A Survey on Multi-Task Learning

Yu Zhang, *Member, IEEE* and Qiang Yang, *Fellow, IEEE*

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 from the perspective of algorithmic modeling, applications, and theoretical analyses. For algorithmic modeling, we give a definition of MTL and then classify different MTL algorithms into five categories, including feature learning approach, low-rank approach, task clustering approach, task relation learning approach, and decomposition approach as well as discussing 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, we review online, parallel, and distributed MTL models as well as dimensionality reduction and feature hashing 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

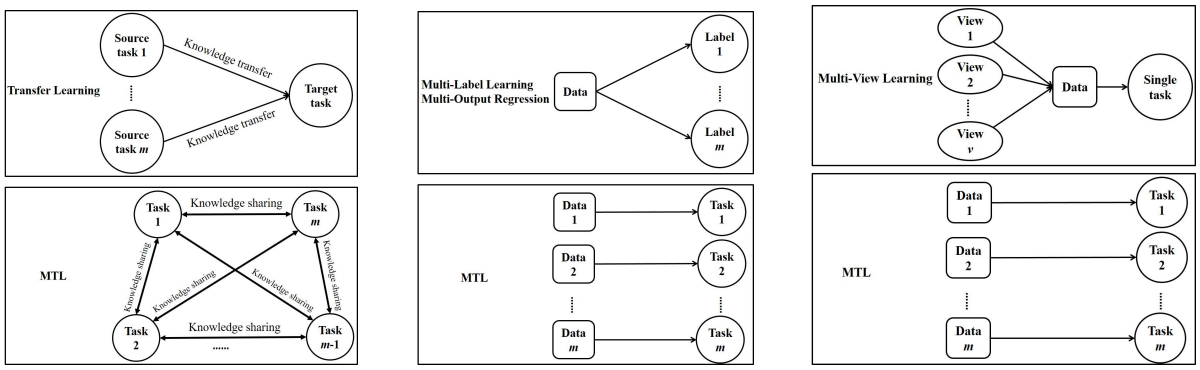
HUMAN can learn multiple tasks simultaneously and during this learning process, human can use the knowledge learned in a task to help the learning of another task. For example, according to our experience in learning to play tennis and squash together, we find that the skill of play tennis can help learn to play squash and vice versa. Inspired by such human learning ability, Multi-Task Learning (MTL) [1], a learning paradigm in machine learning, aims to jointly learn multiple related tasks so that the knowledge contained in a task can be leveraged by other tasks with the hope of improving the generalization performance of all the tasks at hand.

At its early stage, an important motivation of MTL is to alleviate the data sparsity problem where each task has a limited number of labeled data. In the data sparsity problem, the number of labeled data in each task is insufficient to train an accurate learner and instead MTL aggregates the labeled data in all the tasks in the spirit of data augmentation to obtain a more accurate learner for each task. From this perspective, MTL can help reuse existing knowledge and reduce the cost of manual labeling for learning tasks. When the era of "big data" comes in some areas such as computer vision and Natural Language Processing (NLP), it is found that deep MTL models can achieve better performance than their single-task counterparts. One reason that MTL is effective is that it utilizes more data from different learning tasks when compared with single-task learning. With more data, MTL can learn more robust and universal representations for multiple tasks and more powerful models, leading to better knowledge sharing among tasks, better performance of each task, and a low risk of overfitting in each task.

MTL is related to other learning paradigms in machine learning, including transfer learning [2], multi-label learning [3], and multi-output regression. The setting of MTL is similar to that of transfer learning but 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, transfer learning is to improve the performance of a target task with the help of source tasks, hence the target task plays a more important role than source tasks. In a word, MTL treats all the tasks equally but in transfer learning the target task attracts most attentions. From the perspective of the knowledge flow, flows of knowledge transfer in transfer learning are from source task(s) to the target task but in multi-task learning, there are flows of knowledge sharing between any pair of tasks, which is illustrated in Fig. 1(a). Continual learning [4], in which tasks come sequentially, learns tasks one by one, while MTL is to learn multiple tasks together. In multi-label learning and multi-output regression, each data point is associated with multiple labels which can be categorical or numeric. If we treat each of all the possible labels as a task, multi-label learning and multi-output regression can be viewed in some sense as a special case of multi-task learning where different tasks always share the same data during both the training and testing phrases. Such characteristic in multi-label learning and multi-output regression leads to different research issues from MTL. For example, the ranking loss, which enforces the scores (e.g., the classification probability) of labels associated with a data point to be larger than those of absent labels, can be used for multi-label learning but it does not fit MTL where different tasks possess different data. On the other hand, this characteristic in multi-label learning and multi-output regression is invalid in MTL problems. For example, in a MTL problem discussed in Section 2.7 where each task is to predict the disease symptom score of Parkinson for a patient based on 19 bio-medical features, different patients/tasks should not share the bio-medical data. In a word, multi-label learning and multi-output regression are different from multi-task learning as illustrated in Fig. 1(b) and hence we will not survey literature on multi-label learning and multi-output regression. Moreover, multi-view learning is another learning paradigm in machine learning, where each data point is associated with multiple views each of which consists of a set of features. Even though different views have different sets of

- *Y. Zhang is with Department of Computer Science and Engineering, Southern University of Science and Technology and Q. Yang is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology.*
  E-mail: yu.zhang.ust@gmail.com, qyang@cse.ust.hk

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2021.3070203, IEEE Transactions on Knowledge and Data Engineering



(a) MTL vs. transfer learning

(b) MTL vs. multi-label learning/multi-output regression

(c) MTL vs. multi-view learning

Fig. 1. Illustrations for differences between MTL and other learning paradigms.

features, all the views are used together to learn for the same task and hence multi-view learning belongs to single-task learning with multiple sets of features, which is different from MTL as shown in Fig. 1(c).

Over past decades, MTL has attracted many attentions in the artificial intelligence and machine learning communities. Many MTL models have been devised and many MTL applications in other areas have been exploited. Moreover, many analyses have been conducted to study theoretical problems in MTL. This paper serves as a survey on MTL from the perspective of algorithmic modeling, applications, and theoretical analyses. For algorithmic modeling, we first give a definition for MTL and then classify different MTL algorithms into five 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. After that, we discuss the combination of MTL with other learning paradigms, including semi-supervised learning, active learning, unsupervised learning, reinforcement learning, multi-view learning, and graphical models. To handle a large number of tasks, we review online, parallel, and distributed MTL models. For data in a high-dimensional space, feature selection, dimensionality reduction, and feature hashing are introduced as vital tools to process them. As a promising learning paradigm, MTL has many applications in various areas and here we briefly review its applications in computer vision, bioinformatics, health informatics, speech, NLP, web, and so on. From the perspective of theoretical analyses on MTL, we review relevant works. At last, we discuss several future directions for MTL.

2 MTL Models

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

Definition (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 learn the m tasks together to improve the learning of a model for each task $\mathcal{T}_i$ by using the knowledge contained in all or some of other tasks.*

Based on the definition of MTL, we focus on supervised learning tasks in this section since most MTL studies fall in this setting and for other types of tasks, we review them in the next section. In the setting of supervised learning tasks, a task $\mathcal{T}_i$ is usually 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 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 [5], 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 works to study it. A simple solution is to formulate 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. An advanced solution we think is to use multi-task models which can degenerate to their single-task counterparts given some form of model parameters, for example, problem (33) presented in Section 2.8 which can reduce to multiple single-task models with the learning of different tasks decoupled when a parameter $\mathbf{\Sigma}$ 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 or weights in deep 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 [6], 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 is to identify task clusters each of which contains similar tasks. 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 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 2 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 [7], 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. [8] 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.

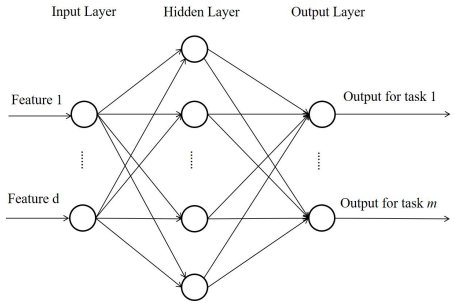


Fig. 2. An example for the multi-task feedforward neural network with an input layer, a hidden layer, and an output layer.

