Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

FSPMTL: A New Framework for Self-Paced Multi-Task Learning

LIJIAN SUN[1], YUN ZHOU[2]

[1]Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, China, 410072 (e-mail: sunlijian14@nudt.edu.cn)

[2]Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, China, 410072 (e-mail: zhouyun@nudt.edu.cn)

Corresponding author: Yun Zhou (e-mail: zhouyun@nudt.edu.cn).

This work was supported in part by National Natural Science Foundation of China under Grant 61703416, Natural Science Foundation of Hunan Province under Grant 2018JJ3614 and the Postgraduate Scientific Research Innovation Project of Hunan Province under Grant CX20190041.

ABSTRACT Multi-Task Learning (MTL) is a method to simultaneously utilize commonalities and differences across tasks to improve the learning performances with limited data. However, in most real-world problems, there are many sample noises which might decline the performance of MTL significantly. To address this challenge, Self-Paced Learning (SPL) method is introduced to improve its performance by increasing the numbers of instances gradually from the simplest samples to the most difficult samples. In the current self-paced multi-task learning methods, most of them are introduced as a SPL term in the optimization process, which causes significant limitations in the combination of SPL and MTL. In this paper, we propose a new flexible framework, which combines MTL with SPL and has two stages in the learning process to make it more suitable for learning difficult samples and tasks. With this framework, we are able to take advantages of both of the existing MTL models and SPL models. Further experiments with the synthetic and real-world datasets demonstrate the higher efficiency of our approach when compared with other state-of-the-art algorithms.

INDEX TERMS Multi-task learning, Self-paced learning, A new flexible framework, Multi-task self-paced learning

I. INTRODUCTION

Inspired by human-like reasoning process, MTL can learn multiple related tasks simultaneously rather than separately, and also can utilize the shared representations among the related tasks to fine-tune a generalized model on the original task [1], [2].

In the past decade, numbers of MTL methods have been proposed and then applied in various instances, which could be roughly divided into three major categories [3]–[6]. The first category assumes that all tasks share a common low-rank feature representation [7]–[10]. The second category assumes that different tasks might have shared parameters in the trained model [5], [8], [11], [12]. Although the above two strategies have achieved good results, they ignored the differences of difficulties in the learning process among different tasks and different learning samples.

To address this deficiency, the third category of MTL is developed recently and named as Self-Paced Multi-Task Learning. Specifically, this method adapts a human-like learning mechanism that trains the model from the simplest samples and tasks to the most difficult samples and tasks. Thus, this method achieves improved performances, e.g. SPMTL [13] and *sp*MMTL [14]. Since these models both have a strong coupling between SPL and MTL, it will make the scalability problem of SPL and MTL worse for limited data scenarios.

In this paper, we introduce a flexible framework, which is named *Flexible Self-Paced Multi-Task Learning* (FSPMTL), for self-paced multi-task learning to solve the scalability problem of these methods. Our FSPMTL model contains two stages and can flexibly embed different types of SPL models and MTL models. Specifically, using the SPL mechanism named *the Balanced Self-Paced Learning (BSPL)* [15], our FSPMTL model first selects samples of each task according to the sample difficulties to get sample difficulty levels. Then it uses the state-of-the-art MTL models to iteratively train the samples of different difficulty levels to get the final model.

The main contributions of this paper are summarized as follows:

- To the best our knowledge, this is the first work that presents a common framework to combine SPL with MTL.
- We propose a progressive self-paced multi-task learning mechanism, which is distinct from the conventional ones.
- We make extensive experiments on both synthetic and real datasets to show the effectiveness of our proposed framework.

II. RELATED WORK

As one of the current research hotspots, MTL can effectively improve the overall performance and also increase the robustness of the model by sharing information among the related tasks. Conventional MTL methods assume that the objective function parameters of different tasks should be similar [16], or that multiple related tasks should share the same feature subset [17]. Those early MTL methods have tried to use regular term constraints to minimize the differences between related tasks. However, these methods are prone to negative transfer. Thus, the recent studies on MTL are primarily based on sparse representations [18], [19].

Argyriou et al. [7] came up with a MTL-FEAT model which shared information by learning sparse representations among multiple tasks. Kang et al. [3] relaxed the constraints of the MTL-FEAT model and then presented the DG-MTL model. Based on the MTL-FEAT and DG-MTL models, Kumar et al. [20] proposed the GO-MTL model to selectively share the information across the tasks. Subsequently, based on previous models, Jeong et al. [21] proposed the VSTG-MTL model, performing the variable group structure between variable selections and learning tasks. Compared with the previous MTL models, the VSTG-MTL model greatly improves the performance of model prediction.

As we know, sample qualities might also affect the model performance, which could be considered in the model learning process. Curriculum Learning (CL) mimics the cognitive process of humans and favors a learning algorithm to follow the logical learning sequence from simple examples to more difficult ones [22]. Such "starting small" strategy is very similar to the human's knowledge acquisition process from childhood to adulthood, and also has been demonstrated effectively in multi-modal learning [23], [24] and semi-supervised learning [25]. CL was usually realized under two frameworks: *Self-Paced Learning* (SPL) [26] and *Teaching-to-Learn and Learning-toTeach* (TLLT) [27], [28].

SPL was formally developed in [26], which initiates the training process with simple samples, and then gradually takes more difficult samples into the training. It has been recently shown that SPL is an effective robust learning regime [29], [30] and has achieved rapid development such as SPMoR [31] and C-SPCL [32]. Jiang et al. [33] proved that SPL could avoid falling into local optimum by taking into account both prior knowledge known before training and the learning progress during training. Recently, Ren et al. [15] presented the BSPL model to solve the common imbalanced classification problem in SPL. The BSPL model can select training data proportionally from different category labels, so as to avoid large changes in the category label ratio of sampled data according to different distribution of sample difficulty.

