symmetric depending on λ_1 and the second one is to penalize the complexity of $\mathbf{\Sigma}$. The constraints in problem (24) make sure that the similarity from one task to itself is positive and also the largest. Similarly, a multi-task kernel regression is proposed in [98] for regression problems.

While the aforementioned methods whose task relations are symmetric except [98], Lee et al. [99] focus on learning asymmetric task relations. Since different tasks are assumed to be related, $\mathbf{w}_i$ can lie in the space spanned by $\mathbf{W}$, i.e., $\mathbf{w}_i \approx \mathbf{W}\mathbf{a}_i$, and hence we have $\mathbf{W} \approx \mathbf{W}\mathbf{A}$. Here matrix $\mathbf{A}$ can be viewed as asymmetric task relations between pairs of tasks. By assuming that $\mathbf{A}$ is sparse, the objective function is formulated as

$$\begin{aligned} \min_{\mathbf{W},\mathbf{b},\mathbf{A}} \ & \sum_{i=1}^m (1 + \lambda_1\|\hat{\mathbf{a}}_i\|_1) \sum_{j=1}^{n_i} l(y_j^i, (\mathbf{w}^i)^T\mathbf{x}_j^i + b_i) + \lambda_2\|\mathbf{W} - \mathbf{W}\mathbf{A}\|_F^2 \\ \text{s.t.} \ & a_{ij} \ge 0 \ \forall i,j \in [m], \end{aligned} \tag{25}$$

where $\hat{\mathbf{a}}_i$ denotes the ith row of $\mathbf{A}$ by deleting a_{ii}. The term before the training loss of each task, i.e., $1 + \lambda_1\|\hat{\mathbf{a}}_i\|_1$, not only enforces $\mathbf{A}$ to be sparse but also allows asymmetric information transfer from easier tasks to difficult ones. The regularizer in problem (25) can make $\mathbf{W}$ approach $\mathbf{W}\mathbf{A}$ with the closeness depending on λ_2. To see the connection between problems (25) and (21), we rewrite the regularizer in problem (25) as $\|\mathbf{W} - \mathbf{W}\mathbf{A}\|_F^2 = \mathrm{tr}\left(\mathbf{W}(\mathbf{I}-\mathbf{A})(\mathbf{I}-\mathbf{A})^T\mathbf{W}^T\right)$. Based on this reformulation, the regularizer in problem (25) is a special case of that in problem (21) by assuming $\mathbf{\Omega}^{-1} = (\mathbf{I}-\mathbf{A})(\mathbf{I}-\mathbf{A})^T$ where $\mathbf{A}$ is a nonnegative matrix. Even though $\mathbf{A}$ is asymmetric, from the perspective of the regularizer, the task relations here are symmetric and act as the task precision matrix which has a restrictive form.

2.5 Decomposition Approach

The decomposition approach assumes that the parameter matrix $\mathbf{W}$ can be decomposed into two or more component matrices $\{\mathbf{W}_k\}_{k=1}^h$ where $h \ge 2$, i.e., $\mathbf{W} = \sum_{k=1}^h \mathbf{W}_k$. The objective functions of most methods in this approach can be unified as

$$\min_{\{\mathbf{W}_i\}\in\mathcal{C}_W,\mathbf{b}} L(\mathbf{W},\mathbf{b}) + \sum_{k=1}^h g_k(\mathbf{W}_k), \tag{26}$$

where the regularizer is decomposable with respect to $\mathbf{W}_k$'s and $\mathcal{C}_W$ denotes a set of constraints for component matrices. To help understand problem (26), we introduce several instantiations.

In [100] where h equals 2 and $\mathcal{C}_W = \emptyset$ is an empty set, $g_1(\cdot)$ and $g_2(\cdot)$ are defined as

$$g_1(\mathbf{W}_1) = \lambda_1 \|\mathbf{W}_1\|_{\infty,1},\ g_2(\mathbf{W}_2) = \lambda_2 \|\mathbf{W}_2\|_1,$$

where λ_1 and λ_2 are positive regularization parameters. Similar to problem (5), each row of $\mathbf{W}_1$ is likely to be a zero row and hence $g_1(\mathbf{W}_1)$ can help select important features. Due to the ℓ_1 norm regularization, $g_2(\mathbf{W}_2)$ makes $\mathbf{W}_2$ sparse. Because of the characteristics of two regularizers, the parameter matrix $\mathbf{W}$ can eliminate unimportant features for all the tasks when the corresponding rows in both $\mathbf{W}_1$ and $\mathbf{W}_2$ are sparse. Moreover, $\mathbf{W}_2$ can identify features for tasks which have their own useful features and may be outliers for other tasks. Hence this model can be viewed as a 'robust' version of problem (5).

With two component matrices, Chen et al. [101] define

$$g_2(\mathbf{W}_2) = \lambda_2 \|\mathbf{W}_2\|_1,\ \mathcal{C}_W = \{\mathbf{W}_1 | \|\mathbf{W}_1\|_{S(1)} \leq \lambda_1\}, \tag{27}$$

where $g_1(\mathbf{W}_1) = 0$. Similar to problem (14), $\mathcal{C}_W$ makes $\mathbf{W}_1$ low-rank. With a sparse regularizer $g_2(\mathbf{W}_2)$, $\mathbf{W}_2$ makes the entire model matrix $\mathbf{W}$ more robust to outlier tasks in a way similar to [100]. When λ_2 is large enough, $\mathbf{W}_2$ will become a zero matrix and hence problem (27) will act similarly to problem (14).

$g_i(\cdot)$'s in [102] with $\mathcal{C}_W = \emptyset$ are defined as

$$g_1(\mathbf{W}_1) = \lambda_1 \|\mathbf{W}_1\|_{S(1)}, \quad g_2(\mathbf{W}_2) = \lambda_2 \|\mathbf{W}_2^T\|_{2,1}. \tag{28}$$

Different from the above two models which assume that $\mathbf{W}_2$ is sparse, here $g_2(\mathbf{W}_2)$ enforces $\mathbf{W}_2$ to be column-sparse. For related tasks, their columns in $\mathbf{W}_1$ are correlated via the trace norm regularization and the corresponding columns in $\mathbf{W}_2$ are zero. For outlier tasks which are unrelated to other tasks, the corresponding columns in $\mathbf{W}_2$ can take arbitrary values and hence model parameters for them in $\mathbf{W}$ have no low-rank structure even though those in $\mathbf{W}_1$ may have.

In [103], these functions are defined as

$$g_1(\mathbf{W}_1) = \lambda_1 \|\mathbf{W}_1\|_{2,1},\ g_2(\mathbf{W}_2) = \lambda_2 \|\mathbf{W}_2^T\|_{2,1},\ \mathcal{C}_W = \emptyset. \tag{29}$$

Similar to problem (4), $g_1(\mathbf{W}_1)$ makes $\mathbf{W}_1$ row-sparse. Here $g_2(\mathbf{W}_2)$ is identical to that in [102] and it makes $\mathbf{W}_2$ column-sparse. Hence $\mathbf{W}_1$ helps select useful features while non-zero columns in $\mathbf{W}_2$ capture outlier tasks.

With $h = 2$, Zhong and Kwok [104] define

$$g_1(\mathbf{W}_1) = \lambda_1 c(\mathbf{W}_1) + \lambda_2 \|\mathbf{W}_1\|_F^2, g_2(\mathbf{W}_2) = \lambda_3 \|\mathbf{W}_2\|_F^2, \mathcal{C}_W = \emptyset,$$

where $c(\mathbf{U}) = \sum_{i=1}^d \sum_{k>j} |u_{ij} - u_{ik}|$ with u_{ij} as the (i,j)th entry in a matrix $\mathbf{U}$. Due to the sparse nature of the ℓ_1 norm, $c(\mathbf{W}_1)$ enforces corresponding entries in different columns of $\mathbf{W}_1$ to be identical, which is equivalent to clustering tasks in terms of individual model parameters. Both the squared Frobenius norm regularizations in $g_1(\mathbf{W}_1)$ and $g_2(\mathbf{W}_2)$ penalize the complexities of $\mathbf{W}_1$ and $\mathbf{W}_2$. The use of $\mathbf{W}_2$ improves the model flexibility when not all the tasks exhibit clear cluster structure.

Different from the aforementioned methods which have only two component matrices, an arbitrary number of component matrices are considered in [105] with

$$g_k(\mathbf{W}_k) = \lambda\big[(h-k)\|\mathbf{W}_k\|_{2,1} + (k-1)\|\mathbf{W}_k\|_1\big]/(h-1), \tag{30}$$

where $\mathcal{C}_W = \emptyset$. According to Eq. (30), $\mathbf{W}_k$ is assumed to be both sparse and row-sparse for all $k \in [h]$. Based on different regularization parameters on the regularizer of $\mathbf{W}_k$, we can see that when k increases, $\mathbf{W}_k$ is more likely to be sparse than to be row-sparse. Even though each $\mathbf{W}_k$ is sparse or row-sparse, the entire parameter matrix $\mathbf{W}$ can be non-sparse and hence this model can discover the latent sparse structure among multiple tasks.

In the above methods, different component matrices have no direct connection. When there is a dependency among component matrices, problem (26) can model more complex structure. For example, Han and Zhang [106] define

$$g_k(\mathbf{W}_k) = \lambda \sum_{i>j} \|\mathbf{w}_k^i - \mathbf{w}_k^j\|_2 / \eta^{k-1}\ \forall k \in [h]$$
$$\mathcal{C}_W = \{\{\mathbf{W}_k\} |\ |\mathbf{w}_{k-1}^i - \mathbf{w}_{k-1}^j| \geq |\mathbf{w}_k^i - \mathbf{w}_k^j|\ \forall k \geq 2,\ \forall i > j\},$$

where $\mathbf{w}_k^i$ denotes the ith column of $\mathbf{W}_k$. Note that the constraint set $\mathcal{C}_W$ relates component matrices and the regularizer $g_k(\mathbf{W}_k)$ makes each pair of $\mathbf{w}_k^i$ and $\mathbf{w}_k^j$ has a chance to become identical. Once this happens for some i, j, k, then based on the constraint set $\mathcal{C}_W$, $\mathbf{w}_{k'}^i$ and $\mathbf{w}_{k'}^j$ will always have the same value for $k' \geq k$. This corresponds to sharing all the ancestor nodes for two internal nodes in a tree and hence this method can learn a hierarchical structure to characterize task relations. When the constraints are removed, this method reduces to the multi-level task clustering method [67], which is a generalization of problem (17).

Another way to relate different component matrices is to use a non-decomposable regularizer as [107] did, which is slightly different from problem (26) in terms of the regularizer. Specifically, given m tasks, there are $2^m - 1$ possible and non-empty task clusters. All the task clusters can be organized in a tree, where the root node represents a dummy node, nodes in the second level represents groups with a single task, and the parent-child relations are the 'subset of' relation. In total, there are 2^m component matrices each of which corresponds to a node in the tree and hence an index a is used to denote both a level and the corresponding node in the tree. The objective function is formulated as

$$\min_{\{\mathbf{W}_i\},\mathbf{b}} L(\mathbf{W},\mathbf{b}) + \Big(\sum_{v\in V} \lambda_v \Big(\sum_{a\in D(v)} r(\mathbf{W}_a)^p\Big)^{\frac{1}{p}}\Big)^2$$
$$\text{s.t. } \mathbf{w}_a^i = \mathbf{0}\ \forall i \notin t(a), \tag{31}$$

where p takes a value between 1 and 2, $D(a)$ denotes the set of all the descendants of a, $t(a)$ denotes the set of tasks contained in node a, $\mathbf{w}_a^i$ denotes the ith column of $\mathbf{W}_a$, and $r(\mathbf{W}_a)$ reflects relations among tasks in node a based on $\mathbf{W}_a$. The regularizer in problem (31) is used to prune the subtree rooted at each node v based on the ℓ_p norm. The constraint in problem (31) implies that for tasks not contained in a node a, the corresponding columns in $\mathbf{W}_a$ are zero. In [107], $r(\mathbf{W}_a)$ adopts the regularizer proposed in [73] which enforces the parameters of all the tasks to approach their average.

