Memetic Computing (2020) 12:355–369
https://doi.org/10.1007/s12293-020-00316-3

REGULAR RESEARCH PAPER

Multi-task gradient descent for multi-task learning

Lu Bai[1] · **Yew-Soon Ong**[1] · **Tiantian He**[1] · **Abhishek Gupta**[2]

Received: 23 September 2020 / Accepted: 1 October 2020 / Published online: 19 October 2020

Abstract
Multi-Task Learning (MTL) aims to simultaneously solve a group of related learning tasks by leveraging the salutary knowledge memes contained in the multiple tasks to improve the generalization performance. Many prevalent approaches focus on designing a sophisticated cost function, which integrates all the learning tasks and explores the task-task relationship in a predefined manner. Different from previous approaches, in this paper, we propose a novel Multi-task Gradient Descent (MGD) framework, which improves the generalization performance of multiple tasks through knowledge transfer. The uniqueness of MGD lies in assuming individual task-specific learning objectives at the start, but with the cost functions *implicitly* changing during the course of parameter optimization based on task-task relationships. Specifically, MGD optimizes the individual cost function of each task using a reformative gradient descent iteration, where relations to other tasks are facilitated through effectively transferring parameter values (serving as the computational representations of memes) from other tasks. Theoretical analysis shows that the proposed framework is convergent under any appropriate transfer mechanism. Compared with existing MTL approaches, MGD provides a novel easy-to-implement framework for MTL, which can mitigate negative transfer in the learning procedure by asymmetric transfer. The proposed MGD has been compared with both classical and state-of-the-art approaches on multiple MTL datasets. The competitive experimental results validate the effectiveness of the proposed algorithm.

Keywords Multi-task gradient descent · Knowledge transfer · Multi-task learning · Multi-label learning

1 Introduction

Inspired by human learning activities where people often apply the knowledge learned from other tasks to help learn a related task, knowledge (memes) transfer has been investigated to enhance the optimization performance in many related optimization tasks, which can be real-world problems that have similarities in nature or different methods solving one complicated problem [1,14,17,19,49]. Based on this idea, evolutionary multitasking algorithms have been developed and verified on a range of optimization tasks [3,8,18,48]. Similarly, in the community of machine learning, Multi-task Learning (MTL) solves multiple related tasks simultaneously to leverage knowledge contained in one task to help learn other tasks. MTL has shown to outperform single-task learning in many cases, from computer vision [28,54], drug discovery [21,22], to natural language processing [9], which validate the idea of utilizing knowledge from related tasks.

MTL handles multiple related tasks by jointly training them to improve generalization ability, either by shallow or deep models [37,51]. There are different mechanisms to utilize information from similar tasks in MTL. In a majority of shallow model based approaches, the similarities are promoted through regularization in the global cost function, which is composed of all the tasks, such as feature based approaches [2,6,20,31,33] and task-relation based approaches [11,16,53]. In the deep model based approaches, one way is to let different tasks share the first several hidden layers, where common feature representations for multiple tasks are learned, and then have task-specific parameters

✉ Lu Bai
bailu@ntu.edu.sg

Yew-Soon Ong
asysong@ntu.edu.sg

Tiantian He
tiantian.he@ntu.edu.sg

Abhishek Gupta
abhishek_gupta@simtech.a-star.edu.sg

[1] School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore

[2] Singapore Institute of Manufacturing Technology, Agency for Science, Technology and Research, Singapore, Singapore

in the subsequent layers [23,32,37,54]. Inspired by regularization techniques for shallow MTL, the other way of parameter sharing in deep models is to regularize the distance between the parameters of different models to encourage the parameters to be similar [10,44]. As a subproblem of MTL, multi-label learning deals with the problem that one instance is associated with multiple labels. Due to the special form of multi-label learning problems, a lot of methods have been proposed to improve the performance of multi-label learning by exploring the label correlations [46]. Classical methods such as classifier chains [36], calibrated label ranking [15], and random k-labelsets [43] transform the problem into a combination of classification problems. [24,25] considered taking the symmetric label correlations as prior knowledge and incorporating it into the model training. Local label correlations are also exploited in [26,55] by partitioning the dataset into groups and then learn local label correlations within the groups.

In most existing MTL models, the information transfer is symmetric between any two participating tasks based on the coupled cost functions, or a common feature representation is learned from all the tasks. The symmetric information sharing will deteriorate the performance of some of the participating learners since not always all tasks benefit from the joint learning [30], and the indistinguishable common feature sharing may result in performance degeneration if there are noisy and outlier tasks. Although some methods proposed for multi-label learning, such as classifier chains, can achieve asymmetric information transfer, they rely largely on the ordering or relation combination of the labels, which usually have high complexity with a large number of tasks.

Inspired by the merits of first-order gradient descent and taking into account the importance of relations among the tasks, a novel Multi-task Gradient Descent (MGD) algorithm is proposed in this paper to solve the MTL problem. In MGD, instead of modeling all the tasks into one coupled cost function, each task minimizes its individual cost function using the gradient descent algorithm. The similarities among the tasks are then facilitated through transferring model parameter values during the model learning process of each task. By implicitly changing the cost function during the learning process, MGD achieves MTL with proper transfer mechanisms. The convergence of MGD when the transfer mechanism and the step size of gradient descent satisfy certain easily achievable conditions is theoretically proven, which allows a variety of similarity measurements to be leveraged to promote information sharing. Compared with the existing approaches, the advantages of MGD are threefold:

1. MGD provides a novel easy-to-implement framework for MTL and more flexible way to conduct information transfer between related tasks.
2. It can achieve asymmetric transfer easily such that negative transfer is mitigated.
3. It can benefit from parallel computing with a small amount of information processed centrally when the number of tasks is large.

The rest of the paper is organized as follows. Section 2 briefly introduces related works, including MTL, multi-label learning, and transfer learning. Then, the proposed MGD is presented in Sect. 3, where the convergence of the proposed algorithm is theoretically proven, and the relation of MGD and the regularization based MTL is analyzed. Experiments on a synthetic problem, a set of real-world multi-label learning datasets, and MultiMNIST dataset using LeNet are conducted in Sect. 4 to evaluate the performance of the algorithm. Finally, Sect. 5 gives the conclusion and future work.

2 Related work

In the existing MTL works, information sharing is achieved by designing a common model concerning the related tasks, either is shallow or deep. In most of the deep model based MTL, the first several hidden layers are trained on all the tasks to learn common feature representations for multiple tasks, and the subsequent layers are left as task specific parameters which are only trained on a specific task [32,47,54]. The deep MTL models can learn powerful feature representations. However, the sharing hidden layers approach usually lacks of interpretability, and it is vulnerable to noisy and outlier tasks. The shallow model usually comes up with a global cost function, the relations among the tasks are promoted through a regularization term in the cost function that composed of all the tasks' parameters, such as feature based approaches [2,6,20,31,33] and task relation based approaches [11,16,53]. Taking the classical feature selection method in [33] as an example, the cost function under the regularization framework is

$$\min \sum_{i=1}^{T} f_i(\boldsymbol{w}_i) + \sigma \|W\|_{2,1},$$

where T is the number of tasks, $\boldsymbol{w}_i$ is the model parameters for the ith task, f_i denotes a function of $\boldsymbol{w}_i$ including the loss function and other regularization functions of $\boldsymbol{w}_i$, W is a matrix with the ith column being $\boldsymbol{w}_i$, and σ is a positive regularization parameter. The $\ell_{2,1}$-norm regularization on W equals the sum of ℓ_2 norm of rows in W, which enforces W to be row sparse and results in selecting important features shared across multiple tasks. In this kind of regularization based approaches, same shared features are used for all the participating tasks.

Treating a single-label learning problem as one task, the multi-label learning problem can be seen as a special case of MTL problem, where the feature vectors $\boldsymbol{x}_j$ for $j = 1, \ldots, n$ are the same for different tasks. The relations between the tasks can be learned from the relations between the labels. For first-order methods, the label correlations are ignored and the multi-label learning problem is handled in a label by label manner, such as BR [5] and LIFT [45]. Second-order methods consider pairwise relations between labels, such as LLSF [24] and JFSC [25]. High-order methods, where high-order relations among label subsets or all the labels are considered, such as RAkEL [43], ECC [36], LLSF-DL [24], and CAMEL [13]. Generally, the higher the order of correlations being considered, the stronger is the correlation-modeling capabilities, while on the other hand, the more computationally demanding and less scalable the approach becomes.

In contrast to the existing MTL approaches which incorporate correlation information into the modeling in the form of regularization or shared hidden layers, MGD serves as the first attempt to incorporate the correlations by transferring model parameter values during the model learning process of each task. Unlike the multi-label learning methods which rely on the ordering or relation combination of the labels to achieve asymmetric information transfer, MGD can achieve asymmetric transfer easily even with a large number of tasks. When considering only one task receive information from other tasks in an asymmetric transfer manner, it seems similar to transfer learning [7,34], however, they have significant difference. The objective in transfer learning is to improve the performance of a target task with the help of source tasks which are already learned, while in MGD with only one task receives information, all the tasks are simultaneously solved. The asymmetric information transfer is utilized to mitigate negative transfer.

3 The MGD approach

In this section, we elaborate the proposed MGD algorithm for MTL. The mathematical notations used in the manuscript are first introduced. We then generically formulate the MTL problem and introduce the proposed MGD. At last, we perform the theoretical analysis of MGD, including convergence proof, analysis on relation with regularization based MTL, and computational complexity.

Throughout this paper, normal font small letters denote scalars, boldface small letters denote column vectors, and capital letters denote matrices. $\mathbf{0}$ denotes zero column vector with proper dimension, I_n denotes identity matrix of size $n \times n$. A' denotes the transpose of matrix A and $\otimes$ denotes the Kronecker product. $[z_i]_{\text{vec}}$ denotes a concatenated column vector formed by stacking z_i on top of each other, and $\text{diag}\{z_i\}$ denotes a diagonal matrix with the ith diagonal element being z_i. The norm $\|\cdot\|$ without specifying the subscript represents the Euclidean norm by default.