Since 2017, people started to utilize the benefits of both SPL and MTL. Li et al. [13] and Murugesan et al. [14] suggested a method to couple MTL closely with SPL, and achieved relatively good results through simple-to-difficult MTL. In addition, the SPMTL [13] attempts to learn the tasks by simultaneously taking into consideration the complexities of both tasks and instances per task, and the *sp*MMTL [14] embeds task selection into the model learning based on the shared knowledge. In their models, they both optimized the parameters of SPL and MTL at the same time, by learning the difficulty level coefficient and coefficient matrix simultaneously. However, the optimization method leads the models to a low level of scalability and flexibility.

Therefore, we propose a two-stage framework named F-SPMTL to solve this problem. In the first stage, we use the SPL model to obtain the sample difficulty matrix $\boldsymbol{E}$, whose elements show the difficulties of the samples in each task. In the second stage, we select new training samples according to coefficient matrix $\boldsymbol{E}$, so that we could update the optimized MTL model to obtain a coefficient matrix $\boldsymbol{W}$. More details are discussed in the next section.

III. FLEXIBLE SELF-PACED MULTI-TASK LEARNING

A. VARIABLE SELECTION AND TASK GROUPING FOR MULTI-TASK LEARNING

Suppose there exists $\mathcal{T}$ supervised learning tasks, each of which contains D variables and N_t training instances. For the t-th task, it has an input matrix $\boldsymbol{X}_t = \left[\left(\boldsymbol{x}_t^1\right)^T, \cdots, \left(\boldsymbol{x}_t^{N_t}\right)^T\right]^T \in \mathbb{R}^{N_t \times D}$ with $\boldsymbol{x}_t^n \in \mathbb{R}^D$ and an output vector $\boldsymbol{y}_t = \left[y_t^1, \cdots, y_t^{N_t}\right]^T \in \mathbb{R}^{N_t}$. Later on, we can use a linear model to describe the relationship between inputs and outputs,

$$y_t^n = f\left(\boldsymbol{w}_t^T \boldsymbol{x}_t^n\right) \tag{1}$$

where f is a logit function for the binary classification problem $y_t^n \in \{-1, 1\}$ and $\boldsymbol{w}_t^T \in \mathbb{R}^D$ represents a coefficient vector for the t-th task. Then, the coefficient vector $\boldsymbol{w}_t$ of $\mathcal{T}$ tasks generates a coefficient matrix $\boldsymbol{W} = [\boldsymbol{w}_1, \cdots, \boldsymbol{w}_\mathcal{T}]$.

There is such a low-dimensional latent space that the coefficient matrix $\boldsymbol{W}$ can be represented on low rank factorization and sparse space. We denote $\boldsymbol{W}$ as the product of two low rank matrices $\boldsymbol{U}$ and $\boldsymbol{V}$, i.e. $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{V}$. where $\boldsymbol{U} \in \mathbb{R}^{D \times M}$ is the variable-latent matrix, $\boldsymbol{V} \in \mathbb{R}^{M \times \mathcal{T}}$ is the latent-task matrix, and $M \ll \min(D, \mathcal{T})$, M is the number of latent basis. For the t-th task, $\boldsymbol{w}_t = \boldsymbol{U}\boldsymbol{v}_t$, where the t-th column vector $\boldsymbol{v}_t$ of $\boldsymbol{V}$ is weighting vector for the t-th task.

The optimization function for this problem is,

$$\min_{\boldsymbol{U},\boldsymbol{V}} \sum_{t=1}^{\mathcal{T}} \frac{1}{N_t} L\left(y_t, \boldsymbol{X}_t \boldsymbol{U} \boldsymbol{v}_t\right) + \gamma_1 \|\boldsymbol{U}\|_1 + \gamma_2 \|\boldsymbol{U}\|_{1,\infty} + \mu \sum_{t=1}^{\mathcal{T}} \left(\|\boldsymbol{v}_t\|_k^{sp}\right)^2 \tag{2}$$

where $L(\cdot,\cdot)$ is the empirical loss function, which is the logistic loss $\sum_{n=1}^{N_t} \log\left(1+\exp\left(-y_t^n \boldsymbol{v}_t^T \boldsymbol{U}^T \boldsymbol{x}_t^n\right)\right)$ for a binary classification problem; $\|\boldsymbol{U}\|_1 = \sum_{d=1}^{D}\sum_{m=1}^{M}|\boldsymbol{u}_{dm}|$ is the ℓ_1 norm; $\|\boldsymbol{U}\|_{1,\infty} = \sum_{d=1}^{D}\left\|\boldsymbol{u}^d\right\|_\infty$ is the $\ell_{1,\infty}$ norm; $\|v_t\|_k^{sp}$ is the k-support norm; where γ_1, γ_2, and μ are the regularization parameters.

B. BALANCED SELF-PACED LEARNING

In this section, we will use the BSPL model for a single task to obtain the sample difficulty matrix $\boldsymbol{E}$ of the training samples. Assume that the training data can be divided into L levels based on the difficulty of the data samples, $\boldsymbol{E}$ is a three-dimensional matrix of $\mathbb{R}^{L\times\mathcal{T}\times N_t}$, whose row vector is denoted as $\boldsymbol{E} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_L]^T$ and whose column vector is denoted as $\boldsymbol{E} = \left[\boldsymbol{e}^1, \boldsymbol{e}^2, \cdots, \boldsymbol{e}^{\mathcal{T}}\right]$. Each element $\boldsymbol{e}_l^t$ in $\boldsymbol{E}$ can be represented as a vector of $\mathbb{R}^{N_t}$ and its value range is the discrete set $\{0,1\}$.