Different from deep MTL models which are deep in terms of the feature representation, the decomposition approach can be viewed as a 'deep' approach in terms of model parameters while most of previous approaches are just shallow ones, making this approach have more powerful capacity. Moreover, the decomposition approach can reduce to other approaches such as the feature learning, low-rank, and task clustering approaches when there is only one component matrix and hence it can be considered as an improved version of these approaches.

2.6 Comparisons among Different Approaches

Based on the above introduction, we can see that different approaches exhibit their own characteristics. Specifically, the feature

TABLE 1
The performance comparison of representative MTL models in the five approaches on benchmark datasets in terms of some evaluation metric. nMSE stands for 'normalized mean squared error' and RMSE is for 'root mean squared error'. ↑ after the evaluation metric implies that the larger value the better performance and ↓ indicates the opposite case.

Dataset (Reference)	Evaluation Metric	Feature Learning	Low-Rank	Task Clustering	Task Relation Learning	Decomposition
		[11]	[49]	[65]/ [66]/ [70]/ [71]	[4]/ [88]/ [97]	[100]/ [101]/ [102]/ [107]/ [106]
School ([106])	nMSE↓	0.4393	—	0.4374/-/0.6466/-	—	0.4445/-/-/-/0.4169
SARCOS ([102])	nMSE↓	0.1568	0.1531	—	—	0.1495/0.1456/-/-/-
Computer Survey ([104])	RMSE↓	—	—	2.072/-/-/-	2.110/-/-	2.138/2.052/2.074/-/-
Parkinson ([88])	Explained Variance↑	—	—	2.7%/33.6%/-/-	12.0%/27.0%/-	-/-/-/16.8%/-
Sentiment ([4])	Classification Error↓	0.2756	—	—	0.2324/-/-	—
MHC-I ([69])	Classification Error↓	—	—	0.1890/0.2050/-/-	0.1870/-/-	0.2030/-/0.2070/-/-
Landmine ([88])	AUC↑	—	—	75.9%/76.7%/-/-	76.1%/76.8%/-	-/-/-/76.4%/-
Office-Caltech ([97])	Classification Error↓	0.0740	—	-/-/-/0.0670	0.0690/-/0.0450	-/-/0.0760/-/-
Office-Home ([97])	Classification Error↓	0.4170	—	-/-/-/0.3350	0.4070/-/0.3310	-/-/0.4140/-/-
ImageCLEF ([97])	Classification Error↓	0.3440	—	-/-/-/0.2780	0.3350/-/0.2470	-/-/0.3510/-/-

learning approach can learn common features, which are generic and transferable to all the tasks at hand and even new tasks, for all the tasks. When there exist outlier tasks which are unrelated to other tasks, the learned features can be influenced by outlier tasks significantly and they can cause the performance deterioration, making this approach not so robust to outlier tasks. By assuming that the parameter matrix is low-rank, the low-rank approach can explicitly learn the subspace of the parameter matrix or implicitly achieve that via some convex or non-convex regularizer. This approach is powerful but it seems applicable to only linear models, making nonlinear extensions non-trivial to be devised. The task clustering approach performs clustering on the task level in terms of model parameters and it can identify task clusters each of which consists of similar tasks. A major limitation of the task clustering approach is that it can capture positive correlations among tasks in the same cluster but ignore negative correlations among tasks in different clusters. Moreover, even though some methods in this category can automatically determine the number of clusters, most of them still need a model selection method such as cross validation to determine it, which may bring more computational cost. The task relation learning approach can learn model parameters and pairwise task relations simultaneously. The learned task relations can give us insight about the relationships between tasks and they improve the interpretability. The decomposition approach can be viewed as extensions of other parameter-based approaches by equipping multi-level parameters and hence they can model more complex task structure, i.e., tree structure. The number of components in the decomposition approach is important to the performance and needs to be carefully determined.

2.7 Benchmark Datasets and Performance Comparison

In this section, we introduce some benchmark datasets for MTL and compare the performance of different MTL models on them.

Some benchmark datasets for MTL are listed as follows.

- School dataset [55]: This dataset is to estimate examination scores of 15,362 students from 139 secondary schools in London from 1985 to 1987 where each school is treated as a task. The input consists of four school-specific and three student-specific attributes.
- SARCOS dataset[2]: This dataset studies a multi-output problem of learning the inverse dynamics of 7 SARCOS anthropomorphic robot arms, each of which corresponds to a task, based on 21 features, including seven joint positions, seven joint velocities, and seven joint accelerations. This dataset contains 48,933 data points.
- Computer Survey dataset [13]: This dataset is taken from a survey of 180 persons/tasks who rated the likelihood of purchasing one of 20 different personal computers, resulting in 36,000 data points in all the tasks. The features contain 13 different computer characteristics (e.g., price, CPU, and RAM) while the output is an integer rating on the scale 0-10.
- Parkinson dataset [107]: This dataset is to predict the disease symptom score of Parkinson for patients at different times using 19 bio-medical features. This dataset has 5,875 data points for 42 patients each of which is a task.
- Sentiment dataset[3]: This dataset is to classify reviews of four products/tasks, i.e., books, DVDs, electronics, and kitchen appliances, from Amazon into two classes: positive and negative reviews. For each task, there are 1,000 positive and 1,000 negative reviews, respectively.
- MHC-I dataset [65]: This databset contains binding affinities of 15,236 peptides with 35 MHC-I molecules. Each MHC-I molecule is considered as a task and the goal is to predict whether a peptide binds a molecule.
- Landmine dataset [3]: This dataset consists of 9-dimensional data points, whose features are extracted from radar images, from 29 landmine fields/tasks. Each task is to classify a data point into two classes (landmine or clutter). There are 14,820 data points in total.
- Office-Caltech dataset [108]: The dataset contains data from 10 common categories shared in the Caltech-256 dataset and the Office dataset which consists of images collected from three distinct domains/tasks: Amazon, Webcam, and DSLR, making this dataset contain 4 tasks. There are 2,533 images in all the tasks.
- Office-Home dataset[4]: This dataset consists of images from 4 different domains/tasks: artistic images, clip art, product images, and real-world images. Each task contains images of 65 object categories collected in the office and home settings. In total, there are about 15,500 images in all the tasks.
- ImageCLEF dataset[5]: This dataset contains 12 common categories shared by four tasks: Caltech-256, ImageNet ILSVRC 2012, Pascal VOC 2012, and Bing. There are about 2,400 images in all the tasks.

In the above benchmark datasets, the first four datasets consist

[2] http://www.gaussianprocess.org/gpml/data/

[3] http://www.cs.jhu.edu/~mdredze/datasets/sentiment/

[4] http://hemanthdv.org/OfficeHome-Dataset

[5] http://imageclef.org/2014/adaptation

of regression tasks while the other datasets are classification tasks, where each task in the Sentiment, MHC-I, and Landmine datasets is a binary classification problem and that in the other three image datasets is a multi-class classification problem. In order to compare different MTL approaches on those benchmark datasets, we select some representative MTL methods from each of the five approaches introduced in the previous sections and list in Table 1 their performance reported in the MTL literature. Usually, different datasets have their own characteristics, making them more suitable for some MTL approach. For example, according to the studies in [55], [75], [104], different tasks in the School dataset are found to be very similar to each other. According to [3], the Landmine dataset can have two task clusters, where the first cluster consisting of the first 15 tasks corresponds to regions that are relatively highly foliated and the rest tasks belong to another cluster with regions that are bare earth or deserts. According to [65], it is well known in the vaccine design community that some molecules/tasks in the MHC-I dataset can be grouped into empirically defined supertypes known to have similar binding behaviors. For these three datasets, according to Table 1 we can see that the task clustering, task relation learning and decomposition approaches have better performance since they can identify the cluster structure contained in the data in a plain or hierarchical way. For other datasets, they do not have so obvious structure among tasks but some MTL models can learn task correlations, which can bring much insight for model design and the interpretation of experimental results. For example, the task correlations in the SARCOS and Sentiment datasets are shown in Tables 2 and 3 of [4], and the task similarities in the Office-Caltech dataset is shown in Figure 3(b) of [97]. Moreover, for image datasets (i.e., Office-Caltech, Office-Home and ImageCLEF), deep MTL models (e.g., [71], [97]) achieve better performance than shallow models since they can learn powerful feature representations, while the rest datasets are from diverse areas, making shallow models perform well on them.

2.8 Another Taxonomy for Regularized MTL Methods

Regularized methods form a main methodology for MTL. Here we classify many regularized MTL algorithms into two main categories: learning with feature covariance and learning with task relations. Learning with feature covariance can be viewed as a representative formulation in feature-based MTL while learning with task relations is for parameter-based MTL.

Objective functions in the first category can be unified as

$$\min_{\mathbf{W},\mathbf{b},\boldsymbol{\Theta}} L(\mathbf{W},\mathbf{b}) + \frac{\lambda}{2}\mathrm{tr}(\mathbf{W}^T\boldsymbol{\Theta}^{-1}\mathbf{W}) + f(\boldsymbol{\Theta}), \tag{32}$$

where $f(\cdot)$ denotes a regularizer or constraint on $\boldsymbol{\Theta}$. From the perspective of probabilistic modeling, the regularizer $\frac{\lambda}{2}\mathrm{tr}(\mathbf{W}^T\boldsymbol{\Theta}^{-1}\mathbf{W})$ corresponds to a matrix-variate normal distribution on $\mathbf{W}$ as $\mathbf{W} \sim \mathcal{MN}(\mathbf{0}, \frac{1}{\lambda}\boldsymbol{\Theta} \otimes \mathbf{I})$. Based on this probabilistic prior, $\boldsymbol{\Theta}$ models the covariance between the features since $\frac{1}{\lambda}\boldsymbol{\Theta}$ is the row covariance matrix where each row in $\mathbf{W}$ corresponds to a feature and different tasks share the feature covariance. All the models in this category differ in the choice of the function $f(\cdot)$ on $\boldsymbol{\Theta}$. For example, methods in [11], [44], [45] use $f(\cdot)$ to restrict the trace of $\boldsymbol{\Theta}$ as shown in problems (2) and (13). Moreover, multi-task feature selection methods based on the $\ell_{2,1}$ norm such as [24], [25], [26], [27] can be reformulated as instances of problem (32) by using an alternative form as

$$\|\mathbf{W}\|_{2,1} = \min_{\boldsymbol{\theta}\in\mathbb{R}^d,\boldsymbol{\theta}\geq\mathbf{0}} \frac{1}{2}\left(\mathrm{tr}\left(\mathbf{W}^T\mathrm{diag}(\boldsymbol{\theta})^{-1}\mathbf{W}\right) + \mathbf{1}^T\boldsymbol{\theta}\right), \tag{33}$$

where $\mathrm{diag}(\cdot)$ converts a vector to a diagonal matrix.