Different from multi-layer feedforward neural networks which are connectionist models, the multi-task feature learning (MTFL) method [9] is formulated under the regularization framework with the objective function as

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

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, $\mathbf{A} \in \mathbb{R}^{d\times m}$ contains model parameters of all the tasks with its ith column $\mathbf{a}^i$ as 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. As proved in [9], problem (1) is equivalent to

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

where $L(\mathbf{W},\mathbf{b}) = \sum_{i=1}^{m} \frac{1}{n_i} \sum_{j=1}^{n_i} l(y_j^i, (\mathbf{w}^i)^T\mathbf{x}_j^i + b_i)$ denotes the total training loss, $\mathrm{tr}(\cdot)$ 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 a 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 $\mathbf{D} = (\mathbf{W}^T\mathbf{W})^{\frac{1}{2}} / \mathrm{tr}\left((\mathbf{W}^T\mathbf{W})^{\frac{1}{2}}\right)$ 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. [10] extend problem (2) to a general formulation where the second term in the objective function becomes $\lambda \mathrm{tr}(\mathbf{W}^T f(\mathbf{D})\mathbf{W})$ 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 [11] 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 $\mathbf{a}^i$, the ith column of $\mathbf{A}$, contains model parameters of the ith task, $\mathbf{u}^j$ is the jth column in $\mathbf{U}$, $[c]$ for an integer c denotes a set of integers from 1 to c, $\|\cdot\|_1$ 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. [12] propose a multi-task infinite support vector machine via the Indian buffet process and the difference is that in [12] the dictionary is sparse and model parameters are non-sparse. In [13], 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, which facilitates the learning of invariant features for multiple tasks, and hence many deep multi-task models belonging to this approach have been proposed with each task modeled by a deep neural network. Here we classify deep multi-task models in this approach into three main categories. The first category [14], [15], [16], [17], [18] is to learn a common feature representation for multiple tasks by sharing first several layers in a similar architecture to Fig. 2. However, different from Fig. 2, deep MTL models in this category have a large number of shared layers, which have general structures such as convolutional layers and pooling layers. Building on the first category, the second category is to use adversarial learning, which is inspired by generative adversarial networks, to learn a common feature representation for MTL as did in [19], [20]. Specifically, there are three networks in such adversarial multi-task models, including a feature network N_f, a classification network N_c, and a domain network N_d. Based on N_f, N_c is to minimize the training loss for all the tasks, while N_d aims to distinguish which task a data instance is from. The objective function of such models is usually formulated as

$$\min_{\theta_f,\theta_c} \max_{\theta_d} \sum_{i=1}^{m} \frac{1}{n_i} \sum_{j=1}^{n_i} \left(l(y_j^i, N_c(N_f(\mathbf{x}_j^i))) - l_{ce}(d_j^i, N_d(N_f(\mathbf{x}_j^i))) \right),$$

where θ_f, θ_c, θ_d denotes parameters of three networks N_f, N_c, N_d, respectively, $d_j^i \in \{1, \ldots, m\}$ denotes the task index/label of $\mathbf{x}_j^i$, and $l_{ce}(\cdot,\cdot)$ denotes the cross-entropy loss. Based on this min-imax problem, N_f is to minimize the training loss for all the tasks and maximize the cross-entropy loss to fool the domain network to make the learned feature representation indistinguishable to all the tasks. When there is no domain network, this category can reduce to the first category. Moreover, in [20], each task can learn its specific feature representation to increase the expressive power of the whole model. The last category is to learn different but related feature representations for different tasks with the cross-stitch network [21] as a representative model. 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. 3. Here matrix $\boldsymbol{\alpha}$, which is defined as $\boldsymbol{\alpha} \equiv \begin{pmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{pmatrix}$, encodes feature-level task relations between the two tasks and it can be learned via the backpropagation method.

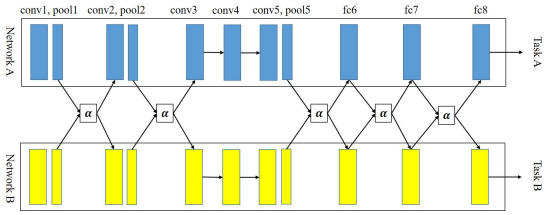


Fig. 3. The architecture for the cross-stitch network.

2.1.2 Feature Selection Approach

One way to do feature selection in MTL is to use the $\ell_{p,q}$ norm denoted by $\|\mathbf{W}\|_{p,q} \equiv \|(\|\mathbf{w}_1\|_p, \ldots, \|\mathbf{w}_d\|_p)\|_q$, where $\mathbf{w}_i$ denotes the ith row of $\mathbf{W}$ and $\|\cdot\|_p$ denotes the ℓ_p norm of a vector, to achieve the group sparsity. Obozinski et al. [22] 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

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

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 [22], a path-following algorithm is proposed to solve problem (4) and then Liu et al. [23] 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. [24] 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 [25] 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 [26]. Moreover, in order to make speedup, a safe screening method is proposed in [27] to filter out useless features corresponding to zero rows in $\mathbf{W}$ before optimizing problem (4). Liu et al. [28] 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.

In order to attain a more sparse subset of features, Gong et al. [29] propose a capped-$\ell_{p,1}$ regularizer for multi-task feature selection where $p = 1$ or 2 and the objective function is formulated as

$$\min_{\mathbf{W},\mathbf{b}} \; L(\mathbf{W},\mathbf{b}) + \lambda\sum_{i=1}^{d}\min(\|\mathbf{w}_i\|_p,\theta), \tag{6}$$

where $\mathbf{w}_i$ denotes the ith row of $\mathbf{W}$. With a given threshold θ, the capped-$\ell_{p,1}$ regularizer (i.e., the second term in problem (6)) focuses on rows with smaller ℓ_p norms than θ, which is more likely to be sparse. When θ becomes large enough, the capped-$\ell_{p,1}$ regularizer becomes $\|\mathbf{W}\|_{p,1}$ and hence problem (6) degenerates to problem (4) or (5) when p equals 2 or ∞.

Lozano and Swirszcz [30] 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

$$\min_{\boldsymbol{\theta},\hat{\mathbf{W}},\mathbf{b}} \; L(\mathbf{W},\mathbf{b}) + \lambda_1\|\boldsymbol{\theta}\|_1 + \lambda_2\|\hat{\mathbf{W}}\|_1 \text{ s.t. } w_{ji} = \theta_j\hat{w}_{ji}, \theta_j \geq 0, \tag{7}$$

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 [30] that problem (7) 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. [31] extend problem (7) 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. [32] 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 [30], [31]. Similar to [30], Jebara [33] proposes to learn a binary indicator vector to do multi-task feature selection based on the maximum entropy discrimination formalism.

Similar to [32] where a priori information is given to describe task relations in a hierarchical/tree structure, Kim and Xing [34] 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 the 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. [35] 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. [36] propose a probabilistic interpretation where the $\ell_{p,1}$ regularizer corresponds to a generalized normal prior: $w_{ji} \sim \mathcal{GN}(\cdot|0,\rho_j,p)$, where $\cdot$ denotes a (random) variable when we do not want to introduce it explicitly. Based on this interpretation, Zhang et al. [36] 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 [37], 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 $\mathcal{N}(\cdot|\mathbf{m},\boldsymbol{\sigma})$ 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.

Hernández-Lobato et al. [38] propose a probabilistic model based on the horseshoe prior as

$$\begin{aligned}\mathbb{P}(w_{ji}) = &\left[\pi(w_{ji})^{\eta_{ji}}\delta_0^{1-\eta_{ji}}\right]^{z_j}\left[\pi(w_{ji})^{\tau_{ji}}\delta_0^{1-\tau_{ji}}\right]^{\omega_i(1-z_j)}\\ &\left[\pi(w_{ji})^{\gamma_j}\delta_0^{1-\gamma_j}\right]^{(1-\omega_i)(1-z_j)},\end{aligned} \tag{8}$$

where δ_0 is the probability mass function at zero and $\pi(\cdot)$ denotes the density function of non-zero coefficients. In Eq. (8), 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 a 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. (8) 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 different way from [36].

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. By selecting a subset of the original features as the new representation, the feature selection approach has a better interpretability.

2.2 Low-Rank Approach

The relatedness among multiple tasks can imply the low-rank of $\mathbf{W}$, leading to the low-rank approach. For example, if the ith, jth, and kth tasks are related in that the model parameter $\mathbf{w}^i$ of the ith task is a linear combination of those of the other two tasks, then it is easy to show that the rank of $\mathbf{W}$ is at most $m-1$ and hence of low rank. From this perspective, the more the relatedness is, the lower the rank of $\mathbf{W}$ is.

Ando and Zhang [39] assume that model parameters of different tasks share a low-rank subspace in part and specifically, $\mathbf{w}^i$ takes the following form as

$$\mathbf{w}^i = \mathbf{u}^i + \mathbf{\Theta}^T \mathbf{v}^i. \tag{9}$$

Here $\mathbf{\Theta} \in \mathbb{R}^{h \times d}$ is the shared low-rank subspace by multiple tasks where $h < d$. Then we can write in a matrix form as $\mathbf{W} = \mathbf{U} + \mathbf{\Theta}^T \mathbf{V}$. Based on the form of $\mathbf{W}$, the objective function proposed in [39] is formulated as

$$\min_{\mathbf{U},\mathbf{V},\mathbf{\Theta},\mathbf{b}} L(\mathbf{U} + \mathbf{\Theta}^T \mathbf{V}, \mathbf{b}) + \lambda \|\mathbf{U}\|_F^2 \ \text{s.t.}\ \mathbf{\Theta}\mathbf{\Theta}^T = \mathbf{I}, \tag{10}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. The orthonormal constraint on $\mathbf{\Theta}$ in problem (10) makes the subspace non-redundant. When λ is large enough, the optimal $\mathbf{U}$ can become a zero matrix and hence problem (10) is very similar to problem (1) except that there is no regularization on $\mathbf{V}$ in problem (10) and that $\mathbf{\Theta}$ has a smaller number of rows than columns. Chen et al. [40] generalize problem (10) as

$$\begin{aligned} \min_{\mathbf{U},\mathbf{V},\mathbf{\Theta},\mathbf{b}} \quad & L(\mathbf{W}, \mathbf{b}) + \lambda_1 \|\mathbf{U}\|_F^2 + \lambda_2 \|\mathbf{W}\|_F^2 \\ \text{s.t.} \quad & \mathbf{W} = \mathbf{U} + \mathbf{\Theta}^T \mathbf{V},\ \mathbf{\Theta}\mathbf{\Theta}^T = \mathbf{I}. \end{aligned} \tag{11}$$