In terms of the t-th single task, the goal of the BSPL model is to jointly learn the model parameter θ_l^t which is the parameter of the decision function g and the latent sample difficulty variable $\boldsymbol{e}_l^t = \left[e_l^{t,1}, \cdots, e_l^{t,N_t}\right]$ by minimizing:

$$\min_{\theta_l^t, \boldsymbol{e}_l^t} \sum_{n=1}^{N_t} e_l^{t,n} L\left(y_t^n; g\left(\boldsymbol{x}_t^n, \theta_l^t\right)\right) + \mu R\left(\theta_l^t\right) - \sum_{k=1}^{K}\sum_{\substack{nk \\ \boldsymbol{x}_t^{nk}\in Cl_k}} \lambda_k e_l^{t,nk} \tag{3}$$

where $e_l^{t,n} \in \{0,1\}$, K is the number of classes, $R\left(\theta_l^t\right)$ is the regularization term that can be expressed as $R\left(\theta_l^t\right) = \sum_{n=1}^{N_t} G_\xi\left[A\left(\left(\theta_l^t\right)^T \tilde{\boldsymbol{x}}_t^n\right)\right] - A\left(\left(\theta_l^t\right)^T \boldsymbol{x}_t^n\right)$, where $\tilde{\boldsymbol{x}}_t^n$ is the noise feature and $G_\xi[\cdot]$ is the expectation according to a certain distribution. In our method, we add Gaussian noise to $R\left(\theta_l^t\right)$. The function $A(\cdot)$ depends on the specific loss function. μ denotes the corresponding coefficient, and Cl_k means the k-th class. With a fixed θ_l^t, the global optimum $\boldsymbol{e}_l^{t*} = \left[e_l^{t,1}, \cdots, e_l^{t,N_t}\right]$ can be calculated by the following rule,

$$e_l^{t,n*} = \begin{cases} 1, & if \quad L_k\left(y_t^n; g\left(\boldsymbol{x}_t^n, \theta_l^t\right)\right) < \lambda_k \\ 0, & otherwise \end{cases} \tag{4}$$

where $L_k\left(y_t^n; g\left(\boldsymbol{x}_t^n, \theta_l^t\right)\right)$ represents the loss of instances in the k-th class.

C. UPDATING PROCESS

During the training process of the l-th level, the l-th row vector $\boldsymbol{e}_l$ of $\boldsymbol{E}$ represents the sample difficulty vector of the training samples. Therefore, we can train the model parameters $(\boldsymbol{E}, \boldsymbol{W})$ with the following strategies.

The update process of parameters in the algorithm is generally completed in two stages. In the first stage, by solving the equation 3, we can obtain the sample difficulty coefficient $\boldsymbol{e}_l^t$ in the l level of the t-th task by the BSPL model. Specific calculations are shown in Step 1 and 2. In the second stage, using the training samples which are selected by $\boldsymbol{e}_l$, the VSTG-MTL model is used to train $\boldsymbol{W}_l$ by solving the equation 2. Iteratively, when training processes of all L levels are completed, the final training coefficient matrix $\boldsymbol{W}$ is obtained by calculating the expectation of $\boldsymbol{W}_1$ to $\boldsymbol{W}_L$. Specific calculations are shown in Step 3 and 4 as follows.

Specifically, the algorithm updating process can be divided into four steps. When L, θ_l, $\boldsymbol{U}_l^{init}$ and $\boldsymbol{V}_l^{init}$ are initialized, the following four steps will be iteratively completed.

Step 1: Fix θ_l^t, update $\boldsymbol{e}_l^t$.

First, for the t-th task, we fix θ_l^t and then solve the following problem to update $\boldsymbol{e}_l^t$:

$$\begin{aligned} \boldsymbol{e}_l^{t*} &= \arg\min \sum_{n=1}^{N_t} e_l^{t,n} L\left(y_t^n; g\left(\boldsymbol{x}_t^n, \theta_l^t\right)\right) - \sum_{k=1}^{K}\sum_{\substack{nk \\ \boldsymbol{x}_t^{nk}\in Cl_k}} \lambda_k e_l^{nk} \\ &= \arg\min \sum_{k=1}^{K}\sum_{\substack{nk \\ \boldsymbol{x}_t^{nk}\in Cl_k}} \boldsymbol{e}_l^{t,nk}\left(L\left(y_t^{nk}; g\left(\boldsymbol{x}_t^{nk}, \theta_l^{t,nk}\right)\right) - \lambda_k\right) \end{aligned} \tag{5}$$

Step 2: Fix $\boldsymbol{e}_l^t$, update θ_{l+1}^t.

For the t-th task, we fix $\boldsymbol{e}_l^t$ and update θ_{l+1}^t by solving:

$$\theta_{l+1}^{t*} = \arg\min\left(\sum_{n=1}^{N_t} e_l^{t,n} L\left(y_t^n; g\left(\boldsymbol{x}_t^n, \theta_l^t\right)\right) + \mu R\left(\theta_l^t\right)\right) \tag{6}$$

We use the gradient descent algorithm to update the equation above, then increase λ_k and return to the Step 1 to be iteratively executed until all the instances are selected. Then, we can get the sample difficulty coefficient vector $\boldsymbol{e}_l^t$ of the t-th task.

When all tasks are executed, we can obtain $\boldsymbol{e}_l = \left[\boldsymbol{e}_l^1, \boldsymbol{e}_l^2, \cdots, \boldsymbol{e}_l^{\mathcal{T}}\right]$ and $\boldsymbol{E} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_L]^T$.

Step 3: Fix $\boldsymbol{e}_l$, update $\boldsymbol{U}_l$.

For each level of training tasks, we select the training sample through $\boldsymbol{e}_l$, i.e.

$$\boldsymbol{X}_l = \boldsymbol{X}\left(\boldsymbol{e}_l \; is \; equal \; to \; 1\right) \tag{7}$$

Similarly, we select the label $\boldsymbol{y}_l$ corresponding to $\boldsymbol{X}_l$. Then, we update $\boldsymbol{U}_l$ with an alternating direction method of multipliers and an early stopping. The objective function is as follows:

$$\boldsymbol{U}_l = \arg\min_{\boldsymbol{U}_l}\left(\sum_{t=1}^{T} \frac{1}{N_j} L\left(\boldsymbol{y}_l^t, \boldsymbol{X}_l^t \boldsymbol{U}_l \boldsymbol{v}_l^t\right) + \gamma_1 \|\boldsymbol{U}_l\|_1 + \gamma_2 \|\boldsymbol{U}_l\|_{1,\infty}\right) \tag{8}$$

Step 4: Fix $\boldsymbol{U}_l$, update $\boldsymbol{V}_l$.