3.1 Problem formulation

Suppose we have T tasks to be solved simultaneously. For simplicity, we assume that all the tasks share the common data space and the tasks are positively correlated. Each task $i \in \{1, \ldots, T\}$ aims to solve the following minimization problem,

$$\min_{\boldsymbol{w}_i} \ f_i(\boldsymbol{w}_i), \tag{1}$$

where $\boldsymbol{w}_i \in \mathbb{R}^d$ is the model parameter and $f_i : \mathbb{R}^d \to \mathbb{R}$ is the cost function of the ith task. In this paper, we do not restrict the specific form of the cost functions. In particular, the cost functions $f_i(\boldsymbol{w}_i)$ is assumed to be strongly convex, twice differentiable, and the gradient of f_i is Lipschitz continuous with constant L_{f_i}, i.e.,

$$\|\nabla f_i(\boldsymbol{u}) - \nabla f_i(\boldsymbol{v})\| \leq L_{f_i}\|\boldsymbol{u} - \boldsymbol{v}\|, \quad \forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d.$$

Cost functions in machine learning problems such as mean squared error with ℓ_2 norm regularization and cross-entropy with ℓ_2 norm regularization apply. Non-differentiable cost functions where ℓ_1 norm regularization is used can also be approximately considered [40]. Since $f_i(\boldsymbol{w}_i)$ is strongly convex and twice differentiable, there exists positive constant ξ_i such that $\nabla^2 f_i(\boldsymbol{u}) \geq \xi_i I_d$. As a result, we have

$$\xi_i I_d \leq \nabla^2 f_i(\boldsymbol{u}) \leq L_{f_i} I_d, \ \forall \boldsymbol{u} \in \mathbb{R}^d.$$

3.2 The proposed framework

Using the gradient descent, problem (1) can be solved using the following iteration,

$$\boldsymbol{w}_i^{t+1} = \boldsymbol{w}_i^t - \alpha \nabla f_i(\boldsymbol{w}_i^t), \tag{2}$$

where t is the iteration index, α is the step size, and $\nabla f_i(\boldsymbol{w}_i^t) \in \mathbb{R}^d$ is the gradient of f_i at $\boldsymbol{w}_i^t$. As there are relations among tasks, we are able to improve the learning performance by considering the correlation of parameters belonging to different tasks. Based on this idea, we propose a reformative gradient descent iteration, which allows the values of the model parameters to be transferred across tasks during each iteration. The MGD is designed as follows,

$$\boldsymbol{w}_i^{t+1} = \sum_{j=1}^{T} m_{ij}^t \boldsymbol{w}_j^t - \alpha \nabla f_i(\boldsymbol{w}_i^t), \ i = 1, \ldots, T, \tag{3}$$

where m_{ij}^t is the transfer coefficient describes the information flow from task j to task i, which satisfies the following conditions,

$$m_{ij}^t \geq 0, \tag{4a}$$

$$\sum_{j=1}^{T} m_{ij}^t = 1. \tag{4b}$$

From (3), we can see that under MGD, the learning of different tasks can be decoupled with only a small amount of information need to be transferred among the tasks, thus can benefit from parallel computing.

From (4b), we have $m_{ii}^t = 1 - \sum_{j\neq i} m_{ij}^t$. Rewriting iteration (3) as follows

$$\begin{aligned}\boldsymbol{w}_i^{t+1} &= m_{ii}^t \boldsymbol{w}_i^t + \sum_{j\neq i} m_{ij}^t \boldsymbol{w}_j^t - \alpha \nabla f_i(\boldsymbol{w}_i^t) \\ &= \left(1 - \sum_{j\neq i} m_{ij}^t\right) \boldsymbol{w}_i^t + \sum_{j\neq i} m_{ij}^t \boldsymbol{w}_j^t - \alpha \nabla f_i(\boldsymbol{w}_i^t).\end{aligned}$$

Rescale m_{ij}^t as

$$\bar{m}_{ij}^t = \begin{cases} \frac{1}{\alpha\sigma} m_{ij}^t, & j \neq i, \\ 1 - \frac{1}{\alpha\sigma} \sum_{j\neq i} m_{ij}^t, & j = i, \end{cases} \tag{5}$$

where σ is a positive constant and satisfies the condition

$$1 - \frac{1}{\alpha\sigma} \sum_{j\neq i} m_{ij}^t > 0. \tag{6}$$

Given (5), m_{ij}^t is parameterized by σ. With the rescaling, the iteration in (3) can be alternatively expressed as

$$\begin{aligned}\boldsymbol{w}_i^{t+1} &= \left(1 - \alpha\sigma \sum_{j\neq i} \bar{m}_{ij}^t\right) \boldsymbol{w}_i^t + \alpha\sigma \sum_{j\neq i} \bar{m}_{ij}^t \boldsymbol{w}_j^t - \alpha \nabla f_i(\boldsymbol{w}_i^t) \\ &= \boldsymbol{w}_i^t - \alpha\sigma(1 - \bar{m}_{ii}^t)\boldsymbol{w}_i^t + \alpha\sigma \sum_{j\neq i} \bar{m}_{ij}^t \boldsymbol{w}_j^t - \alpha \nabla f_i(\boldsymbol{w}_i^t) \\ &= (1 - \alpha\sigma)\boldsymbol{w}_i^t + \alpha\sigma \sum_{j=1}^{T} \bar{m}_{ij}^t \boldsymbol{w}_j^t - \alpha \nabla f_i(\boldsymbol{w}_i^t).\end{aligned} \tag{7}$$

3.3 Convergence analysis

In this section, we give the convergence property of the proposed MGD iteration based on the expression in (7).

Denote $\boldsymbol{w}_i^*$ as the underlying target model parameter for task i, $\tilde{\boldsymbol{w}}_i{}^t = \boldsymbol{w}_i^t - \boldsymbol{w}_i^*$, and $\bar{L}_{f_i} = \max_i\{L_{f_i}\}$. The following theorem gives the convergence property of the iteration (7) under certain conditions on the transfer parameter $\bar{m}_{ij}^t$ and step-size α.

Theorem 1 *Under the iteration in (7) with the transfer coefficient $\bar{m}_{ij}^t$ satisfying*

$$\sum_{j=1}^{T} \bar{m}_{ij}^t = 1, \ \forall i,$$

$$\bar{m}_{ij}^t \geq 0, \ \forall i, j,$$

$\boldsymbol{w}_i^t$ is convergent if the step size α is chosen to satisfy

$$0 < \alpha < \frac{2}{2\sigma + \bar{L}_{f_i}}. \tag{8}$$

Specifically,

$$\lim_{t\to\infty} \max_i \|\tilde{\boldsymbol{w}}_i{}^t\| \leq \frac{\alpha\sigma \max_{i,j} \|\boldsymbol{w}_i^* - \boldsymbol{w}_j^*\| + \alpha \max_i \|\nabla f_i(\boldsymbol{w}_i^*)\|}{1 - (\bar{\gamma} + \alpha\sigma)}, \tag{9}$$

where $\bar{\gamma} = \max_i\{|1 - \alpha\sigma - \alpha\xi_i|, |1 - \alpha\sigma - \alpha L_{f_i}|\}$.

Proof Let the i, j-th element of $\bar{M}^t \in \mathbb{R}^{T\times T}$ at iteration time t being $\bar{m}_{ij}^t$, denote $\bar{\mathcal{M}}^t = \bar{M}^t \otimes I_d \in \mathbb{R}^{dT\times dT}$, $\mathbf{w} = [\boldsymbol{w}_1^{'}, \ldots, \boldsymbol{w}_T^{'}]' \in \mathbb{R}^{dT}$, and $\nabla f(\mathbf{w}^t) = [\nabla f_1(\boldsymbol{w}_1^t)', \ldots, \nabla f_T(\boldsymbol{w}_T^t)']' \in \mathbb{R}^{dT}$. Note that we are using the typeface $\mathbf{w}$ to distinguish this from the single vector-valued variable $\boldsymbol{w}_i$. Write (7) into a concatenated form gives

$$\mathbf{w}^{t+1} = (1 - \alpha\sigma)\mathbf{w}^t + \alpha\sigma\bar{\mathcal{M}}^t\mathbf{w}^t - \alpha\nabla f(\mathbf{w}^t). \tag{10}$$

Denote $\mathbf{w}^* = [\boldsymbol{w}_1^{*'}, \ldots, \boldsymbol{w}_T^{*'}]'$ and $\tilde{\mathbf{w}}^t = \mathbf{w}^t - \mathbf{w}^*$. Subtracting $\mathbf{w}^*$ from both sides of (10) gives

$$\begin{aligned}\tilde{\mathbf{w}}^{t+1} &= ((1 - \alpha\sigma)I_{dT} + \alpha\sigma\bar{\mathcal{M}}^t)\mathbf{w}^t - \mathbf{w}^* - \alpha\nabla f(\mathbf{w}^t) \\ &= ((1 - \alpha\sigma)I_{dT} + \alpha\sigma\bar{\mathcal{M}}^t)\tilde{\mathbf{w}}^t - \alpha(\nabla f(\mathbf{w}^t) - \nabla f(\mathbf{w}^*)) \\ &\quad + \alpha(\sigma(\bar{\mathcal{M}}^t - I_{dT})\mathbf{w}^* - \nabla f(\mathbf{w}^*)) \\ &= ((1 - \alpha\sigma)I_{dT} + \alpha\sigma\bar{\mathcal{M}}^t)\tilde{\mathbf{w}}^t \\ &\quad - \alpha\int_0^1 \nabla^2 f(\mathbf{w}^* + \mu(\mathbf{w}^t - \mathbf{w}^*))d\mu\tilde{\mathbf{w}} \\ &\quad + \alpha(\sigma(\bar{\mathcal{M}}^t - I_{dT})\mathbf{w}^* - \nabla f(\mathbf{w}^*)) \\ &= ((1 - \alpha\sigma)I_{dT} + \alpha\sigma\bar{\mathcal{M}}^t - \alpha H^t)\tilde{\mathbf{w}}^t \\ &\quad + \alpha(\sigma(\bar{\mathcal{M}}^t - I_{dT})\mathbf{w}^* - \nabla f(\mathbf{w}^*)),\end{aligned} \tag{11}$$

where $H^t = \int_0^1 \nabla^2 f(\mathbf{w}^* + \mu(\mathbf{w}^t - \mathbf{w}^*))d\mu \in \mathbb{R}^{dT\times dT}$. It can be verified that H^t is a block diagonal matrix and the block diagonal elements $H_i^t = \int_0^1 \nabla^2 f_i(\boldsymbol{w}_i^* + \mu(\boldsymbol{w}_i^t - \boldsymbol{w}_i^*))d\mu \in \mathbb{R}^{d\times d}$ for $i = 1, \ldots, T$ are Hermitian.