When setting λ_2 to be 0, problem (11) reduces to problem (10). Even though problem (11) is non-convex, with some convex relaxation technique, it can be relaxed to the following convex problem as

$$\min_{\mathbf{W},\mathbf{b},\mathbf{M}} L(\mathbf{W}, \mathbf{b}) + \lambda \text{tr}\big(\mathbf{W}^T (\mathbf{M} + \eta \mathbf{I})^{-1} \mathbf{W}\big) \ \text{s.t.}\ \begin{matrix} \text{tr}(\mathbf{M}) = h \\ \mathbf{0} \preceq \mathbf{M} \preceq \mathbf{I} \end{matrix}, \tag{12}$$

where $\eta = \lambda_2/\lambda_1$ and $\lambda = \lambda_1 \eta(\eta + 1)$. One advantage of problem (12) over problem (11) is that the global optimum of the convex problem (12) is much easier to be obtained than that of the non-convex problem (11). Compare with the alternative objective function (2) in the MTFL method, problem (12) has a similar formulation where $\mathbf{M}$ models the feature covariance for all the tasks. Problem (10) is extended in [41] to a general case where different $\mathbf{w}^i$'s lie in a manifold instead of a subspace. Moreover, in [42], a latent variable model is proposed for $\mathbf{W}$ with the same decomposition as Eq. (9) and it can provide a framework for MTL by modeling more cases than problem (10) 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 [43] as

$$\min_{\mathbf{W},\mathbf{b}} \ L(\mathbf{W}, \mathbf{b}) + \lambda \|\mathbf{W}\|_{S(1)}, \tag{13}$$

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 [44] propose a capped trace regularizer with the objective function formulated as

$$\min_{\mathbf{W},\mathbf{b}} \ L(\mathbf{W}, \mathbf{b}) + \lambda \sum_{i=1}^{\min(m,d)} \min(\mu_i(\mathbf{W}), \theta). \tag{14}$$

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 problem (14) will reduce to problem (13). Moreover, a spectral k-support norm is proposed in [45] as an improvement over the trace norm regularization.

The trace norm regularization has been extended to regularize model parameters in deep multi-task models. Specifically, the weights in the last several fully connected layers of deep multi-task neural networks can be viewed as the parameters of learners for 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, which are based on the trace norm, are used in [46] 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 [47] 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 a task transfer matrix $\mathbf{A}$ is defined with its (i,j)th entry a_{ij} recording the generalization accuracy obtained for task $\mathcal{T}_i$ by using task $\mathcal{T}_j$'s distance metric. 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 learning process [48] in a similar way to k-means clustering.

Bakker and Heskes [49] propose a multi-task Bayesian neural network model with the network structure similar to Fig. 2 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: $\mathbf{w}^i \sim \sum_{j=1}^r \pi_j \mathcal{N}(\cdot|\mathbf{m}_j, \mathbf{\Sigma}_j)$, 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 [50], [51], [52] that are based on Gaussian processes and t processes.

Xue et al. [53] 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 affects the number of task clusters. This model is extended in [54], [55] 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 [56] where each task is a compressive sensing task. Moreover, a nested Dirichlet process is proposed in [57], [58] 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 [59], $\mathbf{w}^i$ is decomposed as $\mathbf{w}^i = \mathbf{u}^i + \mathbf{\Theta}_i^T \mathbf{v}^i$ similar to Eq. (9), where $\mathbf{u}^i$ and $\mathbf{\Theta}_i$ are sampled according to a Dirichlet process.

Different from [49], [53], Jacob et al. [60] 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, a convex objective function is formulated as

$$\begin{aligned} &\min_{\mathbf{W},\mathbf{b},\mathbf{\Sigma}} L(\mathbf{W}, \mathbf{b}) + \lambda \mathrm{tr}(\mathbf{W}\mathbf{1}\mathbf{1}^T\mathbf{W}^T) + \mathrm{tr}(\tilde{\mathbf{W}}\mathbf{\Sigma}^{-1}\tilde{\mathbf{W}}^T) \\ &\text{s.t. } \tilde{\mathbf{W}} = \mathbf{W}\mathbf{\Pi},\ \alpha\mathbf{I} \preceq \mathbf{\Sigma} \preceq \beta\mathbf{I},\ \mathrm{tr}(\mathbf{\Sigma}) = \gamma, \end{aligned} \tag{15}$$

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. [61] extend the MTFL method [9] 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

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

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

Han and Zhang [62] devise a structurally sparse regularizer to cluster tasks with the objective function as

$$\min_{\mathbf{W},\mathbf{b}} L(\mathbf{W}, \mathbf{b}) + \lambda \sum_{j>i} \|\mathbf{w}^i - \mathbf{w}^j\|_2. \tag{16}$$

Problem (16) is a special case of the method proposed in [62] 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 (16), the cluster structure can be discovered by comparing columns in $\mathbf{W}$. One advantage of this structurally sparse regularizer is that the convex problem (16) can automatically determine the number of task clusters.

Barzilai and Crammer [63] propose a task clustering method by defining $\mathbf{W}$ as $\mathbf{W} = \mathbf{F}\mathbf{G}$ 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

$$\min_{\mathbf{F},\mathbf{G},\mathbf{b}} L(\mathbf{F}\mathbf{G}, \mathbf{b}) + \lambda \|\mathbf{F}\|_F^2 \text{ s.t. } \begin{matrix} \mathbf{G} \in \{0,1\}^{r\times m} \\ \|\mathbf{g}^i\|_2 = 1\ \forall i \in [m] \end{matrix}, \tag{17}$$

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 [64] 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 a matrix $\mathbf{Z}$ whose (i,j)th entry is z_{ij}, the objective function is formulated as

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

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 (16) since the regularizer in problem (16) 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 [65], 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 problem (17) as $\mathbf{W} = \mathbf{L}\mathbf{S}$ 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 $\mathbf{L}$ or equivalently belongs to a small number of clusters. The objective function is formulated as

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

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, while here the number of columns in $\mathbf{S}$ is smaller than that of its rows. This method has been extended in [66] to decompose the parameter tensor in the fully connected layers of deep neural networks.

Among the aforementioned methods, the method in [47] 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 [53], [62] can automatically determine the number of task

clusters, where the method in [53] depends on the capacity of the Dirichlet process while the method in [62] relies on the use of a structurally sparse regularizer. Among all those models, some belong to Bayesian learning, i.e., [49], [53], while the rest models are regularized models. Among those regularized methods, only the objective function proposed in [62] 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 ($r < m$) 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. (9), we can see that the decomposition of $\mathbf{W}$ in [39] becomes similar to those in [63], [65], which in some sense shows the relation between those two approaches. Moreover, the equivalence between problems (12) and (15), two typical methods in the low-rank and task clustering approaches, has been proved in [67]. The task clustering approach can visualize the learned 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, task relations are assumed to be known as a priori information. In [68], [69], 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 [70], [71], 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. A similar formulation to [70] is proposed in [72] to estimate the mean of multiple distributions by learning pairwise task relations and another similar formulation is proposed in [73] for log-density gradient estimation. Given a tree structure describing relations among tasks in [74], model parameters of a task corresponding to a node in the tree are 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. [75] propose a multi-task Gaussian process (MTGP) by defining a prior on f_j^i, the functional value for $\mathbf{x}_j^i$, as $\mathbf{f} \sim \mathcal{N}(\cdot|\mathbf{0}, \mathbf{\Sigma})$, 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 [76], the learning curve and generalization bound of the MTGP 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 [77] propose a multi-task generalized t process by placing an inverse-Wishart prior on $\mathbf{\Omega}$ as $\mathbf{\Omega} \sim \mathcal{IW}(\cdot|\nu, \mathbf{\Psi})$, 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).

Different from [75], [77] which are Bayesian models, Zhang and Yeung [78], [79] propose a regularized multi-task model called multi-task relationship learning (MTRL) by placing a matrix-variate normal prior on $\mathbf{W}$ as

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

where $\mathcal{MN}(\cdot|\mathbf{M}, \mathbf{A}, \mathbf{B})$ 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 a modified maximum a posterior solution is formulated as

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

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 [78], [79] 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 [80] and multi-label learning [81] by learning label correlations. Problem (21) can also been interpreted from the perspective of reproducing kernel Hilbert spaces for vector-valued functions [82], [83], [84], [85]. Moreover, Problem (21) is extended to learn sparse task relations in [86] via the ℓ_1 regularization on $\mathbf{\Omega}$ when the number of tasks is large. A model similar to problem (21) is proposed in [87] via a matrix-variate normal prior on $\mathbf{W}$: $\mathbf{W} \sim \mathcal{MN}(\cdot|\mathbf{0}, \mathbf{\Omega}_1, \mathbf{\Omega}_2)$, where $\mathbf{\Omega}_1^{-1}$ and $\mathbf{\Omega}_2^{-1}$ are assumed to be sparse. The MTRL model is extended in [88] to use the symmetric matrix-variate generalized hyperbolic distribution to learn block sparse structure in $\mathbf{W}$ and in [89] 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 [36] 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 a Wishart distribution as $\mathcal{W}(\cdot|\mathbf{0}, \mathbf{\Omega})$, Zhang and Yeung [90] generalize it as

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

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 regularized method is devised to learn high-order task relations in [90]. The MTRL model has been extended to multi-output regression [89], [91], [92], [93] by modeling the structure contained in noises via some matrix-variate priors. For deep neural networks, the MTRL method has been extended in [94] by placing a tensor-variate normal 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 [95] 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

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

The first regularizer in problem (24) enforces $\mathbf{\Sigma}$ to be nearly symmetric and the second one is to penalize the complexity of $\mathbf{\Sigma}$. The constraints in problem (24) guarantee that the similarity from one task to itself is positive and also the largest. Similarly, a multi-task kernel regression is proposed in [95] for regression tasks.