Different from the first category, methods in the second category have a unified objective function as

$$\min_{\mathbf{W},\mathbf{b},\boldsymbol{\Sigma}} L(\mathbf{W},\mathbf{b}) + \frac{\lambda}{2}\mathrm{tr}(\mathbf{W}\boldsymbol{\Sigma}^{-1}\mathbf{W}^T) + g(\boldsymbol{\Sigma}), \tag{34}$$

where $g(\cdot)$ denotes a regularizer or constraint on $\boldsymbol{\Sigma}$. The regularizer $\frac{\lambda}{2}\mathrm{tr}(\mathbf{W}\boldsymbol{\Sigma}^{-1}\mathbf{W}^T)$ corresponds to a matrix-variate normal prior on $\mathbf{W}$ as $\mathbf{W} \sim \mathcal{MN}(\mathbf{0}, \mathbf{I} \otimes \frac{1}{\lambda}\boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is to model the task relations since $\frac{1}{\lambda}\boldsymbol{\Sigma}$ is the column covariance where each column in $\mathbf{W}$ corresponds to a task. From this perspective, the two regularizers for $\mathbf{W}$ in problems (32) and (34) have different meanings even though the formulations seem a bit similar. All the methods in this category use different functions $g(\cdot)$ to learn $\boldsymbol{\Sigma}$ with different functionalities. For example, the methods in [73], [74], [75], [76], [77], which utilize a priori information on task relations, directly learn $\mathbf{W}$ and $\mathbf{b}$ by defining $g(\boldsymbol{\Sigma}) = 0$, some task clustering methods [65], [69] identify task clusters by assuming that $\boldsymbol{\Sigma}$ has a block structure, several task relation learning methods including [4], [5], [89], [99], [109] directly learn $\boldsymbol{\Sigma}$ as a covariance matrix by constraining its trace or sparsity in $g(\boldsymbol{\Sigma})$, and the trace norm regularization [49] has a formulation similar to problem (34) based on an alternative form as

$$\|\mathbf{W}\|_{S(1)} = \min_{\boldsymbol{\Sigma}\succeq\mathbf{0}} \frac{1}{2}\left(\mathrm{tr}(\mathbf{W}\boldsymbol{\Sigma}^{-1}\mathbf{W}^T) + \mathrm{tr}(\boldsymbol{\Sigma})\right), \tag{35}$$

and based on Eqs. (33) and (35), a decomposition method [102] with regularizers defined in Eq. (28) can be treated as an instance of problem (34) by defining that the parameter matrix $\hat{\mathbf{W}}$ is defined as $\hat{\mathbf{W}} = (\mathbf{W}_1, \mathbf{W}_2)$ and that $\boldsymbol{\Sigma}$ has a block structure as $\begin{pmatrix} \lambda_1\boldsymbol{\Sigma}_1 & \mathbf{0} \\ \mathbf{0} & \lambda_2\mathrm{diag}(\boldsymbol{\sigma}_2) \end{pmatrix}$.

Moreover, another decomposition method [103] with regularizers defined in Eq. (29) can be viewed as a hybrid of problems (32) and (34) based on Eqs. (33) and (35).

Even though this taxonomy cannot cover all the regularized MTL methods, it can bring some insights to understand regularized MTL methods better and help devise more MTL models. For example, a learning to multi-task framework is proposed in [110] to learn a suitable multi-task model for a given multi-task problem under problem (34) by utilizing $\boldsymbol{\Sigma}$ to characterize the corresponding multi-task model.

2.9 Novel Settings in MTL

Most aforementioned MTL models assume that different tasks share the same feature representation and that each task is a binary classification problem if the label space is discrete. Moreover, the training data in each task are assumed to be stored in a data matrix. However, in some cases, these assumptions may not hold and in the following, we introduce some works whose problem settings violate these assumptions.

Instead of assuming that different tasks share an identical feature representation, Zhang and Yeung [111] consider a multi-database face recognition problem where face recognition in a database is treated as a task. Since different face databases have different image sizes, here naturally all the tasks do not lie in the same feature space in this application, leading to a heterogeneous-feature MTL problem. To tackle this heterogeneous-feature MTL problem, a multi-task discriminant analysis (MTDA) is proposed in [111] by first projecting different tasks into a common subspace and then learning a common projection in this subspace to discriminate different classes in different tasks. In [112], a latent probit model, a generative model, is proposed to generate data of different tasks in different feature spaces via a sparse transformation on a shared latent space and then to generate labels based on this latent space.

In many MTL classification problems and models, each task is explicitly or implicitly assumed to be a binary classification problem as each column in the parameter matrix $\mathbf{W}$ contains model parameters for the corresponding task. It is not difficult to see that many methods in the feature learning approach, low-rank approach, and decomposition approach can be directly extended to a general setting where each classification task can be a multi-class classification problem and correspondingly multiple columns in $\mathbf{W}$ contains model parameters of a multi-class classification task. Such direct extension is applicable since those methods only rely on the entire $\mathbf{W}$ or its rows as a media to share knowledge among tasks but not its columns. However, to the best of our knowledge, there is no theoretical or empirical study to investigate such direct extension. For most methods in the task clustering approach and task relation learning approach, such direct extension does not work since for multiple columns in $\mathbf{W}$ corresponding to one task, we do not know which one(s) can be used to represent this task. Therefore, the direct extension may not be the best solution to the general setting. In the following, we introduce four main approaches other than the direct extension to tackle the general setting in MTL where each classification task can be a multi-class classification problem. The first method is to transform the multi-class classification problem in each task into a binary classification problem. For example, multi-task metric learning [74], [113] transforms into an imbalanced binary classification problem where a pair of data points from the same class is treated as positive and that from different classes is negative. The second recipe is to utilize the characteristics of learners. For example, the linear discriminant analysis can handle binary and multi-class classification problems in a unified formulation and hence MTDA [111] can naturally handle them without changing the formulation. The third approach is to directly learn label correspondence among different tasks. In [114], two learning tasks, which share the training data, aim to maximize the mutual information to identify the correspondence between labels in different tasks. By assuming that all the tasks share the same label space, the last approach including [52], [71], [97] organizes the model parameters of all the tasks in a tensor where the model parameters of each task form a slice. Then the parameter tensor can be regularized by tensor trace norms [52] and the tensor norm prior [97], or factorized as a product of several low-rank matrices or tensors [71].

Most MTL methods assume that the training data in each task are stored in a data matrix. In some case, the training data exhibit a multi-modal structure in each task and hence they are represented in a tensor instead of a matrix. Multilinear multi-task methods proposed in [115], [116] can handle this situation by employing tensor norms as a generalization of the trace norm to do the regularization.

2.10 Discussions

Even though many MTL methods have been proposed, there are some issues which can be explored further. By minimizing the total training loss of all the tasks, the total generalization performance of MTL models is shown to be better than that of single-task models, but to the best of our knowledge, there is no guarantee that the MTL model will always perform better than the single-task counterpart on each task, which we think is an interesting issue to be studied in the future. In some situation, the performance of the MTL model on some task may be inferior to that of the single-task model. One reason for that is that there may exist some outlier tasks which are unrelated to other tasks and impair the performance. In this case, we can use the task clustering approach, task relation learning approach, and decomposition approach to help identify outlier tasks.

For asymmetric MTL which applies a MTL model trained on multiple tasks to a newly arrived task, one issue to be considered is that whether the trained MTL model is compatible with the new task. If the answer is yes, we can apply the MTL model to the new task and otherwise it is better to learn a single-task model for the new task. To achieve that, as a future direction, it is interesting to design a criterion that can quickly evaluate the applicability of the trained MTL model to the new task. Such criterion may relate the trained MTL model to some characteristics of the new task such as the training loss. Moreover, if this criterion has been devised, it can be extended to transfer learning whose setting is similar to asymmetric MTL.

3 MTL WITH OTHER LEARNING PARADIGMS

In the previous section, we review different MTL approaches for supervised learning tasks. In this section, we present some works on the combination of MTL with other learning paradigms in machine learning, including unsupervised learning such as clustering, semi-supervised learning, active learning, reinforcement learning, multi-view learning, and graphical models, to either improve the performance of supervised MTL further via other information such as unlabeled data or use MTL to help improve the performance of other learning paradigms.

In most applications, labeled data are expensive to collect but unlabeled data are abundant. So in some MTL applications, the training set of each task consists of both labeled and unlabeled data, hence we hope to exploit useful information contained in the unlabeled data to further improve the performance of MTL. In machine learning, semi-supervised learning and active learning are two ways to utilize unlabeled data but in different ways. Semi-supervised learning aims to exploit geometrical information contained in the unlabeled data, while active learning selects representative unlabeled data to query an oracle with the hope to reduce the labeling cost as much as possible. Hence semi-supervised learning and active learning can be combined with MTL, leading to three new learning paradigms including semi-supervised multi-task learning [117], [118], [119], multi-task active learning [120], [121], [122], and semi-supervised multi-task active learning [123]. Specifically, a semi-supervised multi-task classification model is proposed in [117], [118] to use random walk to exploit unlabeled data in each task and then cluster multiple tasks via a relaxed Dirichlet process. In [119], a semi-supervised multi-task Gaussian process for regression tasks, where different tasks are related via the hyperprior on the kernel parameters in Gaussian processes of all the tasks, is proposed to incorporate the unlabeled data into the design of the kernel function in each task to achieve the smoothness in the corresponding functional spaces. Different from these semi-supervised multi-task methods, multi-task active learning adaptively selects informative unlabeled data for multi-task learners and hence the selection criterion is the core research issue. Reichart et al. [120] believe that data instances to be selected should be as informative as possible for a set of tasks instead of only one task and hence they propose two protocols for multi-task active learning. In [121], the expected error reduction is used as a criterion where each task is modeled by a supervised latent Dirichlet allocation model. Inspired by multi-armed bandits which balance the trade-off between the exploitation and exploration, a selection strategy is proposed in [122] to consider both the risk of multi-task learner based on the trace norm regularization and the corresponding confidence bound. In [124], the MTRL method (i.e., problem (21)) is extended to the interactive setting where a human expect is enquired about partial orderings of pairwise task covariances based an inconsistency criterion. Moreover, Li et al. [123] combine semi-supervised learning and active learning to utilize unlabeled data for MTL by using the Fisher information as a criterion to select unlabeled data to acquire their labels with the semi-supervised multi-task classification model [117], [118] as the classifier for multiple tasks.

MTL achieves the performance improvement in not only supervised learning tasks but also unsupervised learning tasks such as clustering. In [125], a multi-task Bregman clustering method is proposed based on single-task Bregman clustering by using the earth mover distance to minimize distances between any pair of tasks in terms of cluster centers and then in [126], [127], an improved version of [125] and its kernel extension are proposed to avoid the negative effect caused by the regularizer in [125] by choosing the better one between single-task and multi-task Bregman clustering. In [128], a multi-task kernel k-means method is proposed by learning the kernel matrix via both MMD that is to minimize distances between any pair of tasks and the Laplacian regularization that helps identify a smooth kernel space. In [129], two proposed multi-task clustering methods are extensions of the MTFL and MTRL methods by treating labels as cluster indicators to be learned. In [130], the principle of MTL is incorporated into the subspace clustering by capturing correlations between data instances. In [131], a multi-task clustering method belonging to instance-based MTL is proposed to share data instances among different tasks. Different from these works, in [132], a derived generalization bound is used to select a subset from multiple unlabeled tasks to acquire labels to improve the generalization performance of all the tasks.