We use the block maximum norm defined in [39] to show the convergence of the above iteration. The block maximum

norm of a vector $\boldsymbol{x} = [\boldsymbol{x}_i]_{\text{vec}} \in \mathbb{R}^{dT}$ with $\boldsymbol{x}_i \in \mathbb{R}^d$ is defined as [39]

$$\|\boldsymbol{x}\|_{b,\infty} = \max_i \|\boldsymbol{x}_i\|.$$

The induced matrix block maximum norm is therefore defined as [39]

$$\|A\|_{b,\infty} = \max_{\boldsymbol{x}\neq 0} \frac{\|A\boldsymbol{x}\|_{b,\infty}}{\|\boldsymbol{x}\|_{b,\infty}}.$$

From the iteration in (11) we have

$$\begin{aligned}
\|\tilde{\mathbf{w}}^{t+1}\|_{b,\infty} &\leq \|((1-\alpha\sigma)I_{dT} + \alpha\sigma\bar{\mathcal{M}}^t - \alpha H^t)\tilde{\mathbf{w}}^t\|_{b,\infty} \\
&\quad + \alpha\|\sigma(\bar{\mathcal{M}}^t - I_{dT})\mathbf{w}^* - \nabla f(\mathbf{w}^*)\|_{b,\infty} \\
&\leq \|(1-\alpha\sigma)I_{dT} + \alpha\sigma\bar{\mathcal{M}}^t - \alpha H^t\|_{b,\infty}\|\tilde{\mathbf{w}}^t\|_{b,\infty} \\
&\quad + \alpha\|\sigma(\bar{\mathcal{M}}^t - I_{dT})\mathbf{w}^* - \nabla f(\mathbf{w}^*)\|_{b,\infty} \\
&\leq (\|(1-\alpha\sigma)I_{dT} - \alpha H^t\|_{b,\infty} + \alpha\sigma\|\bar{\mathcal{M}}^t\|_{b,\infty})\|\tilde{\mathbf{w}}^t\|_{b,\infty} \\
&\quad + \alpha\|\sigma(\bar{\mathcal{M}}^t - I_{dT})\mathbf{w}^* - \nabla f(\mathbf{w}^*)\|_{b,\infty}.
\end{aligned}$$

From Lemma D.3 in [39], we have

$$\|\bar{\mathcal{M}}^t\|_{b,\infty} = \|\bar{M}^t\|_\infty = 1,$$

where the last equality comes from the fact that $\bar{m}^t_{ij} \geq 0$ and the row summation of $\bar{M}^t$ is one. Since $\xi_i I_d \leq \nabla^2 f_i(\boldsymbol{w}_i) \leq L_{f_i} I_d$, $\xi_i I_d \leq \int_0^1 \nabla^2 f_i(\boldsymbol{w}_i^* + \mu(\boldsymbol{w}_i - \boldsymbol{w}_i^*))d\mu \leq L_{f_i} I_d$. Thus, $\|(1-\alpha\sigma)I_d - \alpha H_i^t\| \leq \gamma_i$ where $\gamma_i = \max\{|1-\alpha\sigma - \alpha\xi_i|, |1-\alpha\sigma-\alpha L_{f_i}|\}$. By the definition of induced matrix block maximum norm, we have

$$\begin{aligned}
&\|(1-\alpha\sigma)I_{dT} - \alpha H^t\|_{b,\infty} \\
&= \max_{\boldsymbol{x}\neq 0} \frac{\|((1-\alpha\sigma)I_{dT} - \alpha H^t)\boldsymbol{x}\|_{b,\infty}}{\|\boldsymbol{x}\|_{b,\infty}} \\
&\leq \max_{\boldsymbol{x}\neq 0} \frac{\max_i \|((1-\alpha\sigma)I_d - \alpha H_i^t)\|\|\boldsymbol{x}\|_{b,\infty}}{\|\boldsymbol{x}\|_{b,\infty}} \\
&= \max_i \|(1-\alpha\sigma)I_d - \alpha H_i^t\| \\
&\leq \bar{\gamma},
\end{aligned}$$

where $\bar{\gamma} = \max\{\gamma_i\}$. Thus,

$$\begin{aligned}
\|\tilde{\mathbf{w}}^{t+1}\|_{b,\infty} &\leq (\bar{\gamma} + \alpha\sigma)\|\tilde{\mathbf{w}}^t\|_{b,\infty}\alpha\sigma\|(\bar{\mathcal{M}}^t - I_{dT})\mathbf{w}^*\|_{b,\infty} \\
&\quad + \alpha\|\nabla f(\mathbf{w}^*)\|_{b,\infty}. \qquad (12)
\end{aligned}$$

By choosing the step size α to satisfy $\bar{\gamma} + \alpha\sigma < 1$, the iteration asymptotically converges. To ensure $\bar{\gamma} + \alpha\sigma < 1$, it is sufficient to ensure

$$\begin{aligned}
&|1-\alpha\sigma-\alpha\xi_i| + \alpha\sigma < 1 \text{ and} \\
&|1-\alpha\sigma-\alpha L_{f_i}| + \alpha\sigma < 1, \ \forall i,
\end{aligned}$$

which leads to

$$0 < \alpha < \frac{2}{2\sigma + \bar{L}_{f_i}}.$$

Since $\bar{M}^t$ is row-sum-to-one and the elements of $\bar{M}^t$ are all non-negative, the elements in $\bar{\mathcal{M}}^t\mathbf{w}^*$ are convex combinations of $\boldsymbol{w}_i^*$. Thus, $\|(\bar{\mathcal{M}}^t - I_{dT})\mathbf{w}^*\|_{b,\infty}$ is upper bounded by $\max_{i,j}\|\boldsymbol{w}_i^* - \boldsymbol{w}_j^*\|$. From the iteration in (12), we have

$$\begin{aligned}
&\|\tilde{\mathbf{w}}^{t+1}\|_{b,\infty} \leq (\bar{\gamma} + \alpha\sigma)^{t+1}\|\tilde{\mathbf{w}}^0\|_{b,\infty} \\
&+ (\alpha\sigma \max_{i,j}\|\boldsymbol{w}_i^* - \boldsymbol{w}_j^*\| + \alpha\|\nabla f(\mathbf{w}^*)\|_{b,\infty})\sum_{k=0}^{t}(\bar{\gamma} + \alpha\sigma)^k.
\end{aligned}$$

Under the condition that $\bar{\gamma} + \alpha\sigma < 1$,

$$\lim_{t\to\infty}\|\tilde{\mathbf{w}}^t\|_{b,\infty} \leq \frac{\alpha\sigma\max_{i,j}\|\boldsymbol{w}_i^* - \boldsymbol{w}_j^*\| + \alpha\|\nabla f(\mathbf{w}^*)\|_{b,\infty}}{1 - (\bar{\gamma} + \alpha\sigma)}.$$

From the definition of block maximum norm, (9) is obtained. □

In iteration (3), the transfer coefficient m_{ij}^t between task i and task j is a scalar. In the following, we consider the element-wise feature similarities between task i and task j. The transfer coefficient between task i and task j is assumed to be a diagonal matrix $P_{ij} \in \mathbb{R}^{d\times d}$ with its k-th diagonal element $P_{ij,k}$ being the transfer coefficient from the k-th element of $\boldsymbol{w}_j$ to the k-th element of $\boldsymbol{w}_i$. The MGD iteration in (3) then becomes

$$\boldsymbol{w}_i^{t+1} = \sum_{j=1}^{T} P_{ij}^t \boldsymbol{w}_j^t - \alpha\nabla f_i(\boldsymbol{w}_i^t), \qquad (13)$$

where

$$\begin{aligned}
&\sum_{j=1}^{T} P_{ij}^t = I_d, \\
&P_{ij,k}^t \geq 0, \ \forall i, j = 1,\ldots,T, k = 1,\ldots,d. \qquad (14)
\end{aligned}$$

Following the same rescaling,

$$\bar{P}_{ij}^t = \begin{cases} \frac{1}{\alpha\sigma}P_{ij}^t, & j \neq i, \\ I_d - \frac{1}{\alpha\sigma}\sum_{j\neq i}P_{ij}^t, & j = i, \end{cases} \qquad (15)$$

(13) becomes

$$\boldsymbol{w}_i^{t+1} = (1-\alpha\sigma)\boldsymbol{w}_i^t + \alpha\sigma\sum_{j=1}^{T}\bar{P}_{ij}^t\boldsymbol{w}_j^t - \alpha\nabla f_i(\boldsymbol{w}_i^t). \qquad (16)$$

Corollary 1 *Under* (16) *with the transfer coefficient* $\bar{P}_{ij}^t$ *satisfies*

$$\sum_{j=1}^{T} \bar{P}_{ij}^t = I_d,$$
$$\bar{P}_{ij,k}^t \geq 0,\ \forall i, j = 1, \ldots, T, k = 1, \ldots, d,$$

$\boldsymbol{w}_i^t$ *is convergent if the following conditions are satisfied:*

$$\sigma < \frac{\bar{L}_{f_i}}{T-1},\ \text{for } T > 1,$$
$$0 < \alpha < \frac{2}{(T+1)\sigma + \bar{L}_{f_i}}.$$

Proof Let the i, j-th block element of $\bar{\mathcal{P}}^t \in \mathbb{R}^{dT \times dT}$ being $\bar{P}_{ij}^t \in \mathbb{R}^{d\times d}$. Following the similar procedure of the proof of Theorem 1, we obtain

$$\begin{aligned}\|\tilde{\mathbf{w}}^{t+1}\|_{b,\infty} \leq (&\|(1-\alpha\sigma)I_{dT} - \alpha H^t\|_{b,\infty} \\ &+ \alpha\sigma\|\bar{\mathcal{P}}^t\|_{b,\infty})\|\tilde{\mathbf{w}}^t\|_{b,\infty} \\ &+ \alpha\|\sigma(\bar{\mathcal{P}}^t - I_{dT})\mathbf{w}^* - \nabla f(\mathbf{w}^*)\|_{b,\infty}. \end{aligned} \quad (17)$$