While the aforementioned methods whose task relations are symmetric except [95], Lee et al. [96] 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

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

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 sharing 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 $\|\mathbf{W} - \mathbf{W}\mathbf{A}\|_F^2 = \text{tr}\left(\mathbf{W}(\mathbf{I} - \mathbf{A})(\mathbf{I} - \mathbf{A})^T \mathbf{W}^T\right)$. Based on this reformulation, the regularizer in problem (25) is a special case of that in problem (21) by assuming $\mathbf{\Omega}^{-1} = (\mathbf{I} - \mathbf{A})(\mathbf{I} - \mathbf{A})^T$. Though $\mathbf{A}$ is asymmetric, from the perspective of the regularizer, the task relations here are symmetric and act as the task precision matrix with 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 $h \geq 2$, i.e., $\mathbf{W} = \sum_{k=1}^h \mathbf{W}_k$. The objective functions of most methods in this approach can be unified as

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

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 as follows.

In [97] 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 that 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. [98] 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 (13), $\mathcal{C}_W$ 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 the previous model. When λ_2 is large enough, $\mathbf{W}_2$ will become a zero matrix and then problem (27) will act similarly to problem (13).

$g_i(\cdot)$'s in [99] where $\mathcal{C}_W = \emptyset$ 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 in $\mathbf{W}$ for them have no low-rank structure even though those in $\mathbf{W}_1$ may have.

In [100], 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 [99] 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 [101] 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 a clear cluster structure.

Different from the aforementioned methods which have only two component matrices, an arbitrary number of component matrices are considered in [102] 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 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 among tasks. For example, Han and Zhang [103] define

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

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 $\mathbf{w}_k^j$ have 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 [62], which is a generalization of problem (16).

Another way to relate different component matrices is to use a non-decomposable regularizer as [104] 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 $h \equiv 2^m$ 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

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

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 [104], $r(\mathbf{W}_a)$ adopts the regularizer proposed in [68] which enforces the parameters of all the tasks to approach their average.

Different from deep MTL models which are deep in terms of layers of feature representations, 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 those 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 learning approach can learn common features, which are generic and invariant 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 may cause the performance deterioration. 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 additional computational costs. The task relation learning approach can learn model parameters and pairwise task relations simultaneously. The learned task relations can give us insights about the relations between tasks and hence 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, e.g., 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 [49]: 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[1]: 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 [10]: 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 [104]: 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[2]: 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 [60]: 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 [53]: 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.

1. http://www.gaussianprocess.org/gpml/data/
2. http://www.cs.jhu.edu/~mdredze/datasets/sentiment/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2021.3070203, IEEE Transactions on Knowledge and Data Engineering

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', RMSE is for 'root mean squared error', and AUC stands for 'Area Under Curve'. ↑ after the evaluation metric implies that the larger value the better performance and ↓ indicates the opposite case.

Dataset (Reference)	Evaluation Metric	STL	Feature Learning	Low-Rank	Task Clustering	Task Relation Learning	Decomposition
			[9]	[43]	[60]/ [61]/ [65]/ [66]	[78]/ [85]/ [94]	[97]/ [98]/ [99]/ [104]/ [103]
School ([103])	nMSE↓	—	0.4393	—	0.4374/-/0.6466/-	—	0.4445/-/-/-/0.4169
SARCOS ([99])	nMSE↓	0.1821	0.1568	0.1531	—	—	0.1495/0.1456/-/-/-
Computer Survey ([101])	RMSE↓	2.381	—	—	2.072/-/-/-	2.110/-/-	2.138/2.052/2.074/-/-
Parkinson ([85])	Explained Variance↑	2.8%	—	—	2.7%/33.6%/-/-	12.0%/27.0%/-	-/-/-/16.8%/-
Sentiment ([78])	Classification Error↓	0.2779	0.2756	—	—	0.2324/-/-	—
MHC-I ([64])	Classification Error↓	0.2010	—	—	0.1890/0.2050/-/-	0.1870/-/-	0.2030/-/0.2070/-/-
Landmine ([85])	AUC↑	74.6%	—	—	75.9%/76.7%/-/-	76.1%/76.8%/-	-/-/-/76.4%/-
Office-Caltech ([94])	Classification Error↓	0.0920	0.0740	—	-/-/-/0.0670	0.0690/-/0.0450	-/-/0.0760/-/-
Office-Home ([94])	Classification Error↓	0.3430	0.4170	—	-/-/-/0.3350	0.4070/-/0.3310	-/-/0.4140/-/-
ImageCLEF ([94])	Classification Error↓	0.3640	0.3440	—	-/-/-/0.2780	0.3350/-/0.2470	-/-/0.3510/-/-

- Office-Caltech dataset [105]: 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[3]: 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[4]: 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 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. We also include the performance of Single-Task Learning (STL), which trains a learning model for each task separately, for comparison. It is easy to see that MTL models perform better than STL counterparts in most cases, which verifies the effectiveness of MTL. Usually, different datasets have their own characteristics, making them more suitable for some MTL approach. For example, according to the studies in [49], [70], [101], different tasks in the School dataset are found to be very similar to each other. According to [53], 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 [60], 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 those 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 more insights 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 [78], and the task similarities in the Office-Caltech dataset are shown in Figure 3(b) of [94]. Moreover, for image datasets (i.e., Office-Caltech, Office-Home and ImageCLEF), deep MTL models (e.g., [66], [94]) 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.

3. http://hemanthdv.org/OfficeHome-Dataset
4. http://imageclef.org/2014/adaptation

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

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

where $f(\cdot)$ denotes a regularizer or constraint on $\mathbf{\Theta}$. From the perspective of probabilistic modeling, the regularizer $\frac{\lambda}{2}\mathrm{tr}(\mathbf{W}^T\mathbf{\Theta}^{-1}\mathbf{W})$ corresponds to a matrix-variate normal distribution on $\mathbf{W}$ as $\mathbf{W} \sim \mathcal{MN}(\mathbf{0}, \frac{1}{\lambda}\mathbf{\Theta} \otimes \mathbf{I})$. Based on this probabilistic prior, $\mathbf{\Theta}$ models the covariance between the features since $\frac{1}{\lambda}\mathbf{\Theta}$ is the row covariance matrix with each row in $\mathbf{W}$ corresponding 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 $\mathbf{\Theta}$. For example, methods in [9], [39], [40] use $f(\cdot)$ to restrict the trace of $\mathbf{\Theta}$ as shown in problems (2) and (12). Moreover, multi-task feature selection methods based on the $\ell_{2,1}$ norm such as [22], [23], [24] can be reformulated as instances of problem (32).

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

$$\min_{\mathbf{W},\mathbf{b},\mathbf{\Sigma}} L(\mathbf{W},\mathbf{b}) + \frac{\lambda}{2}\mathrm{tr}(\mathbf{W}\mathbf{\Sigma}^{-1}\mathbf{W}^T) + g(\mathbf{\Sigma}), \tag{33}$$

where $g(\cdot)$ denotes a regularizer or constraint on $\mathbf{\Sigma}$. The regularizer $\frac{\lambda}{2}\mathrm{tr}(\mathbf{W}\mathbf{\Sigma}^{-1}\mathbf{W}^T)$ corresponds to a matrix-variate normal prior on $\mathbf{W}$ as $\mathbf{W} \sim \mathcal{MN}(\mathbf{0}, \mathbf{I} \otimes \frac{1}{\lambda}\mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is to model the task relations since $\frac{1}{\lambda}\mathbf{\Sigma}$ is the column covariance with each column in $\mathbf{W}$ corresponding to a task. From this perspective, the two regularizers for $\mathbf{W}$ in problems (32) and (33) have different meanings even though the formulations seem a bit similar. All the methods in this category use different functions $g(\cdot)$ to learn $\mathbf{\Sigma}$ with different

functionalities. For example, the methods in [68], [69], [70], [71], which utilize a priori information on task relations, directly learn $\mathbf{W}$ and $\mathbf{b}$ by defining $g(\mathbf{\Sigma}) = 0$. Some task clustering methods [60], [64] identify task clusters by assuming that $\mathbf{\Sigma}$ has a block structure. Several task relation learning methods including [78], [79], [86], [96], [106] directly learn $\mathbf{\Sigma}$ as a covariance matrix by constraining its trace or sparsity in $g(\mathbf{\Sigma})$. The trace norm regularization [43] can be formulated as an instance of problem (33).

Even though this taxonomy cannot cover all the regularized MTL methods, it can bring insights to understand regularized MTL methods better and help devise more MTL models. For example, a learning framework is proposed in [107] to learn a suitable multi-task model for a given multi-task problem under problem (33) by utilizing $\mathbf{\Sigma}$ to represent the corresponding multi-task model.

2.9 Other Settings in MTL

Instead of assuming that different tasks share an identical feature representation, Zhang and Yeung [108] 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 [108] by first projecting data in different tasks into a common subspace and then learning a common projection in this subspace to discriminate different classes in different tasks. In [109], a latent probit model is proposed to generate data of different tasks in different feature spaces via sparse transformations on a shared latent space and then to generate labels based on this latent space.

In many MTL classification problems, 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 but not columns as a media to share knowledge among tasks. 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 and task relation learning approaches, 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 [69], [110] can do that by treating a pair of data points from the same class as positive and that from different classes as 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 [108] can naturally handle them without changing the formulation. The third approach is to directly learn label correspondence among different tasks. In [111], 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 [46], [66], [94] 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 [46] and a tensor-variate normal prior [94], or factorized as a product of several low-rank matrices or tensors [66].