Reinforcement learning is a promising area in machine learning and has shown superior performance in many applications such as game playing (e.g., Atari and Go) and robotics. MTL can help boost the performance of reinforcement learning, leading to multi-task reinforcement learning. In [133] where an agent needs to solve a sequence of Markov decision processes (MDP), a hierarchical Bayesian infinite mixture model is used to model the distribution over MDPs and for each new MDP, previously learned distributions are used as an informative prior. In [134], a regionalized policy representation is introduced to characterize the behavior of an agent in each task and the Dirichlet process is placed over regionalized policy representations across multiple tasks to cluster tasks. In [135], the Gaussian process temporal-difference value function model is used for each task and a hierarchical Bayesian approach is to model the distribution over value functions in different tasks. Calandriello et al. [136] assume that parameter vectors of value functions in different tasks are jointly sparse and then extend the MTFS method with the $\ell_{2,1}$ regularization as well as the MTFL method to learn value functions in multiple tasks together. In [137], the proposed Actor-Mimic method combines both deep reinforcement learning and model compression techniques to train a policy network which can learn to act for multiple tasks. In [138], a model associated each subtask with a modular subpolicy is proposed to learn from policy sketches, which annotate tasks with sequences of named subtasks and provide information about high-level structural relationships among tasks. In [139], the problem of multi-task multi-agent reinforcement learning under the partial observability is addressed by distilling decentralized single-task policies into a unified policy across multiple tasks. In [140], a multi-task linearly solvable MDP is proposed to maintain a parallel distributed representation of tasks each of which enables an agent to draw on macro actions simultaneously. In [141], each task has its own policy which is constrained to be close to a shared policy that is trained by distillation to be the centroid of all task policies. In [142], a multi-task contextual bandit is introduced to leverage similarities in contexts among arms to improve the ability to predict rewards from contexts.

Multi-view learning assumes that each data point is associated with multiple sets of features where each set corresponds to a view and it usually exploits information contained in multiple views for supervised or unsupervised tasks. Multi-task multi-view learning extends multi-view learning to the multi-task setting where each task is a multi-view learning problem. Specifically, in [143], a graph-based method is proposed for multi-task multi-view classification problems. In a task, each view is enforced to be consistent with both other views and labels, while different tasks are expected to have similar predictions on views they share, making views a bridge to construct the task relatedness. In [144], both a regularized MTL method [75] and the MTRL method are applied to each view of different tasks and different views in a task are expected to achieve an agreement on unlabeled data. Different from [143], [144] which study the multi-task multi-view classification problem, in [145], [146], two multi-task multi-view clustering methods are proposed and both methods consider three factors: within-view-task clustering which conducts clustering on each view in a task, view relation learning which minimizes the disagreement among views in a task, and task relation learning which aims to learn a shared subspace for different tasks under a common view. The difference between these two methods is that the first method uses a bipartite graph co-clustering method for nonnegative data while another one adopts a semi-nonnegative matrix tri-factorization to cluster general data. Moreover, in multi-task multi-view learning, each task is usually supplied with both labeled and unlabeled data, hence this paradigm can also be viewed as another way to utilize unlabeled information for MTL.

Moreover, MTL can help learn more accurate structure in graphical models. In [147], an algorithm is proposed to learn Bayes network structures by assuming that different networks/tasks share similar structures via a common prior and then a heuristic search is used to find structures with high scores for all the tasks. With a similar idea, multiple Gaussian graphical models are jointly learned in [148] by assuming joint sparsity among precision matrices via the $\ell_{\infty,1}$ regularization. In [149], some domain knowledge about task relations is incorporated into the learning of multiple Bayesian networks in different tasks. By viewing the feature interaction matrix as a form of graphical models to model pairwise interactions between features, two models are proposed in [150] to learn a quadratical function, where the feature interaction matrix defines the quadratic term, for each task based on the $\ell_{2,1}$ and tensor trace norm regularizations, respectively.

4 Handling Big Data

Usually, each task in MTL has a limited number of training data, which are not so big. However, when the number of tasks is large, the total number of training data in all the tasks can be very big and hence a 'big' aspect in MTL is the number of tasks. Another 'big' aspect in MTL is the data dimensionality which can be very high. For a big number of tasks, we can either employ online MTL or devise parallel and distributed MTL methods to accelerate the learning process and when the data lie in a high-dimensional space, we can speedup the learning via feature selection, dimensionality reduction, and feature hashing to reduce the dimension without losing much useful information. In this section, we review some relevant works.

When the number of tasks is very big, we can devise some parallel MTL methods to speedup the learning process on multi-CPU or multi-GPU devices. As a representative formulation in feature-based MTL, problem (32) is easy to parallelize since when given the feature covariance matrix $\mathbf{\Theta}$, the learning of different tasks can be decoupled. However, for problem (34) in parameter-based MTL, the situation is totally different since even given the task covariance matrix $\mathbf{\Sigma}$, different tasks are still coupled, making the direct parallelization fail. In order to parallelize problem (34), Zhang [151] uses the FISTA algorithm to design a surrogate function for problem (34) with a given $\mathbf{\Sigma}$, where the surrogate function is decomposable with respect to tasks, leading to a parallel design based on different loss functions including the hinge, ϵ-insensitive and square losses. Moreover, online multi-task learning is also capable of handling this situation. In [152], [153], under a setting where all the tasks contribute toward a common goal, the relation between tasks is measured via a global loss function and several online algorithms are proposed to use absolute norms as the global loss function. In [154], online MTL algorithms are devised when the relatedness of all the m tasks is modeled by constraining that the m-tuple of actions for tasks needs to satisfy some hard constraints. In [155], perceptron-based online algorithms are proposed for multi-task binary classification problems where similarities among tasks are measured based on either the geometric closeness of the task reference vectors or the dimension of their spanned subspace. In [156], a recursive Bayesian online algorithm based on Gaussian processes is devised to update both estimations and confidence intervals when new data points arrive sequentially. In [157], an online version of the MTRL method is proposed to update both the model parameters and task covariance in a sequential way. Moreover, training data can distribute at different places, making distributed MTL become important. In [158], a communication-efficient distributed MTL algorithm, where each machine learns a task, based on the debiased Lasso is proposed for MTL to learn jointly sparse features in the high-dimensional setting. In [159], the MTRL method (i.e., problem (21)) is extended to the distributed setting based on the stochastic dual coordinate ascent method. In [160], to protect the privacy of data, a privacy-preserving distributed MTL method is proposed based on a privacy-preserving proximal gradient algorithm with asynchronous updates. In [161], a decentralized distributed online multi-task algorithm is proposed.

For high-dimensional data in MTL, we can use multi-task feature selection methods to reduce the dimension and extend single-task dimension reduction techniques to the multi-task setting as did in [111]. Another option is to use the feature hashing and in [162], multiple hashing functions are proposed to accelerate the joint learning of multiple tasks.

5 Applications

MTL has many applications in various areas including computer vision, bioinformatics, health informatics, speech, natural language processing, web applications, ubiquitous computing, and so on. In the following, we introduce them in a chronological order.

5.1 Computer Vision

In [163], a hierarchical kernel stick-breaking process is proposed for multiple image segmentation where the segmentation for an image is treated as a task and modeled by a kernel stick-breaking process, while a Dirichlet process is used to cluster tasks. In [164], a boosted multi-task method is proposed for face verification where different tasks share base learners in the boosting method. In [165], a multi-task warped Gaussian process is proposed for personalized age estimation, where each task corresponds to a person and different tasks share the kernel parameters and warped function but with different noise levels. In [166], a multi-task feature selection model based on the $\ell_{2,1}$ norm is proposed for multi-cue face recognition and object categorization. In [167], a multi-task feature selection method based on both the $\ell_{2,1}$ and ℓ_1 norms is proposed to identify brain imaging predictors for memory performance. In [168], a multi-task low-rank subspace clustering, where different tasks are related via the structural sparsity among the spanning matrices in subspace clustering, is proposed for image segmentation and a similar model is used in [169] for saliency detection. In [170], [171], a multi-task dictionary is learned for visual tracking via the $\ell_{p,1}$ sparsity. In [172], a multi-task decomposition dictionary learning method that follows the idea of [103] is applied to multi-view tracking. In [173], a multi-task decomposition model, where each component matrix is regularized by a priori information organized in a graph, is proposed for head pose classification in an uncontrolled environment. In [174], a multi-task sparse model based on the Beta process is proposed to learn dictionaries for action recognition. In [17], a tasks-constrained deep convolutional network is proposed for facial landmark detection by sharing hidden layers with auxiliary tasks including head pose estimation, gender classification, age estimation, facial expression recognition, and facial attribute inference. In [175], a multi-task model is proposed to learn a low-dimensional feature transformation for scene classification. In [176], a multi-task convolutional neural network (CNN), which has individual CNNs for each task and fuses different CNNs in a common layer via a sparse transformation, is proposed for image-based multi-label attribute prediction. In [177], similar to [100], a decomposition model with one component modeling low-rank via the trace norm and another capturing sparsity via the ℓ_1 norm is proposed for multi-camera person re-identification. In [178], a multi-task deep model is proposed to rotate facial images to a target pose with an auxiliary task as the reconstruction of original images based on generated images. In [18], to select thumbnails for videos, a deep model is proposed to utilize the available sematic information such as titles and descriptions to align the embeddings of video thumbnails to those of sematic information and the auxiliary task is to learn from the click-through image data. In [179], a recurrent neural network (RNN) is used for immediacy prediction and the output layer has multiple units to estimate the interaction, distance, stand orientation, relative orientation, and pose estimation. In [21], a deep convolutional neural network is proposed for pose estimation by sharing hidden layers with auxiliary tasks including body-part and joint-point detections. In [180], the Go-MTL method is generalized to the multilinear multi-task setting for person-specific facial action unit prediction, where each facial action unit is treated as a task and in each task, a function is learned for a person.

5.2 Bioinformatics and Health Informatics

In [181], several regularized multi-task models are proposed to utilize the hierarchical structure among tasks to model organisms. In [182], a sparse multi-task regressor based on the ℓ_1 norm regularization is proposed to identify a common mechanism of responses to therapeutic targets. In [183], a multi-task model to detect commonly useful features based on significant tests is proposed for cross-platform siRNA efficacy prediction. In [184], the MTFS method is used to detect causal genetic markers through a joint association analysis of multiple populations. In [185], a multi-task model by sharing a Gaussian prior on model parameters of different tasks is to construct personalized brain-computer interfaces. In [186], two multi-task multi-kernel methods are proposed for MHC-I binding prediction and splice-site prediction. In [187], two multi-task methods proposed in [11], [73] are used for protein subcellular location prediction. In [188], a multi-task method based on a temporal group Lasso and the $\ell_{2,1}$ norm regularization is proposed for the mini mental state examination and Alzheimers disease assessment scale cognitive subscale. In [189], a ProDiGe method is proposed to share information about known disease genes across diseases by learning from positive and unlabeled data for prioritization of disease genes. In [190], a sparse Bayesian model, which learns correlations between features for all the tasks based on the automatic relevance determination, is proposed to predict cognitive outcomes from neuroimaging measures for the Alzheimers disease. In [191], the identification of longitudinal phenotypic markers for the Alzheimers disease progression prediction is formulated as a multi-task time-series problem where at each time point, a learner is associated with parameters organized in a matrix and the parameter tensor consisting parameter matrices at all the time points is assumed to be both group sparse and low-rank. In [19], biological images and natural images in, for example, the Imagenet dataset are jointly trained in two neural networks with the hope of transferring useful information in abundant natural images to limited biological images to improve its performance. In [192], a task clustering method is proposed to learn a personalized medical model by factorizing the parameter matrix into two low-rank and sparse component matrices with a graph Laplacian regularizer to enforce the smoothness. In [193], the survival analysis problem is formulated as a multi-task classification problem under an assumption that once an event occurs then it will not occur again, and then the MTFS method is extended to solve the resulting problem. In [194], several multi-task models corresponding to problems (4), (16) and (27) are employed for multiple genetic trait prediction.