We use accelerated proximal gradient descent to solve the following equation and $\boldsymbol{V}_l = \left[\boldsymbol{v}_l^1, \cdots, \boldsymbol{v}_l^{\mathcal{T}}\right]$:

$$\boldsymbol{v}_l^t = \arg\min_{\boldsymbol{v}_l^t} \left(\sum_{t=1}^{\mathcal{T}} \frac{1}{N_j} L\left(\boldsymbol{y}_l^t, \boldsymbol{X}_l^t \boldsymbol{U}_l \boldsymbol{v}_l^t\right) + \mu \sum_{t=1}^{\mathcal{T}} \left(\left\|\boldsymbol{v}_l^t\right\|_k^{sp}\right)^2 \right) \tag{9}$$

Next, the algorithm iteratively executes the step 3 and 4 until the $\boldsymbol{U}_l$ and $\boldsymbol{V}_l$ coverage. After that, we calculate $\boldsymbol{W}_l = \boldsymbol{U}_l \boldsymbol{V}_l$.

Finally, it will repeat L times until the samples of all L levels are trained and then the expectation of $\boldsymbol{W}_1$ to $\boldsymbol{W}_L$ could be seemed as the final calculation result $\boldsymbol{W}$.

D. FLEXIBLE FRAMEWORK FOR SELF-PACED MULTI-TASK LEARNING

To sum up, we can extract a general flexible framework for self-paced multi-task learning, which is named the FSPMTL algorithm shown in the following Algorithm 1.

TABLE 1: The Flexible Self-Paced Multi-Task Learning (FSPMTL) Algorithm

Algorithm 1 Flexible Self-Paced Multi-Task Learning (FSPMTL)

Input: Dataset $\boldsymbol{X}$ and $\boldsymbol{y}$
The number of data difficulty levels L
Output: Coefficient matrix $\boldsymbol{W}$

```
for t=1:T do
    for l=1:L do
        select SPL model to train to get e_l^t
    end for
end for
e_l = [e_l^1, e_l^2, ..., e_l^T]
E = [e_1, e_2, ..., e_L]^T
for l=1:L do
    select training samples X_l and y_l from X and y according to e_l
    select MTL model for training based on X_l and y_l to obtain W_l
end for
W = E[W_1, ..., W_L]
return W.
```

Let O_e be the time spent for solving the equation 5 and 6 once, and O_W be the time spent for solving the equation 8 and 9 once. Thus, the time spent for the *SPL* part is $O_e * L * \mathcal{T}$ and the time spent for the *MTL* part is $O_W * L$. From the Algorithm 1, we can see that the model based on the FSPMTL algorithm needs $O_e * L * \mathcal{T} + O_W * L$ time for each run. Moreover, as shown in Algorithm 1, the convergence of the FSPMTL algorithm is depend on the convergence of the *SPL* and *MTL* parts, which means the FSPMTL algorithm will stop after the *SPL* model selects all the instances and the *MTL* model reaches two residuals' thresholds [21].

IV. EXPERIMENT

In this section, we aim to verify the effectiveness of the FSPMTL algorithm under different experimental settings. The Matlab implementation of our method is available at the URL: http://yzhou.github.io/#Code.

A. EXPERIMENT SETTINGS

In order to prove the validity of our framework, the FSPMTL algorithm is implemented based on the VSTG-MTL model and named as FSP-VSTG-MTL. Here, we compared our FSP-VSTG-MTL with the following methods:

- **BSPL-STL method**: it is a single-task learning method based on balanced self-paced learning with Gaussian noises [15].
- **VSTG-MTL method**: it decomposes the weight matrix in the model into the product of two low rank matrices. These matrices would simultaneously perform feature selection among tasks and overlapping group structures among learning tasks [21].
- ***sp*MMTL method**: *sp*MMTL is the acronym of Self-Paced Mean Regularized Multi-task Learning, and the model picks up the simple tasks based on the distance of each task's difficulty [14].

The parameter initialization of the FSP-VSTG-MTL is divided into two parts. For the BSPL part, we selected half of the data points during the first iteration and then updated the λ_k with rise of 10% during the next iteration, that is, L=6. Then we initially set $e_l^{t,n} = 1(n = 1, \cdots, N_t)$ and ran the corresponding classification algorithm for 5 iterations to obtain an estimate of θ_l. For the VSTG-MTL part, the number of latent bases M is selected from the search grid $\{1, 3, 5, 7\}$. we set the third regularization parameter μ to be equal to the first regularization parameter γ_1. The regularization parameters are selected from the search grid $\left\{2^{-10}, \cdots, 2^3\right\}$. Initial estimates of the matrix $\boldsymbol{W}_l^{init}$ is implemented by logistic regression algorithm. The initial estimates of $\boldsymbol{U}_l^{init}$ and $\boldsymbol{V}_l^{init}$ are given by singular value decomposition of $\boldsymbol{W}_l^{init}$.

For experimental datasets, we first randomly selected the data in the datasets with a ratio of 9:1 to obtain the training set and testing set. In the training set, we used the five-fold cross-validation method to get the model outputs. Then, we made predictions on the testing set to get its final classification effect. We ensure that the datasets used in each training and testing process are consistent across different models. We repeated each case 10 times and reported the average results.