Let $\boldsymbol{x} = [\boldsymbol{x}_i]_{\text{vec}} \in \mathbb{R}^{dT}$ being a block column vector with $\boldsymbol{x}_i \in \mathbb{R}^d$.

$$\begin{aligned}\|\bar{\mathcal{P}}^t\boldsymbol{x}\|_{b,\infty} &= \max_i \|\sum_{j=1}^{T} \bar{P}_{ij}^t \boldsymbol{x}_j\| \\ &\leq \max_i \sum_{j=1}^{T} \|\bar{P}_{ij}^t\|\|\boldsymbol{x}_j\| \\ &\leq \left(\max_i \sum_{j=1}^{T} \|\bar{P}_{ij}^t\|\right) \max_j \|\boldsymbol{x}_j\|.\end{aligned}$$

Recall that $\bar{P}_{ij}^t$ is a diagonal matrix and the elements therein are all no greater than 1, thus, $\sum_{j=1}^{T} \|\bar{P}_{ij}^t\| \leq T$. As a result

$$\|\bar{\mathcal{P}}^t\boldsymbol{x}\|_{b,\infty} \leq T \max_j \|\boldsymbol{x}_j\|.$$

By the definition of matrix block maximum norm, we have

$$\|\bar{\mathcal{P}}^t\|_{b,\infty} \leq T.$$

The condition to ensure convergence of the iteration in (17) becomes

$$\bar{\gamma} + \alpha\sigma T < 1,$$

which gives

$$\sigma < \frac{\bar{L}_{f_i}}{T-1},\ \text{for } T \neq 1,$$
$$0 < \alpha < \frac{2}{(T+1)\sigma + \bar{L}_{f_i}}.$$

□

3.4 Relation with regularization based multi-task learning

From the iteration in (7), we have

$$\begin{aligned}\boldsymbol{w}_i^{t+1} &= (1-\alpha\sigma)\boldsymbol{w}_i^t \\ &\quad + \alpha\sigma\sum_{j=1}^{T} \bar{m}_{ij}^t \boldsymbol{w}_j^t - \alpha\nabla f_i(\boldsymbol{w}_i^t) \\ &= \boldsymbol{w}_i^t - \alpha\left(\sigma\sum_{j=1}^{T} \bar{m}_{ij}^t(\boldsymbol{w}_i^t - \boldsymbol{w}_j^t) + \nabla f_i(\boldsymbol{w}_i^t)\right). \end{aligned} \quad (18)$$

If fix $\bar{m}_{ij}^t = \bar{m}_{ij}$ for all t, then, the last term in the brackets can be seen as the gradient of the following function

$$\bar{f}_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}) = f_i(\boldsymbol{w}_i) + \frac{1}{2}\sigma\sum_{j=1}^{T} \bar{m}_{ij}\|\boldsymbol{w}_i - \boldsymbol{w}_j\|^2,$$

where $\boldsymbol{w}_{-i}$ denotes the collection of other tasks' variables, i.e., $\boldsymbol{w}_{-i} = [\boldsymbol{w}_1', \ldots, \boldsymbol{w}_{i-1}', \boldsymbol{w}_{i+1}', \ldots, \boldsymbol{w}_T']'$. Thus, the iteration in (18) with fixed $\bar{m}_{ij}$ can be seen as the gradient descent algorithm which solves the following Nash equilibrium problem

$$\min_{\boldsymbol{w}_i}\ \bar{f}_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}),\ i = 1, \ldots, T. \quad (19)$$

In (19), each task's cost function is influenced by other tasks' decision variables. Since the cost function $\bar{f}_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i})$ is continuous in all its arguments, strongly convex with respect to $\boldsymbol{w}_i$ for fixed $\boldsymbol{w}_{-i}$, and satisfies $\bar{f}_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}) \to \infty$ as $\|\boldsymbol{w}_i\| \to \infty$ for fixed $\boldsymbol{w}_{-i}$, a Nash equilibrium exists [4]. Furthermore, as a result of strongly convexity, the gradient of $\bar{f}_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i})$ with respect to $\boldsymbol{w}_i$ for fixed $\boldsymbol{w}_{-i}$ is strongly monotone. Thus, the Nash equilibrium for (19) is unique [12]. Denote the Nash equilibrium of (19) as $\boldsymbol{w}_i^o, i = \{1, \ldots, T\}$. It is known that the Nash equilibrium satisfies the following condition [4]:

$$\boldsymbol{w}_i^o = \text{argmin}_{\boldsymbol{w}_i} \bar{f}_i(\boldsymbol{w}_i, \boldsymbol{w}_{-i}^o),\ i = 1, \ldots, T,$$

which implies

$$\nabla f_i(\boldsymbol{w}_i^o) + \sigma \sum_{j=1}^{T} \bar{m}_{ij}(\boldsymbol{w}_i^o - \boldsymbol{w}_j^o) = 0,\ i = 1, \ldots, T. \quad (20)$$

Write the conditions in (20) in a concatenated form gives

$$\nabla f(\mathbf{w}^o) + \sigma(I_T - \bar{M}) \otimes I_d \mathbf{w}^o = 0. \quad (21)$$

It has been pointed out in [51] that the regularized MTL algorithms which learn with task relations can be expressed as

$$\min_{\boldsymbol{w}_i, \Sigma} \sum_{i=1}^{T} L_i(\boldsymbol{w}_i) + \frac{1}{2}\lambda \mathbf{w}^T(\Sigma^{-1} \otimes I_d)\mathbf{w} + g(\Sigma), \quad (22)$$

where L_i is the training loss of task i, λ is a positive regularization parameter, $\Sigma \in \mathbb{R}^{T\times T}$ models the task relations, and $g(\Sigma)$ denotes constraints on Σ. For comparison, we eliminate the constraints on Σ, consider the case that Σ is fixed, and let $f(\mathbf{w}) = \sum_{i=1}^{T} L_i(\boldsymbol{w}_i)$. Denote the optimal solutions of problem (22) as $\mathbf{w}^g$. The optimal solution satisfies the following condition,

$$\nabla f(\mathbf{w}^g) + \frac{1}{2}\lambda(\Sigma^{-1} + (\Sigma^{-1})^T) \otimes I_d \mathbf{w}^g = 0. \quad (23)$$

Comparing the optimality conditions (21) and (23) for the Nash equilibrium problem (19) and the MTL problem (22), we find that if $\bar{M}$ can be set as

$$\sigma(I_T - \bar{M}) = \frac{1}{2}\lambda(\Sigma^{-1} + (\Sigma^{-1})^T), \quad (24)$$

the optimal solution $\mathbf{w}^o$ will be the same as $\mathbf{w}^g$. The only limitation is that $\bar{m}_{ij} > 0$, which can not cover the situation where there exists non-negative non-diagonal values in Σ^{-1}. Overall, the regularized multi-tasking learning problem with task relation learning can be solved by the MGD algorithm by setting the coefficients $\bar{m}_{ij}$ between task i and task j properly. In addition, using MGD, we can consider feature-feature relations between different tasks since we can use $\bar{P}_{ij} \in \mathbb{R}^{d\times d}$ as the transfer coefficient. Furthermore, in MGD, $\bar{m}_{ij}$ is not required to be equal to $\bar{m}_{ji}$. This relaxation allows asymmetric task relations in MTL [30], which is hard to achieve by most MTL methods since $\Sigma^{-1} + (\Sigma^{-1})^T$ is always symmetric in (23).

Another category of regularized MTL method is learning with feature relations [51]. The cost function of this kind of method is

$$\min_{\boldsymbol{w}_i, \Theta} \sum_{i=1}^{T} L_i(\boldsymbol{w}_i) + \frac{1}{2}\lambda \mathbf{w}^T(I_T \otimes \Theta^{-1})\mathbf{w} + g(\Theta), \quad (25)$$

where $\Theta \in \mathbb{R}^{d\times d}$ models the covariance between the features. The term $\mathbf{w}^T(I_T \otimes \Theta^{-1})\mathbf{w}$ can be decoupled as $\sum_{i=1}^{T} \boldsymbol{w}_i^T \Theta^{-1} \boldsymbol{w}_i$, which can be incorporated into $f_i(\boldsymbol{w}_i)$ for task i.

3.5 Pseudocode of the MGD algorithm

MGD provides a novel framework of utilizing task similarities to improve the learning performance. The pseudocode of the proposed MGD is summarized in Algorithm 1.

Algorithm 1 Pseudocode of MGD for Task i

Require:
The multi-task training set $\mathcal{D}_i$, $i = 1, ..., T$
Hyperparameters in cost function f_i for $i = 1, ..., T$ and σ
Step size α, random initial values $\boldsymbol{w}_i^1$ for $i = 1, ..., T$
Ensure:
Model parameter $\boldsymbol{w}_i^*$ for $i = 1, ..., T$
1: Set $t = 0$, initialize m_{ij}^0 and $\boldsymbol{w}_i^0$
2: **repeat**
3: Calculate the gradient $\nabla f_i(\boldsymbol{w}_i^t)$, $i = 1, ..., T$;
4: Calculate the transfer coefficient m_{ij}^t, $j \in \mathcal{N}_i$;
5: Update $\boldsymbol{w}_i$ according to (3): $\boldsymbol{w}_i^{t+1} = \sum_{j=1}^{T} m_{ij}^t \boldsymbol{w}_j^t - \alpha \nabla f_i(\boldsymbol{w}_i^t)$;
6: $t = t + 1$;
7: **until** Stop criterion reached;
8: **return** $\boldsymbol{w}_i^* = \boldsymbol{w}_i^{t+1}$, $i = 1, ..., T$.

As shown in Theorem 1, the transfer coefficients are only required to satisfy mild conditions to ensure the convergence of the proposed algorithm, which allows a variety of existing task relation learning methods to be used to design the transfer coefficients. A straightforward way to set the transfer coefficient is based on task similarities. The more similar two tasks are, the larger the corresponding transfer coefficient is expected to be. For a multi-label learning problem, the similarity between task i and task j can be modeled by the correlation between the label sets, which can be calculated by many different similarity measurements, such as cosine similarity and Euclidean distance between the labels. By assuming the task relations be known as a prior like in [11,27], the transfer coefficient can be designed utilizing the relation between MGD and the regularization based MTL. In addition to set the transfer coefficients as a predefined value according to prior assumption or statistical methods, the value of m_{ij}^t can also be learned during the learning of the parameters. Methods placing a prior on the learning parameters like in [50,52,53] can also be utilized to design the transfer coefficients. Furthermore, the regularization based MTL methods usually result in symmetric task relations, while MGD can achieve asymmetric transfer easily for a specific problem such that negative transfer can be mitigated.