5.3 Speech and Natural Language Processing

In [195], a multi-task stacked deep neural network, which consists of multiple neural networks where the former neural network feeds the output of its top-most hidden layer as an input to the next neural network, is proposed for speech synthesis and each neural network has two output units, one for the main task and the other for an auxiliary task, by sharing the hidden layers between the two tasks. In [196], a multi-task deep neural network is used to model cepstra and log amplitudes as primary and auxiliary tasks for sinusoidal speech synthesis.

In [197], a multi-task time-decay neural network is proposed to jointly learn six NLP tasks, including part-of-speech tagging, chunking, named entity recognition, semantic role labeling, language modeling, and identification of semantically related words, and unlabeled data are used to help train the language model. In [198], a multi-task model consisting of a general sparse learner for all the tasks and task-specific sparse learners which are regularized by pre-computed task similarities is proposed for multi-domain sentiment classification. In [20], a multi-task RNN with shared hidden layers among tasks is used for multi-domain dialog state tracking. In [199], three multi-task encoder-decoder architectures are proposed for several applications. The first architecture, where the encoder is shared by all the tasks but decoders are task-specific, is used for machine translation and syntactic parsing. The second one, where each task has its own encoder but the decoder is shared by all the tasks, is proposed for machine translation and image caption generation. The last one, where multiple encoders and decoders are shared among tasks, is for machine translation. Some variants of the MTFS method are proposed in [200], [201] to analyze contents in microblogs at different locations for event forecasting such as civil unrest and influenza outbreak, where learning at each location is treated as a classification task.

5.4 Web Applications

In [202], a multi-task boosting method, where different tasks share a feature representation, is proposed for learning to rank in web search. In [203], a multi-task boosting, where each task has a common classifier and a task-specific classifier in a way similar to [73], is proposed for web search ranking. In [204], a multi-domain collaborative filtering method based on matrix factorization is proposed to utilize rating matrices in multiple related domains to help each other and similar to [4], [5], a matrix-variate normal prior is placed on latent user features to learn correlations between each pair of domains. In [205], the MTFS method with an additional ℓ_1 norm regularization is used for behavioral targeting. In [206], the MTRL method is extended to consider the hierarchical structure as well as structural sparsity for conversion maximization in display advertising.

5.5 Ubiquitous Computing

In [207], a multi-task neural network, which shares hidden layers for different tasks, is used to predict yearly returns of stocks. In [208], a multi-task model, which learns a low-rank transformation and enforces model parameters of different tasks to be similar to each other in the transformed space, is proposed for multi-device localization. In [209], [210], the inverse dynamics problem for robotics is solved from the perspective of MTL. In [211], a multi-task trajectory regression method, which encodes the spatial and temporal information via the Laplacian and fused Lasso regularizations, is proposed to estimate road travel costs on road networks. In [212], a multi-task decomposition trajectory regression method, which captures the spatial and temporal information via the Laplacian regularization and identifies outliers via the $\ell_{\infty,1}$ regularization, is proposed for predicting the travel time of arbitrary trajectories on road networks. In [213], a multi-task low-rank method is proposed for multi-location climate prediction by decomposing the parameter matrix as the product of two low-rank and sparse component matrices. A multi-task model with a tree-structured regularization and the $\ell_{2,1}$ regularization is proposed in [214] to recognition traffic signs. In [215], an online MTL algorithm with a restart strategy is proposed for ensemble forecasting on the seasonal soil moisture.

5.6 Discussions

In previous sections, many application-dependent MTL models have been proposed to solve different application problems. Even though those models are different from each other and application problems they solved are different, there are some characteristics in respective areas. For example, in computer vision, the image and video data usually lie in a high-dimensional space and hence to improve the learning efficiency, the feature selection approach is a good choice to identify low-dimensional representation. Moreover, deep models exhibit good performance in computer vision problems, making deep MTL models popular. In bioinformatics and health informatics, the interpretability of the learning model is more important in some sense. Therefore, the feature selection approach is widely used in this area as this approach can identify useful features. In speech and natural language processing, the data exhibit a sequential structure, which makes RNN-based deep models play a dominate role. The data in web applications is of large scale and hence this area favors simple shallow models or their ensembles based on boosting which can improve the performance further. In ubiquitous computing, since the nature of application problems is diverse, all the approaches can be applicable. In summary, in these application areas, the feature selection approach and the deep feature transformation approach are more popular since the former can reduce the data dimension and provide a better interpretability and the later can lead to good performance by learning powerful feature representations.

6 Theoretical Analyses

As well as designing MTL models, there are many works to study theoretical aspects of MTL and here we review them.

The generalization bound, which is to upper-bound the generalization performance in terms of the training loss, model complexity and confidence, is core in learning theory since it can identify the learnability and induce the sample complexity. In order to derive the generalization bound in single-task learning, there are usually four main tools, including Vapnik-Chervonenkis (VC) dimension, covering number, stability, and Rademacher complexity. In MTL, these four tools are also helpful to derive generalization bounds.

In [216], the first generalization bound for MTL is presented based on the VC dimension and covering number to characterize the relation between the generalization performance and the empirical performance. Ben-David et al. [217], [218], [219] study the generalization bound of MTL based on the VC dimension by assuming that data distributions of different tasks can be transformed via some known family of functions. Ando and Zhang [44] use the covering number to analyze the generalization performance of a low-rank model formulated in problem (11). Given a pool of hypotheses, an algorithm is proposed in [54] to map each task to a hypothesis in the pool to cluster tasks and the VC dimension is used to derive the generalization bound. In [220], Pentina and Ben-David present a generalization bound for learning multiple kernels in multi-task large-margin classifiers such as SVM based on the covering number. Zhang [221] extends the conventional stability to the MTL setting by proving a generalized McDiarmids inequality and uses the proposed multi-task stability to analyze the trace norm regularization [49] in the low-rank approach and a decomposition model in [101] (i.e., problem (26) with Eq. (27)). In [222], Maurer analyzes the generalization bound of a MTL method based on the Rademacher complexity, which learns a common feature transformation for all the tasks and is a special case of problem (11) without $\mathbf{U}$, and then in [223] further analyzes generalization bounds of regularized MTL models based on the given task relations [73], [75] and Schatten-norm-regularized MTL models which are generalizations of the trace norm regularization. Kakade et al. [224] analyze both batch and online multi-task models with (group) sparse regularizations and trace norm regularization based on a famous inequality originally derived in online learning and the Rademacher complexity. For the multi-task sparse coding method shown in problem (3), Maurer et al. [14] analyze its generalization bound with the use of the Rademacher complexity. In [225], a dimension-independent generalization bound of the trace norm regularization (i.e., problem (14)) is derived based on recent advances in random matrices and the Rademacher complexity. The Gaussian average, which is related to the Rademacher complexity, is used in [226] to derive the generalization bound of a general MTL model which can learn a common feature transformation for all the tasks. Moreover, as well as the four tools to upper-bound the generalization performance, the Kolmogorov complexity, an analysis tool from information theory, is used in [227] to bound the generalization performance.

Besides generalization bounds, there are some works to discuss other theoretical problems in MTL. For example, Argyriou et al. [228], [229] discuss conditions for regularized MTL algorithms that representer theorems hold. Several studies [230], [231], [232] investigate conditions to well recover true features for multi-task feature selection methods. Moreover, Solnon et al. [109] show that the key element for an optimal calibration is the covariance matrix of the noise between different tasks and then based on this analysis, they present an algorithm to estimate the covariance matrix based on the minimal penalty.

7 Conclusions

In this paper, we survey different aspects of MTL. First, we give a classification of MTL models into five main approaches, including feature learning approach, low-rank approach, task clustering approach, task relation learning approach, and decomposition approach, and discuss their characteristics. Then we review the combinations of MTL with other learning paradigms including unsupervised learning, semi-supervised learning, active learning, reinforcement learning, multi-view learning, and graphical models. The online, parallel and distributed MTL models as well as dimensionality reduction and feature hashing are discussed to speedup the learning process when there are a large number of tasks or data lie in a high-dimensional space. The applications of MTL in various areas are introduced to show the usefulness of MTL and theoretical aspects of MTL are discussed.

In the future studies, there are several issues to be addressed. Firstly, outlier tasks, which are unrelated to other tasks, are well known to hamper the performance of all the tasks when learning them jointly. There are some methods to alleviate negative effects outlier tasks bring. However, there lacks principled ways and theoretical analyses to study the resulting negative effects. In order to make MTL safe to be used by human, this is an important issue and needs more studies.

Secondly, deep learning has become a dominant approach in many areas and several multi-task deep models have been proposed as reviewed in Sections 2 and 5. As discussed, most of them are just to share hidden layers. This approach is powerful when all the tasks are related but it is vulnerable to noisy and outlier tasks which can deteriorate the performance dramatically. We believe that it is desirable to design flexible and robust multi-task deep models.

Lastly, existing studies mainly focus on supervised learning tasks but only a few ones are on other tasks such as unsupervised learning, semi-supervised learning, active learning, multi-view learning, and reinforcement learning tasks. It is natural to adapt or extend the five approaches introduced in Section 2 to those non-supervised learning tasks but we think that the adaptation and extension require more efforts to design appropriate models. Moreover, it is worth trying to apply MTL to other areas in artificial intelligence such as logic and planning to broaden its application scopes.

REFERENCES

[1] R. Caruana, "Multitask learning," *MLJ*, vol. 28, no. 1, pp. 41–75, 1997.

[2] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE TKDE*, vol. 22, no. 10, pp. 1345–1359, 2010.

[3] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram, "Multi-task learning for classification with Dirichlet process priors," *JMLR*, vol. 8, pp. 35–63, 2007.

[4] Y. Zhang and D.-Y. Yeung, "A convex formulation for learning task relationships in multi-task learning," in *UAI*, 2010.

[5] Y. Zhang and D.-Y. Yeung, "A regularization approach to learning task relationships in multitask learning," *ACM TKDD*, vol. 8, no. 3, p. article 12, 2014.

[6] Y. Zhang and Q. Yang, "An overview of multi-task learning," *National Science Review*, vol. 5, no. 1, pp. 30–43, 2018.

[7] X. Yang, S. Kim, and E. P. Xing, "Heterogeneous multitask learning with joint sparsity constraints," in *NIPS*, 2009.

[8] S. Bickel, J. Bogojeska, T. Lengauer, and T. Scheffer, "Multi-task learning for HIV therapy screening," in *ICML*, 2008.

[9] X. Liao and L. Carin, "Radial basis function network for multi-task learning," in *NIPS*, 2005.

[10] D. L. Silver, R. Poirier, and D. Currie, "Inductive transfer with context-sensitive neural networks," *MLJ*, vol. 73, no. 3, pp. 313–336, 2008.

[11] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in *NIPS*, 2006.

[12] A. Argyriou, T. Evgeniou, and M. Pontil, "Convex multi-task feature learning," *MLJ*, vol. 73, no. 3, pp. 243–272, 2008.

[13] A. Argyriou, C. A. Micchelli, M. Pontil, and Y. Ying, "A spectral regularization framework for multi-task structure learning," in *NIPS*, 2007.

[14] A. Maurer, M. Pontil, and B. Romera-Paredes, "Sparse coding for multitask and transfer learning," in *ICML*, 2013.