B. SYNTHETIC DATASETS

We generated four synthetic datasets as follows, which have different number of *D* dimensional variables and $\mathcal{T}$ tasks. The instance x_t^n is sampled from a Standard Normal Distribution *N*(0,1), and the response is $y_t^n = sign(\boldsymbol{w}_t^T x_t^n + \xi_t^n)$. To create difficult instances, we added different noises to instances by setting $\xi_t^n = \sigma_t^n \theta_t^n$, where σ_t^n is drawn i.i.d. from a Normal Distribution *N*(0,5), and θ_t^n is drawn i.i.d. from *N*(0, 1). A true coefficient matrix $\boldsymbol{W}^* = [\boldsymbol{w}_1^*, \cdots, \boldsymbol{w}_{\mathcal{T}}^*]$ has a low-rank structure $M = rank(\boldsymbol{W}) = 5$ and is estimated by $\boldsymbol{UV}$, where $\boldsymbol{U} \in R^{D \times M}$ and $\boldsymbol{V} \in R^{M \times \mathcal{T}}$. Each synthetic dataset differs on the structure of the two matrices $\boldsymbol{U}$ and $\boldsymbol{V}$.

Syn1: Syn1 has 25 dimensional variables and 20 tasks. For $r = 1, \cdots, M$, the latent basis $\boldsymbol{u}_r$ only has non-zero

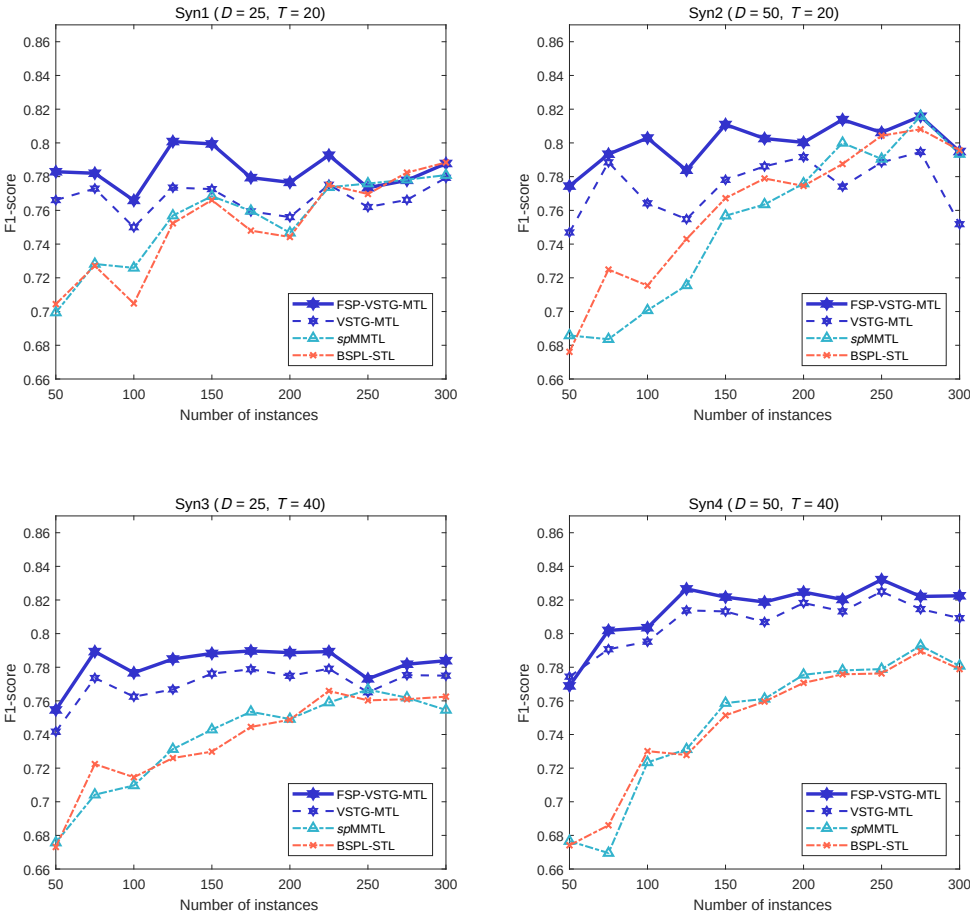

FIGURE 1: Results on four synthetic datasets with different number of instances, the dark blue solid line represents our proposed model.

values from the $(3r-2)$-th to the $(3r+3)$-th components. The nonzero values are generated by the Normal Distribution $N(1,0.25)$. Similarly, for $r = 1, \cdots, M$, the weighting vectors $\boldsymbol{v}_{4r-3}, \cdots, \boldsymbol{v}_{4r}$ only have nonzero values on the r-th and $(r+1)$-th components. The last four weighting vectors $\boldsymbol{v}_{4M-3}, \cdots, \boldsymbol{v}_{4M}$ only have the nonzero values on the $(M-1)$-th and M-th components. The nonzero values are generated through a Uniform Distribution from 1 to 1.5.

Syn2: Syn2 has 50 dimensional variables and 20 tasks. For $r = 1, \cdots, M$, the latent basis $\boldsymbol{u}_r$ only has non-zero values from the $(8r-7)$-th to the $(8r+8)$-th components. Similarly, $\boldsymbol{V}$ is generated in the same way as Syn1. The nonzero values are generated by the same distribution as that used in Syn1.

Syn3: Syn3 has 25 dimensional variables and 40 tasks. $\boldsymbol{U}$ is generated in the same way as Syn1. Similarly, for $r = 1, \cdots, M$, the weighting vectors $\boldsymbol{v}_{8r-7}, \cdots, \boldsymbol{v}_{8r}$ only have nonzero values on the r-th and $(r+1)$-th components. The last four weighting vectors $\boldsymbol{v}_{8M-7}, \cdots, \boldsymbol{v}_{8M}$ only have the nonzero values on the $(M-1)$-th and M-th components. The nonzero values are generated by the same distribution as in Syn1.