Most MTL methods assume that the training data in each task are stored in a data matrix. In some case, the training data in each task exhibit a multi-modal structure and hence they are represented in a tensor instead of a matrix. Multilinear multi-task methods proposed in [112], [113] can handle this situation by employing tensor trace norms as a generalization of the trace norm to perform the regularization.

2.10 Optimization Techniques in MTL

Optimization techniques used in MTL can be categorized into three main classes as follows.

- Gradient descent method and its variants: The gradient descent method can be used to optimize smooth unconstrained objective functions possessed by many MTL models. If the unconstrained objective function is non-smooth, the subgradient can be used instead and then the gradient descent method can also be used. When there are some constraints in the objective function of MTL models [43], [63], the projected gradient descent method can be used to project the updated solution in each step to the space defined by constraints. For deep MTL models, stochastic gradient descent methods can be used. Moreover, the GradNorm [114] is devised to normalize gradients to balance the learning of multiple tasks and [115] proposes the gradient surgery to avoid the interference between task gradients. Differently, [116] studies MTL from the perspective of multi-objective optimization by learning dynamic loss weights.
- Block Coordinate Descent (BCD) method: The parameters in many MTL models can be divided into several blocks. For example, parameters in learning functions of all the tasks form a block and parameters to represent task relations is from another block. Directly optimizing the objective function of such a MTL model with respect to parameters in all blocks together is not easy. The BCD method, which is also known as the alternating method, is widely used in the MTL literature, e.g., [9], [11], [30], [39], [40], [60], [61], [64], [65], [78], [79], [95], [96], to alternatively optimize each block of parameters while fixing parameters in other blocks. Hence, each step of the BCD method will solve several subproblems, each of which is to optimize with respect to a block of parameters. Compared with the original objective function, each subproblem is easier to be solved and so the BCD method can help reduce the optimization complexity.
- Proximal method [117]: For a nonsmooth objective function, which is the sum of smooth and nonsmooth functions, in an MTL model, the proximal method is frequently used (e.g., [23], [26], [62], [98], [99], [100], [101], [102], [103], [118], [119], [120]) to construct a proximal problem by replacing

the smooth function with a quadratic function that may be constructed based on its Taylor series in various ways and the resulting proximal problem is usually easier to be solved than the original problem. The proximal method can accelerate the convergence rate of the optimization process or facilitate the design of distributed optimization algorithms.

3 MTL WITH OTHER LEARNING PARADIGMS

In the previous section, we review different MTL approaches for supervised learning tasks. In this section, we overview 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 additional 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 dataset 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 supervised learning tasks. 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 of increasing the labeling cost as little 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 [121], [122], [123], multi-task active learning [124], [125], [126], and semi-supervised multi-task active learning [127]. Specifically, a semi-supervised multi-task classification model is proposed in [121], [122] to use random walk to exploit unlabeled data in each task and then cluster multiple tasks via a relaxed Dirichlet process. In [123], 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 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. [124] 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 [125], 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 [126] to consider both the risk of a multi-task learner based on the trace norm regularization and the corresponding confidence bound. In [128], 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. In [129], a proposed generalization bound is used to select a subset from multiple unlabeled tasks to acquire labels to improve the generalization performance of all the tasks. For semi-supervised multi-task active learning, Li et al. [127] propose a model to use the Fisher information as a criterion to select unlabeled data to acquire their labels with the semi-supervised multi-task classification model [121], [122] as the classifier for each task.

MTL achieves the performance improvement in not only supervised learning tasks but also unsupervised learning tasks such as clustering. In [130], 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 [131], [132], an improved version of [130] and its kernel extension are proposed to avoid the negative effect caused by the regularizer in [130] via choosing the better one between single-task and multi-task Bregman clustering. In [133], a multi-task kernel k-means method is proposed by learning the kernel matrix via both MMD between any pair of tasks and the Laplacian regularization that helps identify a smooth kernel space. In [134], two proposed multi-task clustering methods are extensions of the MTFL and MTRL methods by treating labels as cluster indicators to be learned. In [135], the principle of MTL is incorporated into the subspace clustering by capturing correlations between data instances. In [136], a multi-task clustering method belonging to instance-based MTL is proposed to share data instances among different tasks. In [137], a multi-task spectral clustering algorithm, which can handle the out-of-sample issue via a linear function to learn the cluster assignment, is proposed to achieve the feature selection among tasks via the $\ell_{2,1}$ regularization [138]. [139] proposes to identify the task cluster structure and learn task relations together.

Reinforcement Learning (RL) 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 (MRL). Some works [140], [141], [142], [143], [144], [145], [146], [147], [148], [149] adapt the ideas introduced in Section 2 to MRL. Specifically, in [140] where a task solves a sequence of Markov Decision Processes (MDPs), 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 [141], a regionalized policy representation is introduced to characterize the behavior of an agent in each task and a Dirichlet process is placed over regionalized policy representations across multiple tasks to cluster tasks. In [142], a 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. [143] 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 [144], a model associating 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 [145], a multi-task contextual bandit is introduced to leverage or learn similarities in contexts among arms to improve the prediction of rewards from contexts. In [146], a multi-task linearly solvable MDP, whose task basis matrix contains a library of component tasks shared by all the tasks, is proposed to maintain a parallel distributed representation of tasks each of which enables an agent to draw on macro actions simultaneously. In [147], a

multi-task deep RL model based on the attention can automatically group tasks into sub-networks on a state-level granularity. In [148], a sharing experience framework is introduced to use task-specific rewards to identify similar parts defined as shared-regions which can guide the experience sharing of task policies. In [149], multi-task soft option learning, a hierarchical framework based on planning as inference, is regularized by a shared prior to avoid training instabilities and allow the fine-tuning of options for new tasks without forgetting learned policies. The idea of compression and distillation have been incorporated into MRL as in [150], [151], [152], [153]. For example, in [150], 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 [151], a policy distillation method is proposed to not only train an efficient network to learn the policy of an agent but also consolidate multiple task-specific policies into a single policy. In [152], 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 [153], each task has its own policy which is constrained to be close to a shared policy that is trained by the distillation. Some works [154], [155], [156], [157], [158], [159] in MRL focus on online and distributed settings. Specifically, in [154], a distributed MRL framework is devised to model it as an instance of general consensus and an efficient decentralized solver is developed. In [155], [156], multiple goal-directed tasks are learned in an online setup without the need for expert supervision by actively sampling harder tasks. In [157], a distributed agent is developed to not only use resources more efficiently in single-machine training but also scale to thousands of machines without sacrificing data efficiency or resource utilization. [158] formulates MRL from a perspective of variational inference and it proposes a novel distributed solver with quadratic convergence guarantees. In [159], an online learning algorithm is proposed to dynamically combine different auxiliary tasks which provide gradient directions to speed up the training of the main reinforcement learning task. Some works study the theoretical foundation of MRL. For example, in [160], sharing representations among tasks is analyzed with theoretical guarantees to highlight conditions to share representations and finite-time bounds of approximated value-iteration are extended to the multi-task setting. Moreover, there are some works to design novel MRL methods. For example, in [161], a MRL framework is proposed to trains agent to employ hierarchical policies that decide when to use a previously learned policy and when to learn a new skill with a temporal grammar that helps the agent learn complex temporal dependencies. [162] studies the problem of parallel learning of multiple sequential-decision tasks and proposes to automatically adapt the contribution of each task to the updates of the agent to make all tasks have comparable impacts on the learning dynamics. In [163], a self-supervised representation learning algorithm is proposed for multi-task deep RL to capture structured information about environment dynamics based on multi-step predictive representations of future observations.

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 semi-supervised learning tasks. Multi-task multi-view learning extends multi-view learning to the MTL setting where each task is a multi-view learning problem. Specifically, in [164], 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 [165], both a regularized MTL method [70] 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 [164], [165] which study the multi-task multi-view classification problem, in [166], [167], 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 low-rank structure 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. In [168], a multi-label multi-view algorithm is proposed to not only learn common features via the $\ell_{2,1}$ regularization but also identify useless views via the Frobenius norm. 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. A deep multi-task multi-view model is proposed in [169] to fuse all the views based on the cross-stitch network.

MTL can help learn more accurate structure in graphical models. In [170], 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 [171] by assuming joint sparsity among precision matrices via the $\ell_{\infty,1}$ norm regularization. In [172], some domain knowledge about task relations is incorporated into the learning of multiple Bayesian networks. By viewing the feature interaction matrix as a form of graphical models to model pairwise relations between features, two models are proposed in [173] 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 regularization, respectively.

According to the above discussions, we can see that most research works discussed in this section follow the spirits of MTL approaches introduced in Section 2 and adapt to their own settings.

4 Handling Big Data

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 denotes the number of tasks. In this case, we can either devise online, parallel, or distributed MTL models to accelerate the learning process. Another 'big' aspect in MTL lies in the data dimensionality which can be very high. In this situation, we can speedup the learning via feature selection, dimensionality reduction, and feature hashing to reduce the dimension without losing too 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 (33) 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 (33), Zhang [174] uses the FISTA algorithm to design a surrogate function for problem (33) with a given $\mathbf{\Sigma}$, where the surrogate function is decomposable with respect to tasks, leading to a parallel design for MTL based on different loss functions including the hinge, ϵ-insensitive and square losses.