[15] J. Zhu, N. Chen, and E. P. Xing, "Infinite latent SVM for classification and multi-task learning," in *NIPS*, 2011.

[16] M. K. Titsias and M. Lázaro-Gredilla, "Spike and slab variational inference for multi-task and multiple kernel learning," in *NIPS*, 2011.

[17] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in *ECCV*, 2014.

[18] W. Liu, T. Mei, Y. Zhang, C. Che, and J. Luo, "Multi-task deep visual-semantic embedding for video thumbnail selection," in *CVPR*, 2015.

[19] W. Zhang, R. Li, T. Zeng, Q. Sun, S. Kumar, J. Ye, and S. Ji, "Deep model based transfer and multi-task learning for biological image analysis," in *KDD*, 2015.

[20] N. Mrksic, D. Ó. Séaghdha, B. Thomson, M. Gasic, P. Su, D. Vandyke, T. Wen, and S. J. Young, "Multi-domain dialog state tracking using recurrent neural networks," in *ACL*, 2015.

[21] S. Li, Z. Liu, and A. B. Chan, "Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network," *IJCV*, vol. 113, no. 1, pp. 19–36, 2015.

[22] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *CVPR*, 2016.

[23] P. Liu, X. Qiu, and X. Huang, "Adversarial multi-task learning for text classification," in *ACL*, 2017.

[24] G. Obozinski, B. Taskar, and M. Jordan, "Multi-task feature selection," tech. rep., University of California, Berkeley, 2006.

[25] G. Obozinski, B. Taskar, and M. Jordan, "Joint covariate selection and joint subspace selection for multiple classification problems," *Statistics and Computing*, vol. 20, no. 2, pp. 231–252, 2010.

[26] J. Liu, S. Ji, and J. Ye, "Multi-task feature learning via efficient $l_{2,1}$-norm minimization," in *UAI*, 2009.

[27] S. Lee, J. Zhu, and E. P. Xing, "Adaptive multi-task lasso: With application to eQTL detection," in *NIPS*, 2010.

[28] N. S. Rao, C. R. Cox, R. D. Nowak, and T. T. Rogers, "Sparse overlapping sets lasso for multitask learning and its application to fMRI analysis," in *NIPS*, 2013.

[29] P. Gong, J. Zhou, W. Fan, and J. Ye, "Efficient multi-task feature learning with calibration," in *KDD*, 2014.

[30] J. Wang and J. Ye, "Safe screening for multi-task feature learning with multiple data matrices," in *ICML*, 2015.

[31] H. Liu, M. Palatucci, and J. Zhang, "Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery," in *ICML*, 2009.

[32] J. E. Vogt and V. Roth, "A complete analysis of the $\ell_{1,p}$ group-lasso," in *ICML*, 2012.

[33] P. Gong, J. Ye, and C. Zhang, "Multi-stage multi-task feature learning," *JMLR*, vol. 14, pp. 2979–3010, 2013.

[34] A. C. Lozano and G. Swirszcz, "Multi-level lasso for sparse multi-task regression," in *ICML*, 2012.

[35] X. Wang, J. Bi, S. Yu, and J. Sun, "On multiplicative multitask feature learning," in *NIPS*, 2014.

[36] L. Han, Y. Zhang, G. Song, and K. Xie, "Encoding tree sparsity in multi-task learning: A probabilistic framework," in *AAAI*, 2014.

[37] T. Jebara, "Multi-task feature and kernel selection for SVMs," in *ICML*, 2004.

[38] T. Jebara, "Multitask sparsity via maximum entropy discrimination," *JMLR*, vol. 12, pp. 75–110, 2011.

[39] S. Kim and E. P. Xing, "Tree-guided group lasso for multi-task regression with structured sparsity," in *ICML*, 2010.

[40] Y. Zhou, R. Jin, and S. C. H. Hoi, "Exclusive lasso for multi-task feature selection," in *AISTATS*, 2010.

[41] Y. Zhang, D.-Y. Yeung, and Q. Xu, "Probabilistic multi-task feature selection," in *NIPS*, 2010.

[42] D. Hernández-Lobato and J. M. Hernández-Lobato, "Learning feature selection dependencies in multi-task learning," in *NIPS*, 2013.

[43] D. Hernández-Lobato, J. M. Hernández-Lobato, and Z. Ghahramani, "A probabilistic model for dirty multi-task feature selection," in *ICML*, 2015.

[44] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *JMLR*, vol. 6, pp. 1817–1853, 2005.

[45] J. Chen, L. Tang, J. Liu, and J. Ye, "A convex formulation for learning shared structures from multiple tasks," in *ICML*, 2009.

[46] A. Agarwal, H. Daumé III, and S. Gerber, "Learning multiple tasks using manifold regularization," in *NIPS*, 2010.

[47] J. Zhang, Z. Ghahramani, and Y. Yang, "Learning multiple related tasks using latent independent component analysis," in *NIPS*, 2005.

[48] J. Zhang, Z. Ghahramani, and Y. Yang, "Flexible latent variable models for multi-task learning," *MLJ*, vol. 73, no. 3, pp. 221–242, 2008.

[49] T. K. Pong, P. Tseng, S. Ji, and J. Ye, "Trace norm regularization: Reformulations, algorithms, and multi-task learning," *SIAM Journal on Optimization*, vol. 20, no. 6, pp. 3465–3489, 2010.

[50] L. Han and Y. Zhang, "Multi-stage multi-task learning with reduced rank," in *AAAI*, 2016.

[51] A. M. McDonald, M. Pontil, and D. Stamos, "Spectral k-support norm regularization," in *NIPS*, 2014.

[52] Y. Yang and T. M. Hospedales, "Trace norm regularised deep multi-task learning," in *ICLR, Workshop Track*, 2017.

[53] S. Thrun and J. O'Sullivan, "Discovering structure in multiple learning tasks: The TC algorithm," in *ICML*, 1996.

[54] K. Crammer and Y. Mansour, "Learning multiple tasks using shared hypotheses," in *NIPS*, 2012.

[55] B. Bakker and T. Heskes, "Task clustering and gating for Bayesian multitask learning," *JMLR*, vol. 4, pp. 83–99, 2003.

[56] K. Yu, V. Tresp, and A. Schwaighofer, "Learning Gaussian processes from multiple tasks," in *ICML*, 2005.

[57] S. Yu, V. Tresp, and K. Yu, "Robust multi-task learning with t-processes," in *ICML*, 2007.

[58] W. Lian, R. Henao, V. Rao, J. E. Lucas, and L. Carin, “A multitask point process predictive model,” in *ICML*, 2015.

[59] Y. Xue, D. B. Dunson, and L. Carin, “The matrix stick-breaking process for flexible multi-task learning,” in *ICML*, 2007.

[60] H. Li, X. Liao, and L. Carin, “Nonparametric Bayesian feature selection for multi-task learning,” in *ICASSP*, 2011.

[61] Y. Qi, D. Liu, D. B. Dunson, and L. Carin, “Multi-task compressive sensing with Dirichlet process priors,” in *ICML*, 2008.

[62] K. Ni, L. Carin, and D. B. Dunson, “Multi-task learning for sequential data via iHMMs and the nested Dirichlet process,” in *ICML*, 2007.

[63] K. Ni, J. W. Paisley, L. Carin, and D. B. Dunson, “Multi-task learning for analyzing and sorting large databases of sequential data,” *IEEE TSP*, vol. 56, no. 8, pp. 3918–3931, 2008.

[64] A. Passos, P. Rai, J. Wainer, and H. Daumé III, “Flexible modeling of latent task structures in multitask learning,” in *ICML*, 2012.

[65] L. Jacob, F. R. Bach, and J.-P. Vert, “Clustered multi-task learning: A convex formulation,” in *NIPS*, 2008.

[66] Z. Kang, K. Grauman, and F. Sha, “Learning with whom to share in multi-task feature learning,” in *ICML*, 2011.

[67] L. Han and Y. Zhang, “Learning multi-level task groups in multi-task learning,” in *AAAI*, 2015.

[68] A. Barzilai and K. Crammer, “Convex multi-task learning by clustering,” in *AISTATS*, 2015.

[69] Q. Zhou and Q. Zhao, “Flexible clustered multi-task learning by learning representative tasks,” *IEEE TPAMI*, vol. 38, no. 2, pp. 266–278, 2016.

[70] A. Kumar and H. Daumé III, “Learning task grouping and overlap in multi-task learning,” in *ICML*, 2012.

[71] Y. Yang and T. M. Hospedales, “Deep multi-task representation learning: A tensor factorisation approach,” in *ICLR*, 2017.

[72] J. Zhou, J. Chen, and J. Ye, “Clustered multi-task learning via alternating structure optimization,” in *NIPS*, 2011.

[73] T. Evgeniou and M. Pontil, “Regularized multi-task learning,” in *KDD*, 2004.

[74] S. Parameswaran and K. Q. Weinberger, “Large margin multi-task metric learning,” in *NIPS*, 2010.

[75] T. Evgeniou, C. A. Micchelli, and M. Pontil, “Learning multiple tasks with kernel methods,” *JMLR*, vol. 6, pp. 615–637, 2005.

[76] T. Kato, H. Kashima, M. Sugiyama, and K. Asai, “Multi-task learning via conic programming,” in *NIPS*, 2007.

[77] T. Kato, H. Kashima, M. Sugiyama, and K. Asai, “Conic programming for multitask learning,” *IEEE TKDE*, vol. 22, no. 7, pp. 957–968, 2010.

[78] N. Görnitz, C. Widmer, G. Zeller, A. Kahles, S. Sonnenburg, and G. Rätsch, “Hierarchical multitask structured output learning for large-scale sequence segmentation,” in *NIPS*, 2011.

[79] E. V. Bonilla, K. M. A. Chai, and C. K. I. Williams, “Multi-task Gaussian process prediction,” in *NIPS*, 2007.

[80] K. M. A. Chai, “Generalization errors and learning curves for regression with multi-task Gaussian processes,” in *NIPS*, 2009.

[81] Y. Zhang and D.-Y. Yeung, “Multi-task learning using generalized t process,” in *AISTATS*, 2010.

[82] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, “A kernel method for the two-sample-problem,” in *NIPS*, 2006.

[83] Y. Zhang and D.-Y. Yeung, “Multi-task boosting by exploiting task relationships,” in *ECMLPKDD*, 2012.

[84] Y. Zhang and D.-Y. Yeung, “Multilabel relationship learning,” *ACM TKDD*, vol. 7, no. 2, p. article 7, 2013.

[85] F. Dinuzzo, C. S. Ong, P. V. Gehler, and G. Pillonetto, “Learning output kernels with block coordinate descent,” in *ICML*, 2011.

[86] C. Ciliberto, Y. Mroueh, T. A. Poggio, and L. Rosasco, “Convex learning of multiple tasks and their structure,” in *ICML*, 2015.

[87] C. Ciliberto, L. Rosasco, and S. Villa, “Learning multiple visual tasks while discovering their structure,” in *CVPR*, 2015.

[88] P. Jawanpuria, M. Lapin, M. Hein, and B. Schiele, “Efficient output kernel learning for multiple tasks,” in *NIPS*, 2015.

[89] Y. Zhang and Q. Yang, “Learning sparse task relations in multi-task learning,” in *AAAI*, 2017.

[90] Y. Zhang and J. G. Schneider, “Learning multiple tasks with a sparse matrix-normal penalty,” in *NIPS*, 2010.

[91] C. Archambeau, S. Guo, and O. Zoeter, “Sparse Bayesian multi-task learning,” in *NIPS*, 2011.