Syn4: Syn4 has 50 dimensional variables and 40 tasks. Similarly, $\boldsymbol{U}$ is generated in the same way as Syn2, $\boldsymbol{V}$ is generated in the same way as Syn3. The nonzero values are generated by the same distribution as in Syn1.

Varying Number of Instances: To verify the effect of numbers of instances on the learning performance of MTL models, we varied the number of instances in parameter learning. For each task, we increased the total number of instances of each dataset from 50 to 300, by adding 25 each time. Each experiment will be repeated for 10 times, and the results are reported with the mean of the F1-score. Figure 1 summarizes the experimental results on four synthetic datasets above. As we can see, for single-task leaning (BSPL-STL), the growth of instance number would improve learning performance significantly. However, for MTL methods, the performance not only relies on the number of task instances, but also the number of dimensions. In addition, compared to the state-of-the-art model (VSTG-MTL, *sp*MMTL) , our proposed FSP-VSTG-MTL is better off in most settings. It's worth noting that the performance of our model is 2.20%, 3.28%, 1.55% and 1.00% higher than that of the VSTG-MTL model on four synthetic datasets, respectively. Besides, to verify the flexibility of our proposed framework, we extended *sp*MMTL with our flexible learning framework to see if it could be further improved.

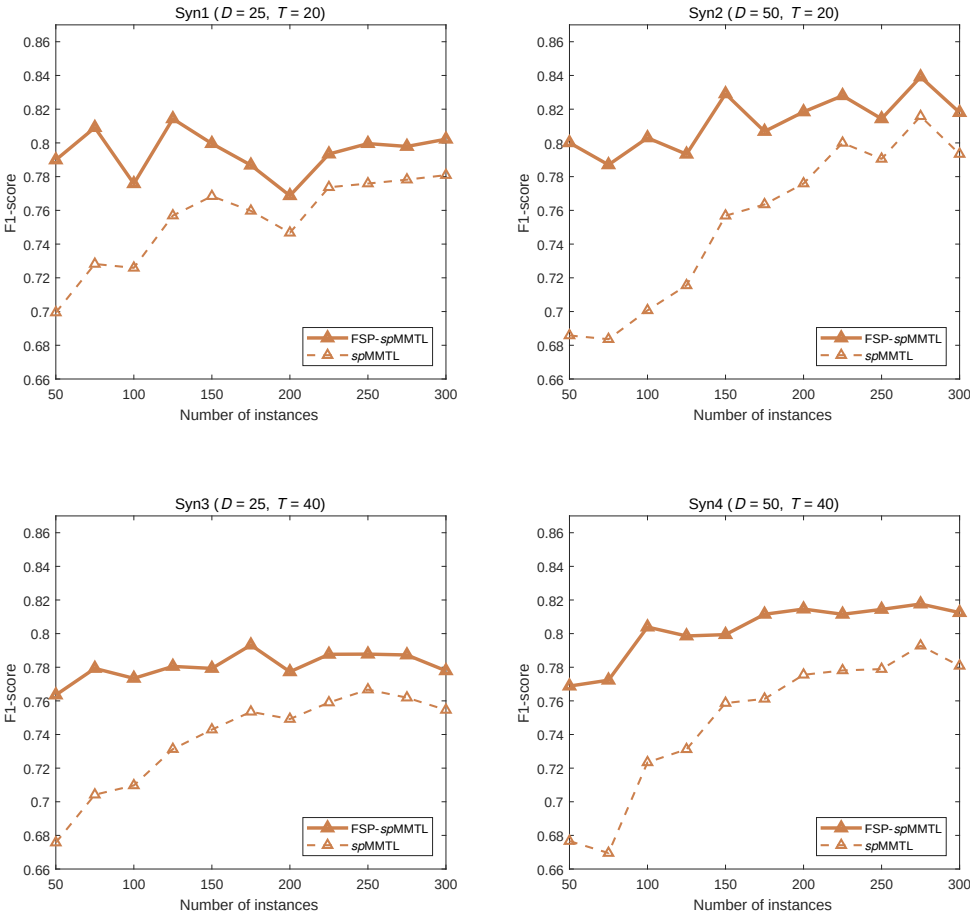

FIGURE 2: Results on four synthetic datasets with different number of instances, the solid lines represent the extended model FSP-*sp*MMTL.

Flexibility Test of the Proposed Framework: To further verify the flexibility of our framework, we extended the framework into the *sp*MMTL model and named it FSP-*sp*MMTL model. The parameter initializations of these models are set with reference to [14]. As can be seen in Figure 2, the solid line always goes higher than its corresponding dotted line. Specifically, the FSP-*sp*MMTL model is better than the *sp*MMTL model by 5.35%, 7.92%, 5.90% and 7.27% on four synthetic datasets, respectively, which demonstrates that our flexible learning framework has achieved excellent results. Therefore, adding the FSPMTL framework during training process can effectively improve the performance of the model.

Robustness Test of The Proposed Framework: If we change the level of noise in the datasets, the learning difficulty of models will change as well. In order to do this, the Normal Distribution of σ_t^n would be varied from $N(0,1)$ to $N(0,10)$, increasing the variance by one each time. Meanwhile, we set 100 instances for each task. Figure 3 shows that each model performances are becoming worse as noises increase. When the noise stays at a low level, the FSPMTL framework has little impact on model learning. When the variance of σ_t^n is 1-5, on average, FSP-VSTG-MTL only performs better than VSTG-MTL by 1.03%, 2.54%, 1.28% and -0.05% and FSP-*sp*MMTL performs better than *sp*MMTL by 4.76%, 10.14%, 4.54% and 10.64% on four synthetic datasets above. However, with the rise of the noise, the advantages of FSPMTL framework are emerging. Specifically, when the variance of σ_t^n is 6-10, averagely, FSP-VSTG-MTL performs better than VSTG-MTL by 4.15%, 6.44%, 3.26% and 2.67% and FSP-*sp*MMTL performs better than *sp*MMTL by 9.42%, 13.91%, 10.87% and 15.97%. The results demonstrate the superiority and flexibility of the FSPMTL framework.