Online multi-task learning is also capable of handling a big number of tasks. In [175], [176], 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 [177], online MTL algorithms are devised by modeling the task relatedness via hard constraints that the m-tuple of actions for tasks satisfies. In [178], perceptron-based online algorithms are proposed for multi-task binary classification problems where task similarities are measured based on either the geometric closeness of the task reference vectors or the dimension of their spanned subspace. In [179], a recursive Bayesian online algorithm based on Gaussian processes is devised to update both estimations and confidence intervals when data instances arrive sequentially. In [180], an online version of the MTRL method is proposed to update both the model parameters and task covariance in a sequential way. An online multi-task learning algorithm is proposed in [181] to jointly learn the per-task model and the task relations by smoothing the loss function of each task w.r.t. a task distribution and adaptively refining this distribution over time. In [182], an online multi-task model is proposed to learn both a low-rank component and a group sparse component to characterize task relations. In [183], a multi-task passive-aggressive method is proposed to learn multiple relative similarity learning tasks, each of which is to learn a similarity function from data with relative constraints. In [184], a Gaussian distribution, whose mean or covariance consists of a local component for each task and a global component shared by all the tasks, over each task is used as a confidence measure to guide the online MTL process.

Training data can locate at different devices, making the design of distributed MTL models important. In [185], a communication-efficient distributed MTL algorithm, where each machine learns a task, based on the debiased Lasso is proposed to learn jointly sparse features in a high-dimensional space. In [186], the MTRL method (i.e., problem (21)) is extended to the distributed setting based on a stochastic dual coordinate ascent method. In [187], 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 [188], federated multi-task learning is proposed as an extension of distributed multi-task learning to consider both stragglers and fault tolerance. In [189], a distributed multi-task algorithm is proposed under the online MTL setting.

For high-dimensional data in MTL, we can use multi-task feature selection methods to reduce the dimension or extend single-task dimension reduction techniques to the multi-task setting as did in [108]. Another option is to use the feature hashing and in [190], 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, NLP, web, and so on. In Table 2, we categorize different MTL problems in each application area according to MTL approaches they used, where the classification of MTL approaches is already introduced in Section 2. In the last column of Table 2, we list some problems in various application areas which are different from other columns. For application problems listed in Table 2, application-dependent MTL models have been proposed to solve them.[5] Though those models are different from each other, there are some characteristics in respective areas. For example, in computer vision, deep MTL models, most of which belong to the feature transformation approach, exhibit good performance, making this approach popular in computer vision. In bioinformatics and health informatics, the interpretability of learning models is more important in some sense. Therefore, the feature selection and task relation learning approaches are widely used in this area as the former approach can identify useful features and the latter one can quantitatively show task relations. In speech and NLP, the data exhibit a sequential structure, which makes recurrent-neural-network-based deep MTL models in the feature transformation approach play an important role. As the data in web applications is of a large scale, this area favors simple models such as linear models or their ensembles based on boosting. Among all the MTL approaches, the feature transformation, feature selection, and task relation learning approaches are among the most widely used MTL approaches in different application areas according to Table 2.

When encountering a new application problem which can be modeled as a MTL problem, we need to judge whether tasks in this problem are related in terms of either low-level features or high-level concepts. If so, by treating Table 2 as a look-up table. we can identify a problem in Table 2 similar to the new problem and then adapt the corresponding MTL model to solve the new problem. Otherwise, we can try popular MTL approaches in the respective area.

6 Theoretical Analyses

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

The generalization bound, which is to upper-bound the generalization loss 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, and there are some works to study generalization bounds of MTL models. There are several works [39], [48], [248], [249], [250], [251], [252], [253], [254], [255], [256], [257], [258], [259] to study the generalization bound of different MTL models. In Table 3, we compare those works in terms of the analyzed MTL model, analysis tool, and the convergence rate of the corresponding bound which is based on the number of tasks (i.e., m) and the average number of data points per task (i.e., n_0). According to Table 3, we can see that some works (i.e., [48], [248], [249], [252], [253], [256], [257]) analyze different MTL models based on various analysis tools and the best convergence rate is $O(\frac{1}{\sqrt{mn_0}})$. Though MTL models analyzed in [11], [39], [250], [258] are not the same, those MTL models exhibit similar objectives to learn a linear or nonlinear

5. For details of those models, refer to an arXiv version [247] of this paper.

TABLE 2
The classification of works about MTL applications in different areas according to different MTL approaches.

Approach	Computer Vision	Bioinformatics & Health Informatics	Speech & NLP	Web	Miscellaneous
Feature Transformation	Visual tracking [191], [192] Action recognition [198] Facial landmark detection [14] Scene classification [204] Attribute prediction [205] Image rotation [207] Immediacy prediction [208] Pose estimation [18] Thumbnail selection [15] Face verification [209]	Protein subcellular location prediction [193] Protein interaction prediction [199] Biological image classification [16]	Speech synthesis [194], [195] Speech recognition [200], [201] Jointly learning of NLP tasks [203] Dialog state tracking [17] Machine translation [206] Syntactic parsing [206]	Learning to rank [196]	Stock prediction [197] Localization [202]
Feature Selection	Face and object recognition [210] Brain imaging [213]	siRNA efficacy prediction [211] Genetic marker detection [214] Mental state examination [215] Predict cognitive outcome [216] Survival analysis [119] Genetic trait prediction [217] Gene expression association [218]	Microblog analysis [118]	Behavioral targeting [212]	
Low-Rank	Image segmentation [219] Saliency detection [221]	Identification of longitudinal phenotypic markers [222]			Climate prediction [220]
Task Clustering	Image segmentation [223] Age estimation [225] Facial action unit prediction [227] Action recognition [228]	Brain-computer interfaces [224] Personalized medical treatment [226] Genetic trait prediction [217]			
Task Relation Learning		Protein subcellular location prediction [193] Organism modeling [231] MHC-I binding prediction [235] Splice-site prediction [235] Prioritization of disease genes [239] Protein interaction prediction [241] Identifying antigenic variants [242]	Sentiment classification [229]	Web search ranking [230] Collaborative filtering [232] Display advertising [236]	Localization [202] Robotics [233], [234] Trajectory regression [237] Traffic sign recognition [238] Soil moisture forecasts [240]
Decomposition	Multi-view tracking [243] Pose estimation [244] Person re-identification [246]	Genetic trait prediction [217] Protein interaction prediction [245]			

feature transformation shared by all the tasks, and the convergence rates are $O(\frac{1}{\sqrt{mn_0}})$ except [11]. The trace norm regularization (i.e., Problem (13)) are analyzed in [254], [255], [257], [259], among which [259] has the best convergence rate based on the local Rademacher complexity. A related MTL model based on the Schatten norm regularization is analyzed in [251] with an $O(\frac{1}{\sqrt{mn_0}})$ convergence rate. For the graph regularization [68], [70] analyzed in [251], [259], the local Rademacher complexity leads to a better convergence rate (i.e., $O(\frac{1}{(mn_0)^\alpha})$ for some constant $\alpha \in (0.5, 1)$) and a similar observation holds for problem (4). In a word, various analysis tools can be used to analyze MTL models and among them, the local Rademacher complexity can derive tighter generalization bounds than others.

Besides generalization bounds, there are some works to study other theoretical problems in MTL. For example, Argyriou et al. [260], [261] discuss conditions where representer theorems hold for regularized MTL algorithms. Several studies [262], [263], [264] investigate conditions to well recover true features for multi-task feature selection models.

TABLE 3
Comparison of generalization bounds derived in different works in terms of the analyzed MTL model, analysis tool, and the convergence rate of the bound. m denotes the number of tasks and n_0 denotes the average number of data points per task.

MTL Model	Reference	Analysis Tool	Convergence rate
Tasks from an environment	[248], [249]	VC dimension & Covering number	$O(1/\sqrt{mn_0})$
Task distributions can be transformed	[253]	VC dimension	$O(1/\sqrt{mn_0})$
Task clustering	[48]	VC dimension	$O(m\ln(n_0/m)/n_0)$
Multi-task kernel classifier	[256]	Covering number	$O(1/\sqrt{mn_0})$
Problem (26) with Eq. (27)	[257]	Multi-task stability	$O(m\sqrt{m}/n_0)$
Multi-task data compression	[252]	Kolmogorov complexity	$O(1/\sqrt{mn_0})$
Problem (10)	[39]	Covering number	$O(1/\sqrt{mn_0})$
Problem (10) without $\mathbf{U}$	[250]	Rademacher complexity	$O(1/\sqrt{mn_0})$
Problem (3)	[11]	Rademacher complexity	$O(1/\sqrt{n_0})$
Learn a common feature transformation	[258]	Gaussian average	$O(1/\sqrt{mn_0})$
Problem (13)	[257]	Multi-task stability	$O(m/n_0)$
	[254]	Rademacher complexity	$O(\ln(m)/\sqrt{n_0})$
	[255]	Rademacher complexity	$O(\max(\sqrt{\ln(mn_0)/mn_0}, 1/\sqrt{n_0}))$
	[259]	Local Rademacher complexity	$O(1/(mn_0)^\alpha), 0.5 < \alpha < 1$
Schatten-norm-regularized MTL models	[251]	Rademacher complexity	$O(1/\sqrt{mn_0})$
Graph regularizers [68], [70]	[251]	Rademacher complexity	$O(1/\sqrt{mn_0})$
	[259]	Local Rademacher complexity	$O(1/(mn_0)^\alpha), 0.5 < \alpha < 1$
Problem (4)	[254]	Rademacher complexity	$O(1/\sqrt{n_0})$
	[259]	Local Rademacher complexity	$O(1/(mn_0)^\alpha), 0.5 < \alpha < 1$

7 Conclusions and Discussions

In this paper, we survey different aspects of MTL. First, after giving the definition of MTL, we give a classification of supervised MTL models into five main approaches and discuss their characteristics. Then we review the combinations of MTL with other learning paradigms. The online, parallel and distributed MTL models as well as dimensionality reduction and feature hashing are discussed to speedup the learning process. The applications of MTL in various areas are introduced to show the usefulness of MTL and theoretical aspects of MTL are discussed.