[92] P. Rai, A. Kumar, and H. Daumé III, “Simultaneously leveraging output and task structures for multiple-output regression,” in *NIPS*, 2012.

[93] Y. Zhang and D.-Y. Yeung, “Learning high-order task relationships in multi-task learning,” in *IJCAI*, 2013.

[94] M. Yang, Y. Li, and Z. Zhang, “Multi-task learning with Gaussian matrix generalized inverse Gaussian model,” in *ICML*, 2013.

[95] B. Rakitsch, C. Lippert, K. M. Borgwardt, and O. Stegle, “It is all in the noise: Efficient multi-task Gaussian process inference with structured residuals,” in *NIPS*, 2013.

[96] A. R. Goncalves, F. J. V. Zuben, and A. Banerjee, “Multi-task sparse structure learning with Gaussian copula models,” *JMLR*, vol. 17, pp. 1–30, 2016.

[97] M. Long, Z. Cao, J. Wang, and P. S. Yu, “Learning multiple tasks with multilinear relationship networks,” in *NIPS*, 2017.

[98] Y. Zhang, “Heterogeneous-neighborhood-based multi-task local learning algorithms,” in *NIPS*, 2013.

[99] G. Lee, E. Yang, and S. J. Hwang, “Asymmetric multi-task learning based on task relatedness and loss,” in *ICML*, 2016.

[100] A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan, “A dirty model for multi-task learning,” in *NIPS*, 2010.

[101] J. Chen, J. Liu, and J. Ye, “Learning incoherent sparse and low-rank patterns from multiple tasks,” in *KDD*, 2010.

[102] J. Chen, J. Zhou, and J. Ye, “Integrating low-rank and group-sparse structures for robust multi-task learning,” in *KDD*, 2011.

[103] P. Gong, J. Ye, and C. Zhang, “Robust multi-task feature learning,” in *KDD*, 2012.

[104] W. Zhong and J. T. Kwok, “Convex multitask learning with flexible task clusters,” in *ICML*, 2012.

[105] A. Zweig and D. Weinshall, “Hierarchical regularization cascade for joint learning,” in *ICML*, 2013.

[106] L. Han and Y. Zhang, “Learning tree structure in multi-task learning,” in *KDD*, 2015.

[107] P. Jawanpuria and J. S. Nath, “A convex feature learning formulation for latent task structure discovery,” in *ICML*, 2012.

[108] B. Gong, Y. Shi, F. Sha, and K. Grauman, “Geodesic flow kernel for unsupervised domain adaptation,” in *CVPR*, 2012.

[109] M. Solnon, S. Arlot, and F. R. Bach, “Multi-task regression using minimal penalties,” *JMLR*, vol. 13, pp. 2773–2812, 2012.

[110] Y. Zhang, Y. Wei, and Q. Yang, “Learning to multitask,” *CoRR, abs/1805.07541*, 2018.

[111] Y. Zhang and D.-Y. Yeung, “Multi-task learning in heterogeneous feature spaces,” in *AAAI*, 2011.

[112] S. Han, X. Liao, and L. Carin, “Cross-domain multitask learning with latent probit models,” in *ICML*, 2012.

[113] P. Yang, K. Huang, and C. Liu, “Geometry preserving multi-task metric learning,” *MLJ*, vol. 92, no. 1, pp. 133–175, 2013.

[114] N. Quadrianto, A. J. Smola, T. S. Caetano, S. V. N. Vishwanathan, and J. Petterson, “Multitask learning without label correspondences,” in *NIPS*, 2010.

[115] B. Romera-Paredes, H. Aung, N. Bianchi-Berthouze, and M. Pontil, “Multilinear multitask learning,” in *ICML*, 2013.

[116] K. Wimalawarne, M. Sugiyama, and R. Tomioka, “Multitask learning meets tensor factorization: Task imputation via convex optimization,” in *NIPS*, 2014.

[117] Q. Liu, X. Liao, and L. Carin, “Semi-supervised multitask learning,” in *NIPS*, 2007.

[118] Q. Liu, X. Liao, H. Li, J. R. Stack, and L. Carin, “Semisupervised multitask learning,” *IEEE TPAMI*, vol. 31, no. 6, pp. 1074–1086, 2009.

[119] Y. Zhang and D. Yeung, “Semi-supervised multi-task regression,” in *ECMLPKDD*, 2009.

[120] R. Reichart, K. Tomanek, U. Hahn, and A. Rappoport, “Multi-task active learning for linguistic annotations,” in *ACL*, 2008.

[121] A. Acharya, R. J. Mooney, and J. Ghosh, “Active multitask learning using both latent and supervised shared topics,” in *SDM*, 2014.

[122] M. Fang and D. Tao, “Active multi-task learning via bandits,” in *SDM*, 2015.

[123] H. Li, X. Liao, and L. Carin, “Active learning for semi-supervised multi-task learning,” in *ICASSP*, 2009.

[124] K. Lin and J. Zhou, “Interactive multi-task relationship learning,” in *ICDM*, 2016.

[125] J. Zhang and C. Zhang, “Multitask Bregman clustering,” in *AAAI*, 2010.

[126] X. Zhang and X. Zhang, “Smart multi-task Bregman clustering and multi-task kernel clustering,” in *AAAI*, 2013.

[127] X. Zhang, X. Zhang, and H. Liu, “Smart multitask Bregman clustering and multitask kernel clustering,” *ACM TKDD*, vol. 10, no. 1, pp. 8:1–8:29, 2015.

[128] Q. Gu, Z. Li, and J. Han, “Learning a kernel for multi-task clustering,” in *AAAI*, 2011.

[129] X. Zhang, “Convex discriminative multitask clustering,” *IEEE TPAMI*, vol. 37, no. 1, pp. 28–40, 2015.

[130] Y. Wang, D. P. Wipf, Q. Ling, W. Chen, and I. J. Wassell, “Multi-task learning for subspace segmentation,” in *ICML*, 2015.

[131] X. Zhang, X. Zhang, and H. Liu, "Self-adapted multi-task clustering," in *IJCAI*, 2016.

[132] A. Pentina and C. H. Lampert, "Multi-task learning with labeled and unlabeled tasks," in *ICML*, 2017.

[133] A. Wilson, A. Fern, S. Ray, and P. Tadepalli, "Multi-task reinforcement learning: A hierarchical Bayesian approach," in *ICML*, 2007.

[134] H. Li, X. Liao, and L. Carin, "Multi-task reinforcement learning in partially observable stochastic environments," *JMLR*, vol. 10, pp. 1131–1186, 2009.

[135] A. Lazaric and M. Ghavamzadeh, "Bayesian multi-task reinforcement learning," in *ICML*, 2010.

[136] D. Calandriello, A. Lazaric, and M. Restelli, "Sparse multi-task reinforcement learning," in *NIPS*, 2014.

[137] E. Parisotto, J. Ba, and R. Salakhutdinov, "Actor-mimic: Deep multitask and transfer reinforcement learning," in *ICLR*, 2016.

[138] J. Andreas, D. Klein, and S. Levine, "Modular multitask reinforcement learning with policy sketches," in *ICML*, 2017.

[139] S. Omidshafiei, J. Pazis, C. Amato, J. P. How, and J. Vian, "Deep decentralized multi-task multi-agent reinforcement learning under partial observability," in *ICML*, 2017.

[140] A. M. Saxe, A. C. Earle, and B. Rosman, "Hierarchy through composition with multitask LMDPs," in *ICML*, 2017.

[141] Y. W. Teh, V. Bapst, W. M. Czarnecki, J. Quan, J. Kirkpatrick, R. Hadsell, N. Heess, and R. Pascanu, "Distral: Robust multitask reinforcement learning," in *NIPS*, 2017.

[142] A. A. Deshmukh, Ü. Dogan, and C. Scott, "Multi-task learning for contextual bandits," in *NIPS*, 2017.

[143] J. He and R. Lawrence, "A graph-based framework for multi-task multi-view learning," in *ICML*, 2011.

[144] J. Zhang and J. Huan, "Inductive multi-task learning with multiple view data," in *KDD*, 2012.

[145] X. Zhang, X. Zhang, and H. Liu, "Multi-task multi-view clustering for non-negative data," in *IJCAI*, 2015.

[146] X. Zhang, X. Zhang, H. Liu, and X. Liu, "Multi-task multi-view clustering," *IEEE TKDE*, vol. 28, no. 12, pp. 3324–3338, 2016.

[147] A. Niculescu-Mizil and R. Caruana, "Inductive transfer for Bayesian network structure learning," in *AISTATS*, 2007.

[148] J. Honorio and D. Samaras, "Multi-task learning of Gaussian graphical models," in *ICML*, 2010.

[149] D. Oyen and T. Lane, "Leveraging domain knowledge in multitask Bayesian network structure learning," in *AAAI*, 2012.

[150] K. Lin, J. Xu, I. M. Baytas, S. Ji, and J. Zhou, "Multi-task feature interaction learning," in *KDD*, 2016.

[151] Y. Zhang, "Parallel multi-task learning," in *ICDM*, 2015.

[152] O. Dekel, P. M. Long, and Y. Singer, "Online multitask learning," in *COLT*, 2006.

[153] O. Dekel, P. M. Long, and Y. Singer, "Online learning of multiple tasks with a shared loss," *JMLR*, vol. 8, pp. 2233–2264, 2007.

[154] G. Lugosi, O. Papaspiliopoulos, and G. Stoltz, "Online multi-task learning with hard constraints," in *COLT*, 2009.

[155] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile, "Linear algorithms for online multitask classification," *JMLR*, vol. 11, pp. 2901–2934, 2010.

[156] G. Pillonetto, F. Dinuzzo, and G. D. Nicolao, "Bayesian online multitask learning of Gaussian processes," *IEEE TPAMI*, vol. 32, no. 2, pp. 193–205, 2010.

[157] A. Saha, P. Rai, H. Daumé III, and S. Venkatasubramanian, "Online learning of multiple tasks and their relationships," in *AISTATS*, 2011.

[158] J. Wang, M. Kolar, and N. Srebro, "Distributed multi-task learning," in *AISTATS*, 2016.

[159] S. Liu, S. J. Pan, and Q. Ho, "Distributed multi-task relationship learning," in *KDD*, 2017.

[160] L. Xie, I. M. Baytas, K. Lin, and J. Zhou, "Privacy-preserving distributed multi-task learning with asynchronous updates," in *KDD*, 2017.

[161] C. Zhang, P. Zhao, S. Hao, Y. C. Soh, B. Lee, C. Miao, and S. C. H. Hoi, "Distributed multi-task classification: A decentralized online learning approach," *MLJ*, vol. 107, no. 4, pp. 727–747, 2018.

[162] K. Q. Weinberger, A. Dasgupta, J. Langford, A. J. Smola, and J. Attenberg, "Feature hashing for large scale multitask learning," in *ICML*, 2009.

[163] Q. An, C. Wang, I. Shterev, E. Wang, L. Carin, and D. B. Dunson, "Hierarchical kernel stick-breaking process for multi-task image analysis," in *ICML*, 2008.

[164] X. Wang, C. Zhang, and Z. Zhang, "Boosted multi-task learning for face verification with applications to web image and video search," in *CVPR*, 2009.

[165] Y. Zhang and D.-Y. Yeung, "Multi-task warped Gaussian process for personalized age estimation," in *CVPR*, 2010.

[166] X. Yuan and S. Yan, "Visual classification with multi-task joint sparse representation," in *CVPR*, 2010.