Visualization of The Selected Samples: In order to make our experiment clearer, we took the first task as an example to visualize the selected samples with our FSP-VSTG-MTL algorithm. When the task contains 100 instances, we first selected 50% of samples when $l = 1$, and then increased each level by 10% of samples. For the selected samples, we used the Principal Component Analysis (PCA) to project original data into a lower-dimensional sub-space, and visualized the first two dimensions. As shown in Figure 4, when $l = 1$, there is a clear boundary between positive samples distribution and negative samples distribution. With the increase the l, the number of the selected samples increases gradually and the overlap area of samples distributions in in two dimensions

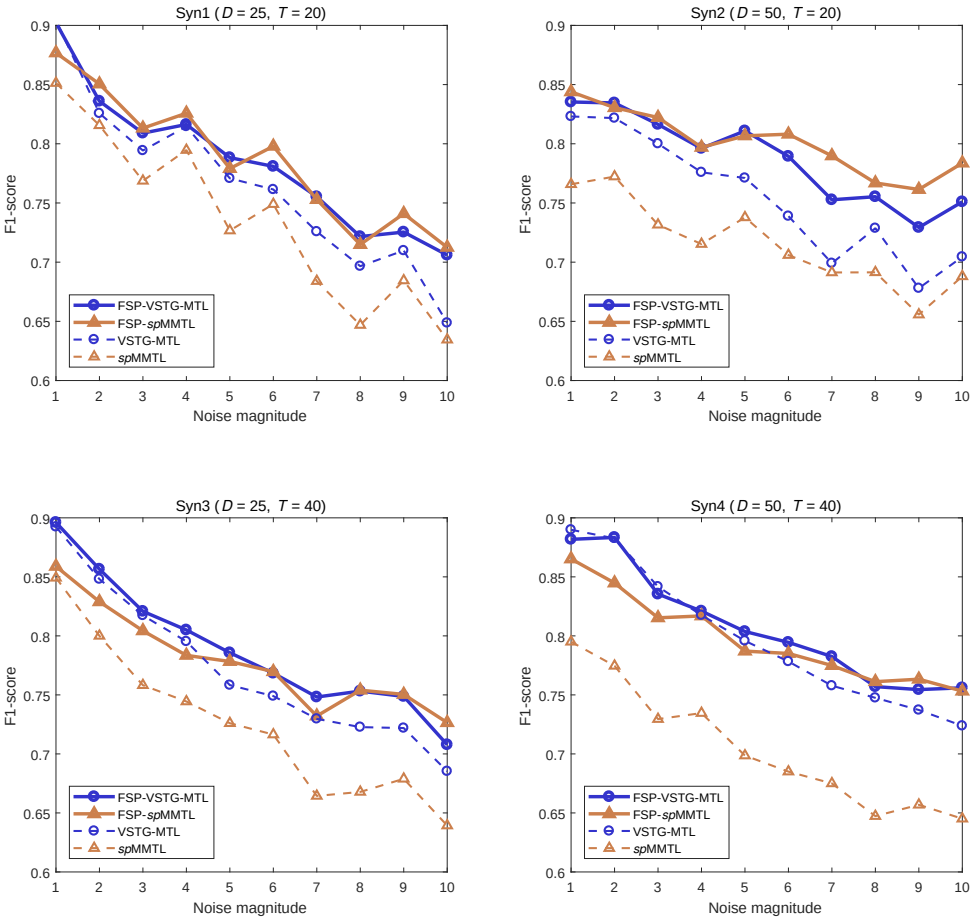

FIGURE 3: Results on four synthetic datasets with 10 different sample noises, two solid lines represent the extended models FSP-VSTG-MTL and FSP-*sp*MMTL.

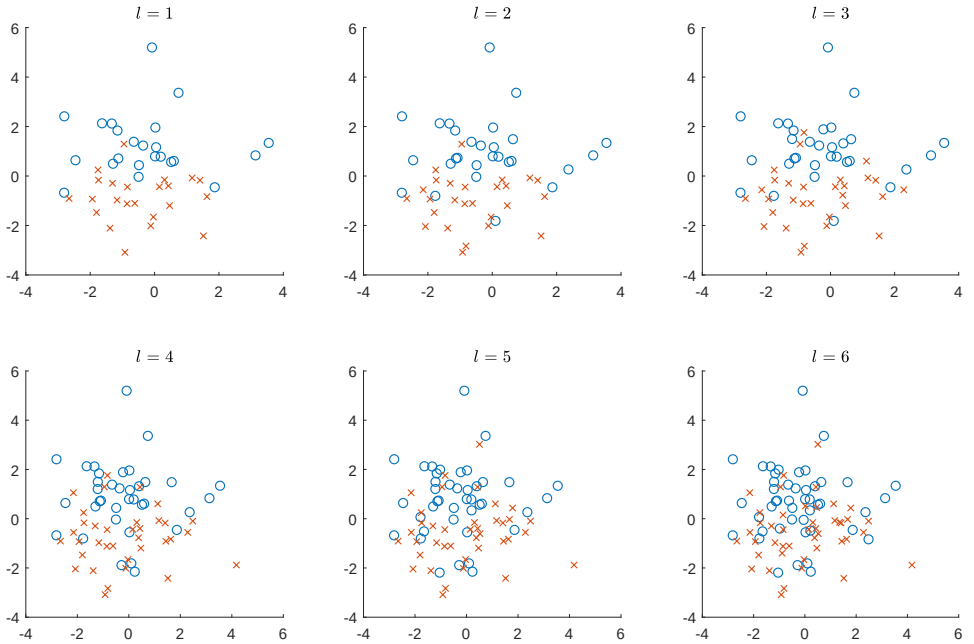

FIGURE 4: 2-D distribution of the selected samples in different levels with our FSP-VSTG-MTL algorithm in different sample difficulty levels. Here "o" represents positive samples and "x" represents negative samples in the classification problem.

also increases gradually, which means the selected samples are increasingly difficult to distinguish. This is the reason why our FSPMTL algorithm could get increased classification accuracy and robustness.