3.6 Complexity analysis

We analyze the complexity of the iteration MGD using predefined transfer coefficients. In each iteration, the gradient calculation leads to a complexity of $\mathcal{O}(g(d)nT)$, where $g(d)$ is the complexity of calculating the gradient w.r.t. the dimension d, which is determined by the cost function used, and the update of the model parameter according to (3) needs $\mathcal{O}(dT^2)$. Therefore, the overall complexity of the MGD algorithms is of order $\mathcal{O}(t(ng(d)T + dT^2))$, where t is the iteration time.

4 Experiments

In this section, we evaluate the MGD algorithm on different types of MTL problems, including regression problems, multi-label learning as multi-task learning problems, and a deep neural network model. Specifically, we first conduct a simple linear regression using synthetic datasets to demonstrate the effect of MGD compared to single-task gradient descent. Then, we validate the effectiveness of MGD for the multi-label learning problem on a series of real-world multi-label learning datasets, and compare it with both classical and state-of-the-art algorithms. Finally, we use MGD for digit classification on the MultiMNIST dataset, an MTL version of the MNIST dataset, based on LeNet. To show the effectiveness of the proposed framework, the transfer coefficients are set manually or through statistical methods, which are simple but already effective. More sophisticated transfer coefficients can be used to further improve the performance.

4.1 Toy problem

We first validate our approach by an experiment with synthetic datasets. We generate two synthetic datasets for regression that have same mean but different variances. Specifically, the noise level for the first task is set to be low while that for the second task is set to be high as follows,

$$T_1 : y_{1j} = w^* x_{1j} + \delta,\ j = 1, \ldots, n,$$
$$T_2 : y_{2j} = w^* x_{2j} + 10\delta,\ j = 1, \ldots, n,$$

where n is the number of data samples and $\delta \sim \mathcal{N}(0, 1)$. Taking the Mean Squared Error (MSE) as the cost function, denote $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ as the data vector for task i, by minimizing the cost function, Single-task Gradient Descent (SGD) gives the following iteration

$$w_i^{t+1} = w_i^t - \frac{\alpha_i}{n} \boldsymbol{x}_i^T (w_i^t \boldsymbol{x}_i - \boldsymbol{y}_i),\ i = 1, 2.$$

Under MGD, the iteration is

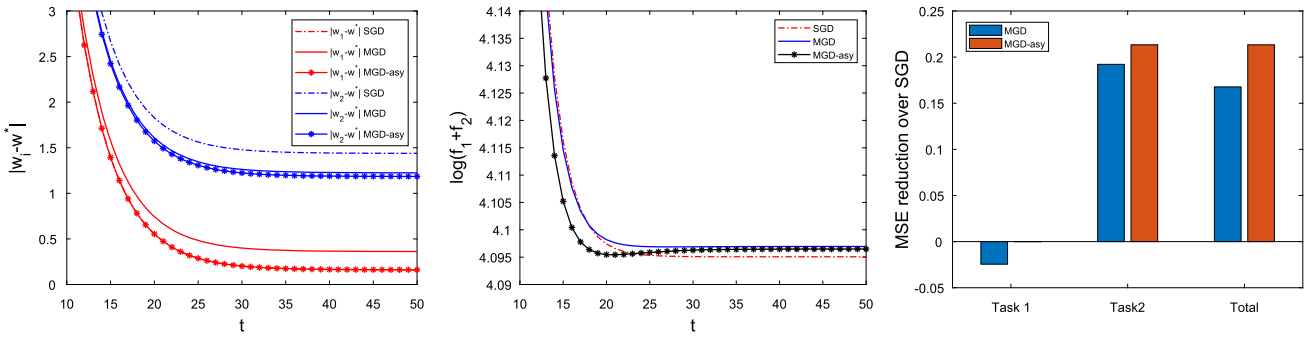
$$w_i^{t+1} = \sum_{j=1}^{2} m_{ij}^t w_j^t - \frac{\alpha_i}{n} \boldsymbol{x}_i^T (w_i^t \boldsymbol{x}_i - \boldsymbol{y}_i),\ i = 1, 2.$$

Note that although we can directly obtain the analytic expression which minimizes the MSE for this simple problem, iteration method is more commonly used for most problems, and we use iteration here to showcase the difference between MGD and SGD.

We set $w^* = 2$, $\alpha_1 = \alpha_2 = 0.2$. We generate 100 points for each task from the standard Gaussian distribution and use 5-folder split for training and test, and run the iteration 50 times. First, we use a fixed transfer coefficients $m_{12} = m_{21} = 0.05$ for symmetric transfer. Then, since we already know that task 2 has a higher noisy level than task 1, it is believed that transfer information from task 2 to task 1 will probably cause negative transfer, thus, we conduct asymmetric transfer with transfer coefficients $m_{12} = 0.0001$ and $m_{21} = 0.05$.

The results are shown in Fig. 1.

Figure 1a shows the updates of the absolute error between w_i and w^*, it can be seen that the parameter of task 2 under both MGD and MGD-asy converge to better solutions than under SGD, while the parameter of task 1 under MGD converges to a solution worse than under SGD, and that of MGD-asy converges to a solution similar with SGD. This indicates that a negative transfer from task 2 to task 1 exists when the transfer is symmetric, and it can be mitigated by asymmetric transfer. Comparing the results obtained by MGD and MGD-asy, we can see that the solution for task 1 improves more under MGD-asy than MGD. This is due to the better solution obtained for task 1 under asymmetric transfer. The update of the log of the total training loss is shown in Fig. 1b. As can be seen, all the iterations converge to steady states, while the steady states are different. SGD produces the lowest training loss since it directly minimizes the cost function. However, note that the cost functions are defined for each task separately, which ignore task similarities, the one which gives the lowest loss doesn't guarantee the best performance. The different steady state values produced by MGD and MGD-asy showcase that the information transfer between the tasks implicitly changes the cost function, which may give better solution when the transfer is properly designed. The MSE reduction of MGD and MGD-asy over SGD on the test set is shown in Fig. 1c. The result shows that MGD has a higher MSE than SGD on task 1, while this negative transfer is suppressed by asymmetric transfer in MGD-asy. Nevertheless, looking at the total MSE, both MGD and MGD-asy have better performance than SGD.



(a) Update of $|w_i - w_i^*|$ **(b)** Update of $\log(\sum_{i=1}^{2} f_i)$ **(c)** MSE reduction over SGD of the test set.

Fig. 1 Linear regression under SGD and MGD

4.2 Multi-label classification

Multi-label learning deals with the problem that one instance is associated with multiple labels. Given the multi-label training set $\mathcal{D} = \{(\boldsymbol{x}_j, \boldsymbol{y}_j)|1 \le j \le n\}$, where n is number of instances, $\boldsymbol{x}_j \in \mathcal{X}$ is the feature vector for the j-th instance and $\boldsymbol{y}_j \in \{0, 1\}^T$ is the set of labels associated with the j-th instance. The task of multi-label learning is to learn a function h from $\mathcal{D}$ which can assign a set of proper labels to an instance. We decompose the multi-label learning problem into T binary classification tasks. For each of the classification tasks, we use the 2-norm regularized logistic loss as the cost function. Thus, for any task i, the following cost function is optimized by each task individually,

$$f_i(\boldsymbol{w}_i) = -\frac{1}{n}\sum_{j=1}^{n}(y_j^i \log h(z_j^i) + (1 - y_j^i)\log(1 - h(z_j^i))) + \frac{1}{2}\rho\|\boldsymbol{w}_{i,-1}\|^2,$$

where $h(z_j^i) = P(y_j^i = 1|x_j) = \frac{1}{1+e^{-z_j^i}}$, $z_j^i = [1\ \boldsymbol{x}_j^T]\boldsymbol{w}_i$, $\boldsymbol{w}_i \in \mathbb{R}^{p+1}$ is the model parameter, $\boldsymbol{w}_{i,-1} \in \mathbb{R}^p$ is the remaining elements in $\boldsymbol{w}_i$ except the first element, and ρ is the regularization parameter. Let

$$X = \begin{bmatrix} 1\ \boldsymbol{x}_1^T \\ \vdots\ \ \vdots \\ 1\ \boldsymbol{x}_n^T \end{bmatrix}, \boldsymbol{y}^i = \begin{bmatrix} y_1^i \\ \vdots \\ y_n^i \end{bmatrix}.$$

The MGD iteration is

$$\boldsymbol{w}_i^{t+1} = \sum_j^T m_{ij}^t \boldsymbol{w}_j^t - \frac{\alpha}{n}X^T(g(X\boldsymbol{w}_i^t) - \boldsymbol{y}^i) - \alpha\rho\begin{bmatrix} 0 \\ \boldsymbol{w}_{i,-1} \end{bmatrix}, \quad (26)$$

where $g(X\boldsymbol{w}_i^t) = [\frac{1}{1+e^{-[1\ \boldsymbol{x}_1^T]\boldsymbol{w}_i^t}}, \ldots, \frac{1}{1+e^{-[1\ \boldsymbol{x}_n^T]\boldsymbol{w}_i^t}}]'$.

In multi-label learning problems, the similarity between task i and task j can be modeled by the correlation between labels $\boldsymbol{y}^i$ and $\boldsymbol{y}^j$. In this experiment, we use the cosine similarity to calculate the correlation matrix C, i.e., $C_{ij} = \frac{<\boldsymbol{y}^i, \boldsymbol{y}^j>}{\|\boldsymbol{y}^i\|\|\boldsymbol{y}^j\|}$. Then, we normalize each row of the correlation matrix C to be row-sum-to-one and set $\bar{m}_{ij} = C_{ij}$. Finally, we rescale $\bar{m}_{ij}$ according to (5) to get the transfer coefficient m_{ij}.

After learning the model parameter $\boldsymbol{w}_i^*$, we can predict the label y_t^i for a test instance $\boldsymbol{x}_t$ by the corresponding prediction function. The ith label prediction for an instant $\boldsymbol{x}_t$ is predicted 1 if $h(z_t^i) \ge \eta$ and 0 otherwise, where η is the threshold. In the experiment, η is chosen from $\{0.1, 0.2, 0.3\}$.