[167] H. Wang, F. Nie, H. Huang, S. L. Risacher, C. H. Q. Ding, A. J. Saykin, and L. Shen, "Sparse multi-task regression and feature selection to identify brain imaging predictors for memory performance," in *ICCV*, 2011.

[168] B. Cheng, G. Liu, J. Wang, Z. Huang, and S. Yan, "Multi-task low-rank affinity pursuit for image segmentation," in *ICCV*, 2011.

[169] C. Lang, G. Liu, J. Yu, and S. Yan, "Saliency detection by multitask sparsity pursuit," *IEEE TIP*, vol. 21, no. 3, pp. 1327–1338, 2012.

[170] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via multi-task sparse learning," in *CVPR*, 2012.

[171] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via structured multi-task sparse learning," *IJCV*, vol. 101, no. 2, pp. 367–383, 2013.

[172] Z. Hong, X. Mei, D. V. Prokhorov, and D. Tao, "Tracking via robust multi-task multi-view joint sparse representation," in *ICCV*, 2013.

[173] Y. Yan, E. Ricci, S. Ramanathan, O. Lanz, and N. Sebe, "No matter where you are: Flexible graph-guided multi-task learning for multi-view head pose classification under target motion," in *ICCV*, 2013.

[174] C. Yuan, W. Hu, G. Tian, S. Yang, and H. Wang, "Multi-task sparse learning with Beta process prior for action recognition," in *CVPR*, 2013.

[175] M. Lapin, B. Schiele, and M. Hein, "Scalable multitask representation learning for scene classification," in *CVPR*, 2014.

[176] A. H. Abdulnabi, G. Wang, J. Lu, and K. Jia, "Multi-task CNN model for attribute prediction," *IEEE TMM*, vol. 17, no. 11, pp. 1949–1959, 2015.

[177] C. Su, F. Yang, S. Zhang, Q. Tian, L. S. Davis, and W. Gao, "Multi-task learning with low rank attribute embedding for person re-identification," in *ICCV*, 2015.

[178] J. Yim, H. Jung, B. Yoo, C. Choi, D. Park, and J. Kim, "Rotating your face using multi-task deep neural network," in *CVPR*, 2015.

[179] X. Chu, W. Ouyang, W. Yang, and X. Wang, "Multi-task recurrent neural network for immediacy prediction," in *ICCV*, 2015.

[180] T. R. Almaev, B. Martínez, and M. F. Valstar, "Learning to transfer: Transferring latent task structures and its application to person-specific facial action unit detection," in *ICCV*, 2015.

[181] C. Widmer, J. Leiva, Y. Altun, and G. Rätsch, "Leveraging sequence classification by taxonomy-based multitask learning," in *RECOMB*, 2010.

[182] K. Zhang, J. W. Gray, and B. Parvin, "Sparse multitask regression for identifying common mechanism of response to therapeutic targets," *Bioinformatics*, vol. 26, no. 12, pp. 97–105, 2010.

[183] Q. Liu, Q. Xu, V. W. Zheng, H. Xue, Z. Cao, and Q. Yang, "Multi-task learning for cross-platform siRNA efficacy prediction: An in-silico study," *BMC Bioinformatics*, vol. 11, p. 181, 2010.

[184] K. Puniyani, S. Kim, and E. P. Xing, "Multi-population GWA mapping via multi-task regularized regression," *Bioinformatics*, vol. 26, no. 12, pp. 208–216, 2010.

[185] M. Alamgir, M. Grosse-Wentrup, and Y. Altun, "Multitask learning for brain-computer interfaces," in *AISTATS*, 2010.

[186] C. Widmer, N. C. Toussaint, Y. Altun, and G. Rätsch, "Inferring latent task structure for multitask learning by multiple kernel learning," *BMC Bioinformatics*, vol. 11, no. S-8, p. S5, 2010.

[187] Q. Xu, S. J. Pan, H. H. Xue, and Q. Yang, "Multitask learning for protein subcellular location prediction," *IEEE/ACM TCBB*, vol. 8, no. 3, pp. 748–759, 2011.

[188] J. Zhou, L. Yuan, J. Liu, and J. Ye, "A multi-task learning formulation for predicting disease progression," in *KDD*, 2011.

[189] F. Mordelet and J. Vert, "ProDiGe: Prioritization of disease genes with multitask machine learning from positive and unlabeled examples," *BMC Bioinformatics*, vol. 12, p. 389, 2011.

[190] J. Wan, Z. Zhang, J. Yan, T. Li, B. D. Rao, S. Fang, S. Kim, S. L. Risacher, A. J. Saykin, and L. Shen, "Sparse Bayesian multi-task learning for predicting cognitive outcomes from neuroimaging measures in Alzheimer's disease," in *CVPR*, 2012.

[191] H. Wang, F. Nie, H. Huang, J. Yan, S. Kim, S. L. Risacher, A. J. Saykin, and L. Shen, "High-order multi-task feature learning to identify longitudinal phenotypic markers for Alzheimer's disease progression prediction," in *NIPS*, 2012.

[192] J. Xu, J. Zhou, and P. Tan, "FORMULA: factorized multi-task learning for task discovery in personalized medical models," in *SDM*, 2015.

[193] Y. Li, J. Wang, J. Ye, and C. K. Reddy, "A multi-task learning formulation for survival analysis," in *KDD*, 2016.

[194] D. He, D. Kuhn, and L. Parida, "Novel applications of multitask learning and multiple output regression to multiple genetic trait prediction," *Bioinformatics*, vol. 32, no. 12, pp. 37–43, 2016.

[195] Z. Wu, C. Valentini-Botinhao, O. Watts, and S. King, "Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis," in *ICASSP*, 2015.

[196] Q. Hu, Z. Wu, K. Richmond, J. Yamagishi, Y. Stylianou, and R. Maia, "Fusion of multiple parameterisations for DNN-based sinusoidal speech synthesis with multi-task learning," in *InterSpeech*, 2015.

[197] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *ICML*, 2008.

[198] F. Wu and Y. Huang, "Collaborative multi-domain sentiment classification," in *ICDM*, 2015.

[199] M. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, "Multi-task sequence to sequence learning," in *ICLR*, 2016.

[200] L. Zhao, Q. Sun, J. Ye, F. Chen, C. Lu, and N. Ramakrishnan, "Multi-task learning for spatio-temporal event forecasting," in *KDD*, 2015.

[201] L. Zhao, Q. Sun, J. Ye, F. Chen, C. Lu, and N. Ramakrishnan, "Feature constrained multi-task learning models for spatiotemporal event forecasting," *IEEE TKDE*, vol. 29, no. 5, pp. 1059–1072, 2017.

[202] J. Bai, K. Zhou, G. Xue, H. Zha, G. Sun, B. L. Tseng, Z. Zheng, and Y. Chang, "Multi-task learning for learning to rank in web search," in *CIKM*, 2009.

[203] O. Chapelle, P. K. Shivaswamy, S. Vadrevu, K. Q. Weinberger, Y. Zhang, and B. L. Tseng, "Multi-task learning for boosting with application to web search ranking," in *KDD*, 2010.

[204] Y. Zhang, B. Cao, and D.-Y. Yeung, "Multi-domain collaborative filtering," in *UAI*, 2010.

[205] A. Ahmed, M. Aly, A. Das, A. J. Smola, and T. Anastasakos, "Web-scale multi-task feature selection for behavioral targeting," in *CIKM*, 2012.

[206] A. Ahmed, A. Das, and A. J. Smola, "Scalable hierarchical multitask learning algorithms for conversion optimization in display advertising," in *WSDM*, 2014.

[207] J. Ghosn and Y. Bengio, "Multi-task learning for stock selection," in *NIPS*, 1996.

[208] V. W. Zheng, S. J. Pan, Q. Yang, and J. J. Pan, "Transferring multi-device localization models using latent multi-task learning," in *AAAI*, 2008.

[209] K. M. A. Chai, C. K. I. Williams, S. Klanke, and S. Vijayakumar, "Multi-task Gaussian process learning of robot inverse dynamics," in *NIPS*, 2008.

[210] D.-Y. Yeung and Y. Zhang, "Learning inverse dynamics by Gaussian process regression under the multi-task learning framework," in *The Path to Autonomous Robots*, pp. 131–142, Springer, 2009.

[211] J. Zheng and L. M. Ni, "Time-dependent trajectory regression on road networks via multi-task learning," in *AAAI*, 2013.

[212] A. Huang, L. Xu, Y. Li, and E. Chen, "Robust dynamic trajectory regression on road networks: A multi-task learning framework," in *ICDM*, 2014.

[213] J. Xu, P. Tan, L. Luo, and J. Zhou, "GSpartan: a geospatio-temporal multi-task learning framework for multi-location prediction," in *SDM*, 2016.

[214] X. Lu, Y. Wang, X. Zhou, Z. Zhang, and Z. Ling, "Traffic sign recognition via multi-modal tree-structure embedded multi-task learning," *IEEE TITS*, vol. 18, no. 4, pp. 960–972, 2017.

[215] J. Xu, P. Tan, J. Zhou, and L. Luo, "Online multi-task learning framework for ensemble forecasting," *IEEE TKDE*, vol. 29, no. 6, pp. 1268–1280, 2017.

[216] J. Baxter, "A model of inductive bias learning," *JAIR*, vol. 12, pp. 149–198, 2000.

[217] S. Ben-David, J. Gehrke, and R. Schuller, "A theoretical framework for learning from a pool of disparate data sources," in *KDD*, 2002.

[218] S. Ben-David and R. Schuller, "Exploiting task relatedness for multiple task learning," in *COLT*, 2003.

[219] S. Ben-David and R. S. Borbely, "A notion of task relatedness yielding provable multiple-task learning guarantees," *MLJ*, vol. 73, no. 3, pp. 273–287, 2008.

[220] A. Pentina and S. Ben-David, "Multi-task and lifelong learning of kernels," in *ALT*, 2015.

[221] Y. Zhang, "Multi-task learning and algorithmic stability," in *AAAI*, 2015.

[222] A. Maurer, "Bounds for linear multi-task learning," *JMLR*, vol. 7, pp. 117–139, 2006.

[223] A. Maurer, "The Rademacher complexity of linear transformation classes," in *COLT*, 2006.

[224] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari, "Regularization techniques for learning with matrices," *JMLR*, vol. 13, pp. 1865–1890, 2012.

[225] M. Pontil and A. Maurer, "Excess risk bounds for multitask learning with trace norm regularization," in *COLT*, 2013.

[226] A. Maurer, M. Pontil, and B. Romera-Paredes, "The benefit of multitask representation learning," *JMLR*, vol. 17, pp. 1–32, 2016.

[227] B. Juba, "Estimating relatedness via data compression," in *ICML*, 2006.

[228] A. Argyriou, C. A. Micchelli, and M. Pontil, "When is there a representer theorem? vector versus matrix regularizers," *JMLR*, vol. 10, pp. 2507–2529, 2009.

[229] A. Argyriou, C. A. Micchelli, and M. Pontil, "On spectral learning," *JMLR*, vol. 11, pp. 935–953, 2010.

[230] K. Lounici, M. Pontil, A. B. Tsybakov, and S. A. van de Geer, "Taking advantage of sparsity in multi-task learning," in *COLT*, 2009.

[231] G. Obozinski, M. J. Wainwright, and M. I. Jordan, "Support union recovery in high-dimensional multivariate regression," *Annals of Statistics*, vol. 39, no. 1, pp. 1–47, 2011.

[232] M. Kolar, J. D. Lafferty, and L. A. Wasserman, "Union support recovery in multi-task learning," *JMLR*, vol. 12, pp. 2415–2435, 2011.