In 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 that 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 belonging to the feature transformation, low-rank, task clustering, and task relation learning approaches have been proposed as reviewed in Sections 2, 3, 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 deep multi-task 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. We think that such 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.

Acknowledgments This work is supported by NSFC 62076118.

REFERENCES

[1] R. Caruana, "Multitask learning," *MLJ*, 1997.

[2] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE TKDE*, 2010.

[3] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE TKDE*, 2014.

[4] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

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

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

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

[8] D. L. Silver, R. Poirier, and D. Currie, "Inductive transfer with context-sensitive neural networks," *MLJ*, 2008.

[9] A. Argyriou, T. Evgeniou, and M. Pontil, "Convex multi-task feature learning," *MLJ*, 2008.

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

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

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

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

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

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

[16] 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.

[17] 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.

[18] S. Li, Z. Liu, and A. B. Chan, "Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network," *IJCV*, 2015.

[19] Y. Shinohara, "Adversarial multi-task learning of deep neural networks for robust speech recognition," in *Interspeech*, 2016.

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

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

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

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

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

[25] 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.

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

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

[28] 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.

[29] P. Gong, J. Ye, and C. Zhang, "Multi-stage multi-task feature learning," *JMLR*, 2013.

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

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

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

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

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

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

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

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

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

[39] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *JMLR*, 2005.

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

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

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

[43] T. K. Pong, P. Tseng, S. Ji, and J. Ye, "Trace norm regularization: Reformulations, algorithms, and multi-task learning," *SIAM Journal on Optimization*, 2010.

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

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

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

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

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

[49] B. Bakker and T. Heskes, "Task clustering and gating for Bayesian multitask learning," *JMLR*, 2003.

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

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

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

[53] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram, "Multi-task learning for classification with Dirichlet process priors," *JMLR*, 2007.

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

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

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

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

[58] 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*, 2008.

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

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

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

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

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

[64] Q. Zhou and Q. Zhao, "Flexible clustered multi-task learning by learning representative tasks," *IEEE TPAMI*, 2016.

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

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

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

[68] T. Evgeniou and M. Pontil, "Regularized multi-task learning," in *KDD*, 2004.

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

[70] T. Evgeniou, C. A. Micchelli, and M. Pontil, "Learning multiple tasks with kernel methods," *JMLR*, 2005.

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

[72] S. Feldman, M. R. Gupta, and B. A. Frigyik, "Revisiting Stein's paradox: Multi-task averaging," *JMLR*, 2014.

[73] I. Yamane, H. Sasaki, and M. Sugiyama, "Regularized multitask learning for multidimensional log-density gradient estimation," *Neural Computation*, 2016.

[74] 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.

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

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

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

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

[79] Y. Zhang and D.-Y. Yeung, "A regularization approach to learning task relationships in multitask learning," *ACM TKDD*, 2014.

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

[81] Y. Zhang and D.-Y. Yeung, "Multilabel relationship learning," *ACM TKDD*, 2013.

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

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

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

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

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

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

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

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

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

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

[92] 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.

[93] A. R. Goncalves, F. J. V. Zuben, and A. Banerjee, "Multi-task sparse structure learning with Gaussian copula models," *JMLR*, 2016.

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

[95] Y. Zhang, "Heterogeneous-neighborhood-based multi-task local learning algorithms," in *NIPS*, 2013.

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

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

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

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

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

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

[102] A. Zweig and D. Weinshall, "Hierarchical regularization cascade for joint learning," in *ICML*, 2013.

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

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

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

[106] M. Solnon, S. Arlot, and F. R. Bach, "Multi-task regression using minimal penalties," *JMLR*, 2012.

[107] Y. Zhang, Y. Wei, and Q. Yang, "Learning to multitask," in *NIPS*, 2018.

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

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

[110] P. Yang, K. Huang, and C. Liu, "Geometry preserving multi-task metric learning," *MLJ*, 2013.

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

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

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

[114] O. Sener and V. Koltun, "Multi-task learning as multi-objective optimization," in *NeurIPS*, 2018.

[115] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn, "Gradient surgery for multi-task learning," in *NeurIPS*, 2020.

[116] Z. Chen, V. Badrinarayanan, C. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *ICML*, 2018.

[117] N. Parikh and S. P. Boyd, "Proximal algorithms," *Foundations and Trends in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

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

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

[120] 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*, 2017.

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

[122] Q. Liu, X. Liao, H. Li, J. R. Stack, and L. Carin, "Semisupervised multitask learning," *IEEE TPAMI*, 2009.

[123] Y. Zhang and D. Yeung, "Semi-supervised multi-task regression," in *ECMLPKDD*, 2009.

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

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

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

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

[128] K. Lin and J. Zhou, "Interactive multi-task relationship learning," in *ICDM*, 2016.

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

[130] J. Zhang and C. Zhang, "Multitask Bregman clustering," in *AAAI*, 2010.

[131] X. Zhang and X. Zhang, "Smart multi-task Bregman clustering and multi-task kernel clustering," in *AAAI*, 2013.

[132] X. Zhang, X. Zhang, and H. Liu, "Smart multitask Bregman clustering and multitask kernel clustering," *ACM TKDD*, 2015.

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

[134] X. Zhang, "Convex discriminative multitask clustering," *IEEE TPAMI*, 2015.

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

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

[137] Y. Yang, Z. Ma, Y. Yang, F. Nie, and H. T. Shen, "Multitask spectral clustering by exploring intertask correlation," *IEEE TCYB*, 2015.

[138] X. Zhu, X. Li, S. Zhang, C. Ju, and X. Wu, "Robust joint graph sparse coding for unsupervised spectral feature selection," *IEEE TNNLS*, 2017.

[139] X. Zhang, X. Zhang, H. Liu, and J. Luo, "Multi-task clustering with model relation learning," in *IJCAI*, 2018.

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

[141] H. Li, X. Liao, and L. Carin, "Multi-task reinforcement learning in partially observable stochastic environments," *JMLR*, 2009.

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

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

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

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

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

[147] T. Bräm, G. Brunner, O. Richter, and R. Wattenhofer, "Attentive multi-task deep reinforcement learning," in *ECMLPKDD*, 2019.

[148] T. Vuong, D. V. Nguyen, T. Nguyen, C. Bui, H. Kieu, V. Ta, Q. Tran, and T. H. Le, "Sharing experience in multitask reinforcement learning," in *IJCAI*, 2019.

[149] M. Igl, A. Gambardella, N. Nardelli, N. Siddharth, W. Böhmer, and S. Whiteson, "Multitask soft option learning," in *UAI*, 2020.

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

[151] A. A. Rusu, S. G. Colmenarejo, Ç. Gülçehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell, "Policy distillation," in *ICLR*, 2016.

[152] 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.

[153] 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.

[154] S. E. Bsat, H. Bou-Ammar, and M. E. Taylor, "Scalable multitask policy gradient reinforcement learning," in *AAAI*, 2017.

[155] S. Sharma and B. Ravindran, "Online multi-task learning using active sampling," in *ICLR Workshop*, 2017.

[156] S. Sharma, A. K. Jha, P. Hegde, and B. Ravindran, "Learning to multi-task by active sampling," in *ICLR*, 2018.

[157] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, S. Legg, and K. Kavukcuoglu, "IMPALA: scalable distributed deep-RL with importance weighted actor-learner architectures," in *ICML*, 2018.

[158] R. Tutunov, D. Kim, and H. Bou-Ammar, "Distributed multitask reinforcement learning with quadratic convergence," in *NeurIPS*, 2018.

[159] X. Lin, H. S. Baweja, G. Kantor, and D. Held, "Adaptive auxiliary task weighting for reinforcement learning," in *NeurIPS*, 2019.

[160] C. D'Eramo, D. Tateo, A. Bonarini, M. Restelli, and J. Peters, "Sharing knowledge in multi-task deep reinforcement learning," in *ICLR*, 2020.

[161] T. Shu, C. Xiong, and R. Socher, "Hierarchical and interpretable skill acquisition in multi-task reinforcement learning," in *ICLR*, 2018.

[162] M. Hessel, H. Soyer, L. Espeholt, W. Czarnecki, S. Schmitt, and H. van Hasselt, "Multi-task deep reinforcement learning with PopArt," in *AAAI*, 2019.

[163] Z. Guo, B. A. Pires, M. G. Azar, B. Piot, F. Altché, J.-B. Grill, and R. Munos, "Bootstrap latent-predictive representations for multitask reinforcement learning," in *ICML*, 2020.

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

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

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

[167] X. Zhang, X. Zhang, H. Liu, and X. Liu, "Multi-task multi-view clustering," *IEEE TKDE*, 2016.

[168] X. Zhu, X. Li, and S. Zhang, "Block-row sparse multiview multilabel learning for image classification," *IEEE TCYB*, 2016.

[169] L. Zheng, Y. Cheng, and J. He, "Deep multimodality model for multi-task multi-view learning," in *SDM*, 2019.

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

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

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

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

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

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

[176] O. Dekel, P. M. Long, and Y. Singer, "Online learning of multiple tasks with a shared loss," *JMLR*, 2007.

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

[178] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile, "Linear algorithms for online multitask classification," *JMLR*, 2010.

[179] G. Pillonetto, F. Dinuzzo, and G. D. Nicolao, "Bayesian online multitask learning of Gaussian processes," *IEEE TPAMI*, 2010.

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

[181] K. Murugesan, H. Liu, J. G. Carbonell, and Y. Yang, "Adaptive smoothed online multi-task learning," in *NIPS*, 2016.