4.2.1 Experimental setup

We conduct the multi-label classification on six benchmark multi-label datasets, including regular-scale datasets: emotions, genbase, and cal500; and relatively large-scale datasets: enron, corel5k, and bibtex. The details of the datasets are summarized in Table 1, where $|S|$, $\dim(S)$, $L(S)$, $\text{Card}(S)$, and $\text{Dom}(S)$ represent the number of examples, the number of features, the number of class labels, the average number of labels per example, and feature type of dataset S, respectively. The datasets are downloaded from the website of Mulan[1] [42].

Five widely used evaluation metrics are employed to evaluate the performance, including Average precision, Macro-averaging F1, Micro-averaging F1, Coverage score, and Ranking loss. Concrete metric definitions can be found in [46]. Note that for the comparison purpose, the coverage score is normalized by the number of labels. For Average precision, Macro averaging F1, and Micro averaging F1, the larger the values the better the performance. For the other two metrics, the smaller the values the better the performance.

[1] http://mulan.sourceforge.net/datasets-mlc.html

Table 1 Characteristics of the tested multi-label datasets

Dataset	$\|S\|$	dim(S)	L(S)	Card(S)	Dom(S)
emotions	593	72	6	1.869	Music
genbase	662	1186	27	1.252	Biology
cal500	502	68	174	26.044	Music
enron	1702	1001	53	3.378	Text
corel5k	5000	499	374	3.522	Images
bibtex	7395	1836	159	2.402	Text

$|S|$ represents the number of examples, dim(S) represents the number of features, L(S) represents the number of class labels, Card(S) represents the average number of labels per example, and Dom(S) represents feature type of dataset S

We compare our proposed method MGD with three classical algorithms including BR [5], RAkEL [43], ECC [36], and two state-of-the-art multi-label learning algorithms LIFT [45] and LLSF-DL [24].

In the experiments, we used the source codes provided by the authors for implementation. BR, ECC, and RAkEL are implemented under the Mulan multi-label learning package [42] using the logistic regression model as the base classifier. Parameters suggested in the corresponding literatures are used, i.e., RAkEL: ensemble size $2T$ with $k = 3$; ECC: ensemble size 30; LIFT: the ratio parameter r is tuned in {0.1,0.2,…,0.5}; LLSF-DL: α, β, γ are searched in $\{4^{-5}, 4^{-4}, \ldots, 4^{5}\}$, and ρ is searched in $\{0.1, 1, 10\}$. For the proposed approach MGD, α is set to 0.02, ρ is chosen from $\{0.1, 0.2, \ldots, 1\}$, and σ is chosen from $\{0, 0.05, 0.1, 0.15, \ldots, 0.3\}$.

4.2.2 Experimental results

We run the algorithms 5 times on five sets of randomly partitioned training (80%) and testing (20%) data, the mean metric values with standard deviations are recorded in Tables 2 and 3. The best performance is shown in boldface, ↑ indicates the larger the better, and ↓ indicates the smaller the better. From the results, we can see that MGD outperforms other comparing algorithms in most cases. Specifically, MGD ranks first in 86.7% cases. Compared with the existing algorithms, MGD introduced a new approach to incorporate label correlations, which is easy to implement and has low complexity. The results demonstrate the effectiveness of the proposed approach in improving the learning performance.

Compared to single gradient descent, the transfer in MGD also helps to accelerate the convergence. To see this, the iterations of the total loss, i.e., $\sum_{i=1}^{T} f_i(\boldsymbol{w}_i)$, are plotted in the first row of Fig. 2 for three datasets. It can be seen that the MGD converges faster than single gradient descent, especially at early iterations. The iterations of the average precision score are also plotted in the second row of Fig. 2. It can be seen that for limited iteration times, the score under MGD is much better than single gradient descent.

Table 2 Prediction performance (mean ± std. deviation) on the regular-scale tested datasets

Dataset	emotions	genbase	cal500
Algorithm	Average precision ↑		
MGD	**0.815 ± 0.014**	**0.994 ± 0.006**	**0.516 ± 0.012**
BR	0.783 ± 0.027	0.985 ± 0.009	0.323 ± 0.009
RAkEL	0.782 ± 0.030	0.575 ± 0.032	0.143 ± 0.003
ECC	0.774 ± 0.030	0.992 ± 0.005	0.437 ± 0.007
LIFT	0.734 ± 0.013	0.535 ± 0.031	0.502 ± 0.009
LLSF-DL	0.710 ± 0.018	0.619 ± 0.053	0.470 ± 0.023
Algorithm	Macro-averaging F1 ↑		
MGD	**0.668 ± 0.015**	0.652 ± 0.076	**0.191 ± 0.003**
BR	0.619 ± 0.037	**0.915 ± 0.036**	0.155 ± 0.007
RAkEL	0.629 ± 0.034	0.661 ± 0.021	0.060 ± 0.011
ECC	0.622 ± 0.033	0.904 ± 0.042	0.158 ± 0.010
LIFT	0.432 ± 0.017	0.026 ± 0.003	0.045 ± 0.002
LLSF-DL	0.123 ± 0.024	0.006 ± 0.003	0.143 ± 0.006
Algorithm	Micro-averaging F1 ↑		
MGD	**0.679 ± 0.014**	0.966 ± 0.023	**0.481 ± 0.010**
BR	0.632 ± 0.035	**0.974 ± 0.010**	0.331 ± 0.005
RAkEL	0.644 ± 0.035	0.740 ± 0.038	0.073 ± 0.004
ECC	0.636 ± 0.031	0.926 ± 0.015	0.357 ± 0.008
LIFT	0.506 ± 0.012	0.219 ± 0.025	0.316 ± 0.004
LLSF-DL	0.199 ± 0.015	0.038 ± 0.022	0.459 ± 0.014
Algorithm	Coverage score ↓		
MGD	**0.291 ± 0.016**	**0.009 ± 0.004**	0.740 ± 0.005
BR	0.314 ± 0.029	0.016 ± 0.008	0.803 ± 0.007
RAkEL	0.332 ± 0.024	0.370 ± 0.022	0.983 ± 0.002
ECC	0.327 ± 0.036	0.013 ± 0.005	0.794 ± 0.007
LIFT	0.358 ± 0.007	0.161 ± 0.829	0.748 ± 0.012
LLSF-DL	0.373 ± 0.023	0.175 ± 0.036	**0.733 ± 0.007**
Algorithm	Ranking loss ↓		
MGD	**0.152 ± 0.010**	**0.001 ± 0.001**	**0.176 ± 0.003**
BR	0.180 ± 0.027	0.005 ± 0.004	0.243 ± 0.005
RAkEL	0.194 ± 0.027	0.361 ± 0.024	0.604 ± 0.004
ECC	0.194 ± 0.036	0.002 ± 0.002	0.222 ± 0.003
LIFT	0.233 ± 0.007	0.138 ± 0.023	0.181 ± 0.012
LLSF-DL	0.254 ± 0.025	0.161 ± 0.033	0.198 ± 0.010

Best performance is shown in boldface

We investigate the sensitivity of MGD with respect to the two hyperparameters ρ and σ, which control the norm 2 regularization strength in the logistic regression and the transfer strength. Due to space limit, we only report the results on the emotions dataset using the average precision score. Figure 3 shows how the average precision score varies with respect to ρ and σ. Figure 3b, c are obtained by keeping the other parameter fixed at its best setting. It can be seen that both ρ

Table 3 Prediction performance (mean ± std. deviation) on the large-scale tested datasets

Dataset	enron	corel5k	bibtex
Algorithm	Average precision ↑		
MGD	**0.704 ± 0.016**	**0.326 ± 0.010**	**0.596 ± 0.007**
BR	0.384 ± 0.009	0.132 ± 0.004	0.199 ± 0.009
RAkEL	0.168 ± 0.005	0.119 ± 0.004	0.323 ± 0.008
ECC	0.554 ± 0.014	0.232 ± 0.006	0.441 ± 0.011
LIFT	0.696 ± 0.011	0.289 ± 0.005	0.566 ± 0.010
LLSF-DL	0.635 ± 0.018	0.271 ± 0.008	0.593 ± 0.004
Algorithm	Macro-averaging F1 ↑		
MGD	0.226 ± 0.016	0.051 ± 0.003	**0.336 ± 0.003**
BR	0.206 ± 0.021	0.148 ± 0.007	0.136 ± 0.004
RAkEL	0.112 ± 0.012	0.162 ± 0.012	0.202 ± 0.008
ECC	**0.252 ± 0.017**	**0.208 ± 0.014**	0.256 ± 0.009
LIFT	0.141 ± 0.011	0.024 ± 0.001	0.218 ± 0.014
LLSF-DL	0.195 ± 0.017	0.040 ± 0.003	0.210 ± 0.006
Algorithm	Micro-averaging F1 ↑		
MGD	**0.602 ± 0.014**	**0.291 ± 0.009**	**0.413 ± 0.002**
BR	0.356 ± 0.013	0.120 ± 0.003	0.145 ± 0.006
RAkEL	0.182 ± 0.008	0.132 ± 0.003	0.211 ± 0.007
ECC	0.457 ± 0.014	0.091 ± 0.007	0.352 ± 0.009
LIFT	0.560 ± 0.012	0.077 ± 0.004	0.378 ± 0.009
LLSF-DL	0.548 ± 0.019	0.249 ± 0.015	0.397 ± 0.010
Algorithm	Coverage score ↓		
MGD	**0.218 ± 0.015**	**0.292 ± 0.003**	**0.105 ± 0.003**
BR	0.259 ± 0.012	0.704 ± 0.007	0.426 ± 0.013
RAkEL	0.819 ± 0.006	0.864 ± 0.005	0.366 ± 0.012
ECC	0.292 ± 0.011	0.433 ± 0.007	0.236 ± 0.011
LIFT	0.224 ± 0.011	**0.292 ± 0.005**	0.137 ± 0.006
LLSF-DL	0.336 ± 0.013	0.486 ± 0.012	0.185 ± 0.008
Algorithm	Ranking loss ↓		
MGD	**0.075 ± 0.007**	0.136 ± 0.003	**0.056 ± 0.001**
BR	0.308 ± 0.010	0.368 ± 0.007	0.274 ± 0.008
RAkEL	0.587 ± 0.005	0.573 ± 0.007	0.222 ± 0.008
ECC	0.119 ± 0.005	0.192 ± 0.002	0.134 ± 0.008
LIFT	0.077 ± 0.006	**0.123 ± 0.002**	0.075 ± 0.004
LLSF-DL	0.130 ± 0.007	0.238 ± 0.005	0.097 ± 0.004

Best performance is shown in boldface

and σ influence the performance. While, under a relatively wide range of parameters combinations, the score does not vary too much.