[182] P. Yang, P. Zhao, and X. Gao, "Robust online multi-task learning with correlative and personalized structures," *IEEE TKDE*, 2017.

[183] S. Hao, P. Zhao, Y. Liu, S. C. H. Hoi, and C. Miao, "Online multitask relative similarity learning," in *IJCAI*, 2017.

[184] P. Yang, P. Zhao, J. Zhou, and X. Gao, "Confidence weighted multitask learning," in *AAAI*, 2019.

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

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

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

[188] V. Smith, C. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *NIPS*, 2017.

[189] 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*, 2018.

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

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

[192] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via structured multi-task sparse learning," *IJCV*, 2013.

[193] Q. Xu, S. J. Pan, H. H. Xue, and Q. Yang, "Multitask learning for protein subcellular location prediction," *IEEE/ACM TCBB*, 2011.

[194] 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.

[195] 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.

[196] 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.

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

[198] 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.

[199] Y. Qi, O. Tastan, J. G. Carbonell, J. Klein-Seetharaman, and J. Weston, "Semi-supervised multi-task learning for predicting interactions between HIV-1 and human proteins," *Bioinformatics*, 2010.

[200] P. Bell and S. Renals, "Regularization of context-dependent deep neural networks with context-independent multi-task training," in *ICASSP*, 2015.

[201] Z. Chen, S. Watanabe, H. Erdogan, and J. R. Hershey, "Speech enhancement and recognition using multi-task learning of long short-term memory recurrent neural networks," in *InterSpeech*, 2015.

[202] 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.

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

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

[205] A. H. Abdulnabi, G. Wang, J. Lu, and K. Jia, "Multi-task CNN model for attribute prediction," *IEEE TMM*, 2015.

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

[207] 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.

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

[209] 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.

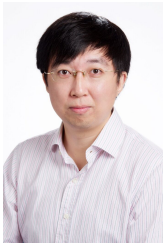
[210] X. Yuan and S. Yan, "Visual classification with multi-task joint sparse representation," in *CVPR*, 2010.

[211] 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*, 2010.

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

[213] 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.
[214] K. Puniyani, S. Kim, and E. P. Xing, "Multi-population GWA mapping via multi-task regularized regression," *Bioinformatics*, 2010.
[215] J. Zhou, L. Yuan, J. Liu, and J. Ye, "A multi-task learning formulation for predicting disease progression," in *KDD*, 2011.
[216] 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.
[217] D. He, D. Kuhn, and L. Parida, "Novel applications of multitask learning and multiple output regression to multiple genetic trait prediction," *Bioinformatics*, 2016.
[218] K. Zhang, J. W. Gray, and B. Parvin, "Sparse multitask regression for identifying common mechanism of response to therapeutic targets," *Bioinformatics*, 2010.
[219] B. Cheng, G. Liu, J. Wang, Z. Huang, and S. Yan, "Multi-task low-rank affinity pursuit for image segmentation," in *ICCV*, 2011.
[220] J. Xu, P. Tan, L. Luo, and J. Zhou, "GSpartan: a geospatio-temporal multi-task learning framework for multi-location prediction," in *SDM*, 2016.
[221] C. Lang, G. Liu, J. Yu, and S. Yan, "Saliency detection by multitask sparsity pursuit," *IEEE TIP*, 2012.
[222] 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.
[223] 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.
[224] M. Alamgir, M. Grosse-Wentrup, and Y. Altun, "Multitask learning for brain-computer interfaces," in *AISTATS*, 2010.
[225] Y. Zhang and D.-Y. Yeung, "Multi-task warped Gaussian process for personalized age estimation," in *CVPR*, 2010.
[226] J. Xu, J. Zhou, and P. Tan, "FORMULA: FactORized MUlti-task LeArning for task discovery in personalized medical models," in *SDM*, 2015.
[227] 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.
[228] A. Liu, Y. Su, W. Nie, and M. S. Kankanhalli, "Hierarchical clustering multi-task learning for joint human action grouping and recognition," *IEEE TPAMI*, 2017.
[229] F. Wu and Y. Huang, "Collaborative multi-domain sentiment classification," in *ICDM*, 2015.
[230] 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.
[231] C. Widmer, J. Leiva, Y. Altun, and G. Rätsch, "Leveraging sequence classification by taxonomy-based multitask learning," in *RECOMB*, 2010.
[232] Y. Zhang, B. Cao, and D.-Y. Yeung, "Multi-domain collaborative filtering," in *UAI*, 2010.
[233] 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.
[234] 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*, Springer, 2009.
[235] C. Widmer, N. C. Toussaint, Y. Altun, and G. Rätsch, "Inferring latent task structure for multitask learning by multiple kernel learning," *BMC Bioinformatics*, 2010.
[236] A. Ahmed, A. Das, and A. J. Smola, "Scalable hierarchical multitask learning algorithms for conversion optimization in display advertising," in *WSDM*, 2014.
[237] J. Zheng and L. M. Ni, "Time-dependent trajectory regression on road networks via multi-task learning," in *AAAI*, 2013.
[238] 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*, 2017.
[239] F. Mordelet and J. Vert, "ProDiGe: Prioritization of disease genes with multitask machine learning from positive and unlabeled examples," *BMC Bioinformatics*, 2011.
[240] J. Xu, P. Tan, J. Zhou, and L. Luo, "Online multi-task learning framework for ensemble forecasting," *IEEE TKDE*, 2017.
[241] M. Kshirsagar, J. G. Carbonell, and J. Klein-Seetharaman, "Multitask learning for host-pathogen protein interactions," *Bioinformatics*, 2013.
[242] L. Han, L. Li, F. Wen, L. Zhong, T. Zhang, and X. Wan, "Graph-guided multi-task sparse learning model: a method for identifying antigenic variants of influenza A(H3N2) virus," *Bioinformatics*, 2019.
[243] Z. Hong, X. Mei, D. V. Prokhorov, and D. Tao, "Tracking via robust multi-task multi-view joint sparse representation," in *ICCV*, 2013.
[244] 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.
[245] M. Kshirsagar, K. Murugesan, J. G. Carbonell, and J. Klein-Seetharaman, "Multitask matrix completion for learning protein interactions across diseases," *Journal of Computational Biology*, 2017.
[246] 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.
[247] Y. Zhang and Q. Yang, "A survey on multi-task learning," *CoRR*, 2017.
[248] J. Baxter, "Learning internal representations," in *COLT*, 1995.
[249] J. Baxter, "A model of inductive bias learning," *JAIR*, 2000.
[250] A. Maurer, "Bounds for linear multi-task learning," *JMLR*, 2006.
[251] A. Maurer, "The Rademacher complexity of linear transformation classes," in *COLT*, 2006.
[252] B. Juba, "Estimating relatedness via data compression," in *ICML*, 2006.
[253] S. Ben-David and R. S. Borbely, "A notion of task relatedness yielding provable multiple-task learning guarantees," *MLJ*, 2008.
[254] S. M. Kakade, S. Shalev-Shwartz, and A. Tewari, "Regularization techniques for learning with matrices," *JMLR*, 2012.
[255] M. Pontil and A. Maurer, "Excess risk bounds for multitask learning with trace norm regularization," in *COLT*, 2013.
[256] A. Pentina and S. Ben-David, "Multi-task and lifelong learning of kernels," in *ALT*, 2015.
[257] Y. Zhang, "Multi-task learning and algorithmic stability," in *AAAI*, 2015.
[258] A. Maurer, M. Pontil, and B. Romera-Paredes, "The benefit of multitask representation learning," *JMLR*, 2016.
[259] N. Yousefi, Y. Lei, M. Kloft, M. Mollaghasemi, and G. Anagnastapolous, "Local Rademacher complexity-based learning guarantees for multi-task learning," *JMLR*, 2018.
[260] A. Argyriou, C. A. Micchelli, and M. Pontil, "When is there a representer theorem? vector versus matrix regularizers," *JMLR*, 2009.
[261] A. Argyriou, C. A. Micchelli, and M. Pontil, "On spectral learning," *JMLR*, 2010.
[262] K. Lounici, M. Pontil, A. B. Tsybakov, and S. A. van de Geer, "Taking advantage of sparsity in multi-task learning," in *COLT*, 2009.
[263] G. Obozinski, M. J. Wainwright, and M. I. Jordan, "Support union recovery in high-dimensional multivariate regression," *Annals of Statistics*, 2011.
[264] M. Kolar, J. D. Lafferty, and L. A. Wasserman, "Union support recovery in multi-task learning," *JMLR*, 2011.



Yu Zhang is an associate professor in Department of Computer Science and Engineering at Southern University of Science and Technology. His research interests mainly include artificial intelligence and machine learning. He serves as a reviewer for various journals and a (senior) program committee member for several top conferences. He has won the best paper awards in UAI 2010 and PAKDD 2019, and the best student paper award in WI 2013.

Qiang Yang is a New Bright Endowed Chair Professor of Engineering in Department of Computer Science and Engineering at Hong Kong University of Science and Technology. His research interests are artificial intelligence and data mining including machine learning, planning and case-based reasoning. He is a fellow of AAAI, IEEE, IAPR, AAAS, and ACM. He was the founding Editor in Chief (EiC) of the ACM Transactions on Intelligent Systems and Technology and IEEE Transactions on Big Data. He has served as a PC co-chair and general co-chair of several top international conferences, including ACM KDD 2010 and 2012, IEEE Big Data 2013, IJCAI 2015, and AAAI 2021. He is on the board of Trustees of IJCAI and a member of the AAAI executive council.