In the above experiment, we use the cosine similarity on the label set to calculate the correlation matrix C, which results in symmetric correlation. To see the effect of asymmetric transfer, we take the emotions dataset as an example and impose asymmetric correlation on the labels. There are six labels in the emotions datasets representing amazed-surprised (L1), happy-pleased (L2), relaxing-calm (L3), quiet-still (L4), sad-longly (L5), and angry-fearful (L6). Based on the ease of predictions, which ranked in the following descending order L4, L6, L5, L1, L3, L2, we added a vector $[3/21, 1/21, 2/21, 6/21, 4/21, 5/21]$ on each row of the cosine similarity matrix, to make more information transferred from easier tasks to harder tasks, and less information transferred vice versa. We compare the evaluation metrics obtained by using the above asymmetric correlation matrix (MGD-asy) and cosine similarity (MGD) in Table 4.

As can be seen from Table 4, MGD with asymmetric transfer improves the performance in all the evaluation metrics. The classification accuracy for each label is further compared in Table 5. It can be seen that the classification accuracy for both easier and harder predicted labels are improved under asymmetric transfer.

4.3 MultiMNIST

MultiMNIST is an MTL version of the MNIST dataset [38], where multiple images are overlaid together to convert digit classification into a multi-task problem. We use the construction from [41]. For each image, a different one is chosen uniformly in random. Then one of the images is put at the top-left and the other is at the bottom-right with partially overlapping. The resulting two tasks are classifying the digit on the top-left and classifying the digit on the bottom-right with the transformed images as the input. We use the dataset created by [41], which contains 60K examples.

The LeNet [29] is used as the base model for each task. During the optimization procedure, we transfer the parameters of the first two convolution layers, and leave the fully

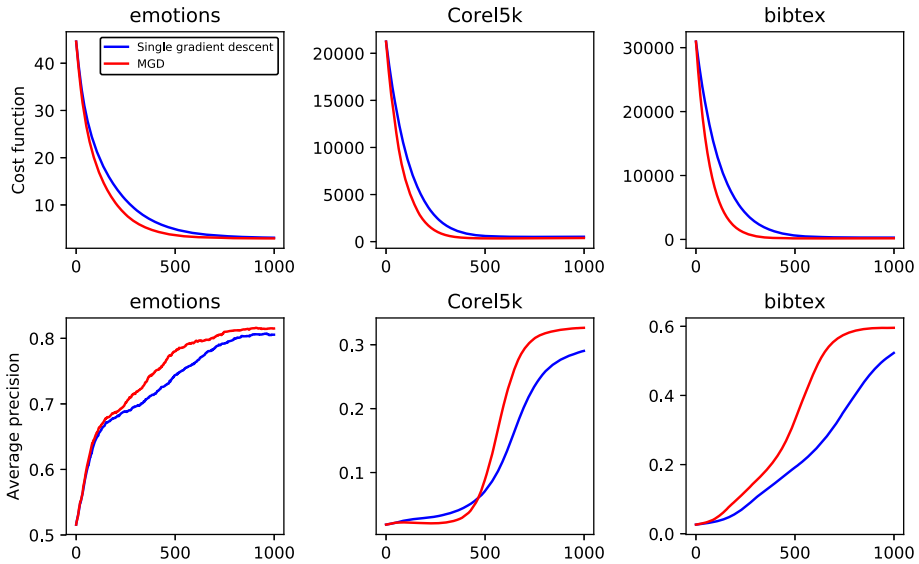


Fig. 2 Convergence test

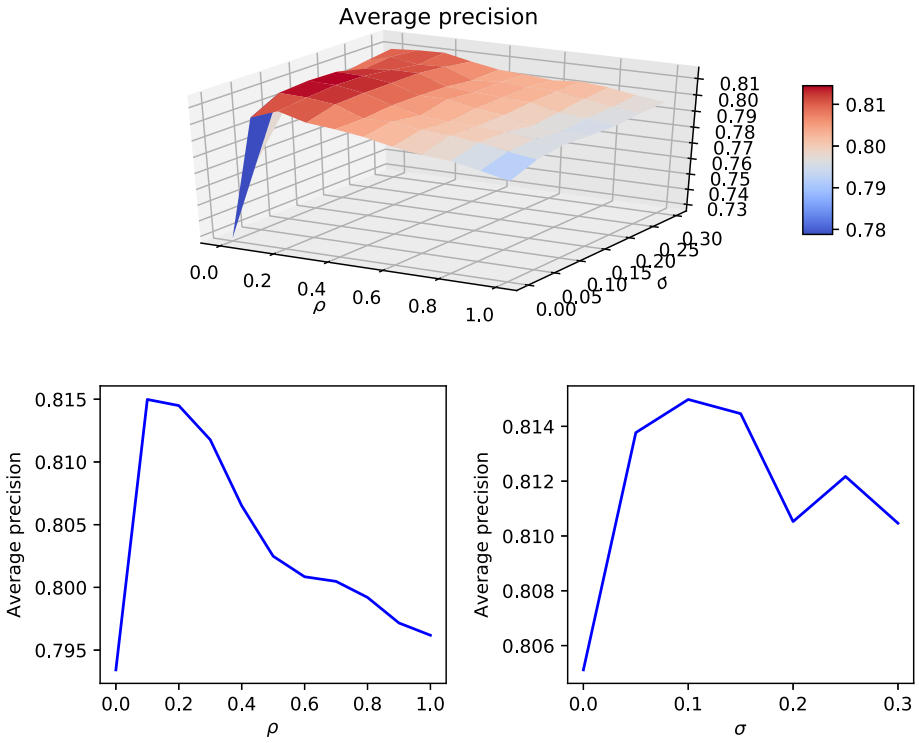


Fig. 3 Sensitivity analysis on the emotions dataset

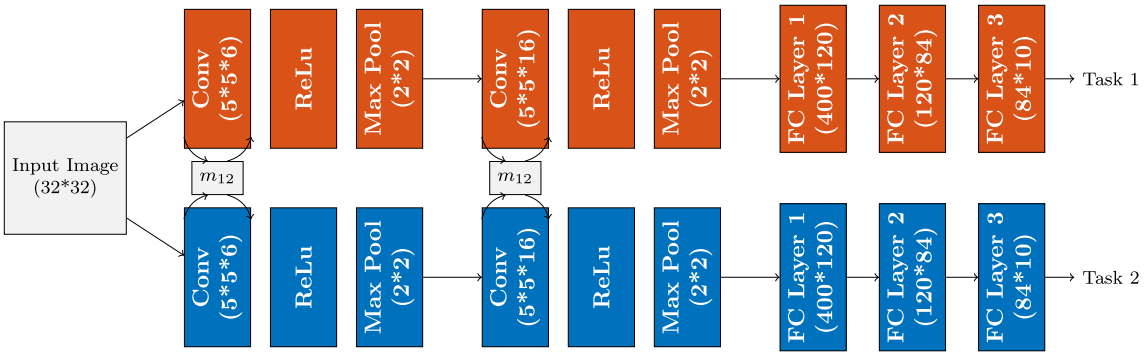


Fig. 4 Architecture of the multiMNIST experiment

Table 4 Prediction performance (mean± std. deviation) on the emotions dataset obtained by asymmetric MGD and MGD

emotions	MGD-asy	MGD
Average precision ↑	**0.816 ± 0.012**	0.815 ± 0.014
Macro-averaging F1 ↑	**0.672 ± 0.015**	0.668 ± 0.015
Micro-averaging F1 ↑	**0.682 ± 0.014**	0.679 ± 0.014
Coverage score ↓	**0.289 ± 0.016**	0.291 ± 0.016
Ranking loss ↓	**0.150 ± 0.009**	0.152 ± 0.010

Table 5 Classification accuracy for each label on the emotions dataset obtained by asymmetric MGD and MGD

emotions	MGD-asy	MGD
L1	**0.74534**	0.74531
L2	**0.65268**	0.64596
L3	**0.76227**	0.75891
L4	**0.85828**	0.85321
L5	**0.75720**	0.75045
L6	**0.76904**	0.76899
Avg	**0.75747**	0.75381

connected layers without transfer. The architecture is visualized in Fig. 4. For both tasks, the cross-entropy loss is used as the cost function. We adapt the SGD and modify it with the multi-task transfer as the optimizer. The transfer parameter between the two tasks is set based on the Euclidean distance (Euc) between the two label sets, which is $1/(Euc + 1)$, where the scaler 1 in the dominator is to ensure the transfer parameter less than 1. The rescale parameter σ is set as 1.

To show the effectiveness, each single task is solved solely by SGD to serve as the baseline. Furthermore, based on the code by the author, we run the MTL method proposed in *Sener and Koltun 2018* [41], which trained the first two convolution layers and the first fully connected layers as a shared encoder and two independent fully-connected layers as task-specific function for the two tasks, as a comparison. For all the experiments, the learning rate is set as 0.001 and halved every 30 epoches, the momentum is set as 0.9. We use batch size of 256 and train for 100 epoches. The results averaged over 5 runs are listed in Table 6.

As can be seen from the results in Table 6, our method performs the best. Specifically, compared with single task baseline, MGD which superimpose transfer on single-task learning achieves a better performance, which showcases the effectiveness of utilizing information from related tasks. Compared with the model-based MTL method [38] which uses the first two convolutional layers and one fully connected layer as shared layers, our method also achieves better results on both tasks. The result validates the efficacy of our method which promotes relations between related tasks through transferring parameter values during the model learning process.

5 Conclusion and future work

In this paper, we propose the MGD algorithm for MTL. Different from the state-of-the-art, MGD treats multi-task learning as multiple learning tasks with independent cost functions, and transfers correlated model parameter values during the model learning process of the independent cost

Table 6 Performance on the MultiMNIST dataset

Approaches	Left digit accuracy (%)	Right digit accuracy (%)
Single task	95.90 ± 0.03%	94.32 ± 0.01%
Sener and Koltun 2018 [38]	94.94 ± 0.00%	93.51 ± 0.00%
Ours	**95.96 ± 0.03%**	**94.39 ± 0.05%**

functions. By implicitly changing the cost function through the learning process, MGD achieves utilizing information from related tasks with proper transfer mechanisms. The convergence of the algorithm has been theoretically proven for any transfer mechanism satisfying easily achievable conditions, which provides flexibility in using different kinds of similarity measurements. Compared to existing MTL approaches, MGD is easy to implement, can achieve seamless asymmetric transformation such that negative transfer is mitigated, and can benefit from parallel computing when the number of tasks is large. The competitive experimental results validate the effectiveness of MGD. In our current work, we only require the transfer coefficients to satisfy easily achievable conditions and utilize simple similarity measurements such as cosine similarity to find task relations in the experiment. It is desirable to design more effective and systematic learning of transfer coefficients to improve the performance, including asymmetric transfer. Besides, it is also interesting to investigate element-wise feature-feature relations rather than only task-task relations in the future.

Acknowledgements This work were supported in part by the A*STAR Cyber-Physical Production System (CPPS)-Towards Contextual and Intelligent Response Research Program, under the RIE2020 IAF-PP Grant A19C1a0018, the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2018-004), and Data Science & Artificial Intelligence Research Centre, Nanyang Technological University. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore.

References

1. Amaya JE, Cotta C, Fernández-Leiva AJ, García-Sánchez P (2020) Deep memetic models for combinatorial optimization problems: application to the tool switching problem. Memet Comput 12(1):3–22
2. Argyriou A, Evgeniou T, Pontil M (2007) Multi-task feature learning. Adv Neural Inf Process Syst 20:41–48
3. Bali KK, Ong Y-S, Gupta A, Tan PS (2019) Multifactorial evolutionary algorithm with online transfer parameter estimation: Mfea-ii. IEEE Trans Evol Comput 24(1):69–83
4. Basar T, Olsder GJ (1999) Dynamic noncooperative game theory, vol 23. SIAM, Philadelphia
5. Boutell MR, Luo J, Shen X, Brown CM (2004) Learning multi-label scene classification. Pattern Recogn 37(9):1757–1771
6. Chen J, Tang L, Liu J, Ye J (2009) A convex formulation for learning shared structures from multiple tasks. In: Proceedings of the 26th annual international conference on machine learning. ACM, pp 137–144
7. Deng Z, Lu J, Wu D, Choi K-S, Sun S, Nojima Y (2019) Guest editorial: special issue on new advances in deep-transfer learning. IEEE Trans Emerg Top Comput Intell 3(5):357–359
8. Dinh TP, Thanh BHT, Ba TT, Binh LN (2020) Multifactorial evolutionary algorithm for solving clustered tree problems: competition among cayley codes. Memet Comput 12(3):185–217
9. Dong D, Wu H, He W, Yu D, Wang H (2015) Multi-task learning for multiple language translation. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing, pp 1723–1732
10. Duong L, Cohn T, Bird S, Cook P (2015) Low resource dependency parsing: cross-lingual parameter sharing in a neural network parser. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 2: short papers), pp 845–850
11. Evgeniou T, Pontil M (2004) Regularized multi–task learning. In: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 109–117
12. Facchinei F, Pang J-S (2007) Finite-dimensional variational inequalities and complementarity problems. Springer, Berlin
13. Feng L, An B, He S (2019) Collaboration based multi-label learning. In: Thirty-third AAAI conference on artificial intelligence
14. Feng L, Ong Y-S, Tan A-H, Tsang IW (2015) Memes as building blocks: a case study on evolutionary optimization + transfer learning for routing problems. Memet Comput 7(3):159–180
15. Fürnkranz J, Hüllermeier E, Mencía EL, Brinker K (2008) Multilabel classification via calibrated label ranking. Mach Learn 73(2):133–153
16. Görnitz N, Widmer C, Zeller G, Kahles A, Rätsch G, Sonnenburg S (2011) Hierarchical multitask structured output learning for large-scale sequence segmentation. Adv Neural Inf Process Syst 24:2690–2698
17. Gupta A, Ong Y-S (2019) Memetic computation: the mainspring of knowledge transfer in a data-driven optimization era, vol 21. Springer
18. Gupta A, Ong Y-S, Feng L (2015) Multifactorial evolution: toward evolutionary multitasking. IEEE Trans Evol Comput 20(3):343–357
19. Gupta A, Ong Y-S, Feng L (2017) Insights on transfer optimization: because experience is the best teacher. IEEE Trans Emerg Top Comput Intell 2(1):51–64
20. Han L, Zhang Y, Song G, Xie K (2014) Encoding tree sparsity in multi-task learning: a probabilistic framework. In: Twenty-eighth AAAI conference on artificial intelligence
21. He T, Liu Y, Ko T-H, Chan K-C, Ong Y-S (2019) Contextual correlation preserving multiview featured graph clustering. IEEE trans cybern 50(10):4318–4331
22. He T, Bai L, Ong Y-S (2019) Manifold regularized stochastic block model. In: International Conference on Tools with Artificial Intelligence, pp 800–807
23. Hou J-C, Wang S-S, Lai Y-H, Tsao Y, Chang H-W, Wang H-M (2018) Audio-visual speech enhancement using multimodal deep convolutional neural networks. IEEE Trans Emerg Top Comput Intell 2(2):117–128
24. Huang J, Li G, Huang Q, Wu X (2016) Learning label-specific features and class-dependent labels for multi-label classification. IEEE Trans Knowl Data Eng 28(12):3309–3323
25. Huang J, Li G, Huang Q, Wu X (2018) Joint feature selection and classification for multilabel learning. IEEE Trans Cybern 48(3):876–889
26. Huang S-J, Zhou Z-H (2012) Multi-label learning by exploiting label correlations locally. In: Twenty-sixth AAAI conference on artificial intelligence
27. Kato T, Kashima H, Sugiyama M, Asai K (2008) Multi-task learning via conic programming. Adv Neural Inf Process Syst 21:737–744
28. Kendall A, Gal Y, Cipolla R (2018) Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: Proceedings of the IEEE conference on computer vision and pattern recognition 7482–7491

29. LeCun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
30. Lee G, Yang E, Hwang S (2016) Asymmetric multi-task learning based on task relatedness and loss. In: International conference on machine learning, pp 230–238
31. Liu H, Palatucci M, Zhang J (2009) Blockwise coordinate descent procedures for the multi-task lasso, with applications to neural semantic basis discovery. In: Proceedings of the 26th annual international conference on machine learning. ACM, pp 649–656
32. Liu W, Mei T, Zhang Y, Che C, Luo J (2015) Multi-task deep visual-semantic embedding for video thumbnail selection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3707–3715
33. Obozinski G, Taskar B, Jordan M (2006) Multi-task feature selection. Statistics Department, UC Berkeley, Tech. Rep, 2(2.2):2
34. Pan SJ, Yang Q (2009) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10):1345–1359
35. Ramsundar B, Kearnes S, Riley P, Webster D, Konerding D, Pande V (2015) Massively multitask networks for drug discovery. arXiv preprint arXiv:1502.02072
36. Read J, Pfahringer B, Holmes G, Frank E (2011) Classifier chains for multi-label classification. Mach Learn 85(3):333
37. Ruder S (2017). An overview of multi-task learning in deep neural networks. arXiv preprint arXiv:1706.05098
38. Sabour S, Frosst N, Hinton GE (2017) Dynamic routing between capsules. Adv Neural Inf Process Syst 3856–3866
39. Sayed AH (2014) Diffusion adaptation over networks. In: Academic Press library in signal processing, vol 3. Elsevier, pp 323–453
40. Schmidt M, Fung G, Rosales R (2007) Fast optimization methods for l1 regularization: a comparative study and two new approaches. In: European conference on machine learning. Springer, pp 286–297
41. Sener O, Koltun V (2018) Multi-task learning as multi-objective optimization. Adv Neural Inf Process Syst 525–536
42. Tsoumakas G, Katakis I, Vlahavas I (2010) Mining multi-label data. In: Data mining and knowledge discovery handbook. Springer, pp 667–685
43. Tsoumakas G, Katakis I, Vlahavas I (2010) Random k-labelsets for multilabel classification. IEEE Trans Knowl Data Eng 23(7):1079–1089
44. Yang Y, Hospedales TM (2016) Trace norm regularised deep multi-task learning. arXiv preprint arXiv:1606.04038
45. Zhang M-L, Wu L (2014) Lift: multi-label learning with label-specific features. IEEE Trans Pattern Anal Mach Intell 37(1):107–120
46. Zhang M-L, Zhou Z-H (2014) A review on multi-label learning algorithms. IEEE Trans Knowl Data Eng 26(8):1819–1837
47. Zhang W, Li R, Zeng T, Sun Q, Kumar S, Ye J, Ji S (2016) Deep model based transfer and multi-task learning for biological image analysis. IEEE Trans Big Data 6(2):322–333
48. Zhang X, Yang Z, Cao F, Cao J-Z, Wang M, Cai N (2020) Conditioning optimization of extreme learning machine by multitask beetle antennae swarm algorithm. Memet Comput 12(2):151–164
49. Zhang X, Zhuang Y, Wang W, Pedrycz W (2016) Transfer boosting with synthetic instances for class imbalanced object recognition. IEEE Trans Cybern 48(1):357–370
50. Zhang Y, Yang Q (2017) Learning sparse task relations in multi-task learning. In: Proceedings of the thirty-first AAAI conference on artificial intelligence, pp 2914–2920
51. Zhang Y, Yang Q (2017) A survey on multi-task learning. arXiv preprint arXiv:1707.08114
52. Zhang Y, Yeung D-Y (2013) Multilabel relationship learning. ACM Trans Knowl Discov Data (TKDD) 7(2):1–30
53. Zhang Y, Yeung D-Y (2014) A regularization approach to learning task relationships in multitask learning. ACM Trans Knowl Discov Data (TKDD) 8(3):12
54. Zhang Z, Luo P, Loy CC, Tang X (2014) Facial landmark detection by deep multi-task learning. In: European conference on computer vision. Springer, pp 94–108
55. Zhu Y, Kwok JT, Zhou Z-H (2018) Multi-label learning with global and local label correlation. IEEE Trans Knowl Data Eng 30(6):1081–1094