C. REAL-WORLD DATASET

London School Data (*school*) In order to further verify the feasibility of our method, we conducted experiments by using the classification dataset which is generated from the dataset of classic *school* dataset. The *school* dataset is a regression dataset obtained internally by the London Education Authority, including test scores of 15,362 students in 139 secondary schools in London during three years from 1985 to 1987. The dataset contains 139 tasks and 15,362 observations corresponding to different schools and their student's test. Each observation is contained by 3 continuous variables and 23 binary variables, representing the professional attributes of the school and students. In this experiment, the *school* dataset is discretized. There are 6984 positive samples whose score are higher than 20, accounting for 45.46% of the sample size and 8,378 negative samples, lower than or equal to 20, accounting for 54.54%. The ratio of positive and negative samples is close to 1:1.

TABLE 2: The F1-score of different methods on the real *school* dataset. The statistically best models are highlighted in bold.

Methods	BSPL-STL	VSTG-MTL	FSP-VSTG-MTL	*sp*MMTL	FSP-*sp*MMTL
F1-score	0.6269±0.0133	0.7234±0.0097	**0.7272±0.0103**	0.7138±0.0104	0.7234±0.0108

Table 2 shows the results of our model on the real dataset. It can be seen that FSP-VSTG-MTL is superior to all other models in *school* dataset, thus confirming the effectiveness of our proposed FSPMTL framework. Specifically, BSPL-STL is one of the latest single-task learning models, yet it is weak in handling *school* dataset. All the MTL methods proposed in the past two years have achieved better classification prediction results than single-task learning model. In addition, FSP-VSTG-MTL and FSP-*sp*MMTL proposed by this paper are superior to the latest VSTG-MTL and *sp*MMTL models, which are 0.53% and 1.39% respectively. To sum up, by incorporating the self-paced learning regime into MTL, our method is effective in these experiments.

V. CONCLUSION

In this paper, we propose the *Flexible Self-Paced Multi-Task Learning* framework with a loosely coupled approach to combine the MTL model with the SPL model. In this way, it can be flexibly embedded different MTL models into SPL models. Extensive experiments show that our new framework not only effectively improves the performance of the traditional model, but also increase the flexibility and robustness of the model. For future work, we would like to introduce the prior knowledge in the framework and apply this method in real-world applications.

REFERENCES

[1] R. Caruana, "Multitask learning," Machine Learning, vol. 28, no. 1, pp. 41–75, 1997. [Online]. Available: https://doi.org/10.1023/A:1007379606734
[2] Y. Zhang and Q. Yang, "A survey on multi-task learning," CoRR, vol. abs/1707.08114, 2017. [Online]. Available: http://arxiv.org/abs/1707.08114
[3] Z. Kang, K. Grauman, and F. Sha, "Learning with whom to share in multi-task feature learning," in Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011, 2011, pp. 521–528. [Online]. Available: https://icml.cc/2011/papers/344_icmlpaper.pdf
[4] J. Pu, Y. Jiang, J. Wang, and X. Xue, "Multiple task learning using iteratively reweighted least square," in IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013, 2013, pp. 1607–1613. [Online]. Available: http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6857
[5] J. Pu, J. Wang, Y. Jiang, and X. Xue, "Multiple task learning with flexible structure regularization," Neurocomputing, vol. 177, pp. 242–256, 2016. [Online]. Available: https://doi.org/10.1016/j.neucom.2015.11.029
[6] Z. Shi, P. Jian, Y. G. Jiang, F. Rui, and X. Xue, "Flexible multi-task learning with latent task grouping," Neurocomputing, vol. 189, no. C, pp. 179–188, 2016.
[7] A. Argyriou, T. Evgeniou, and M. Pontil, "Convex multi-task feature learning," Machine Learning, vol. 73, no. 3, pp. 243–272, 2008.
[8] Y. Zhang and D. Yeung, "Transfer metric learning by learning task relationships," in Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010, 2010, pp. 1199–1208. [Online]. Available: https://doi.org/10.1145/1835804.1835954
[9] Y. Yang, J. Yang, J. Yan, S. Liao, D. Yi, and S. Z. Li, "Salient color names for person re-identification," in Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I, 2014, pp. 536–551. [Online]. Available: https://doi.org/10.1007/978-3-319-10590-1_35
[10] S. Kim and E. P. Xing, "Tree-guided group lasso for multi-task regression with structured sparsity," 2009.
[11] A. Schwaighofer, V. Tresp, and K. Yu, "Learning gaussian process kernels via hierarchical bayes," in Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada], 2004, pp. 1209–1216. [Online]. Available: http://papers.nips.cc/paper/2595-learning-gaussian-process-kernels-via-hierarchical-bayes
[12] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," Journal of Machine Learning Research, vol. 6, no. Nov, pp. 1817–1853, 2005.
[13] C. Li, J. Yan, F. Wei, W. Dong, Q. Liu, and H. Zha, "Self-paced multi-task learning," in Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA., 2017, pp. 2175–2181. [Online]. Available: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14535
[14] K. Murugesan and J. G. Carbonell, "Self-paced multitask learning with shared knowledge," in Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, 2017, pp. 2522–2528. [Online]. Available: https://doi.org/10.24963/ijcai.2017/351
[15] Y. Ren, P. Zhao, Z. Xu, and D. Yao, "Balanced self-paced learning with feature corruption," in 2017 International Joint Conference on Neural Networks, IJCNN 2017, Anchorage, AK, USA, May 14-19, 2017, 2017, pp. 2064–2071. [Online]. Available: https://doi.org/10.1109/IJCNN.2017.7966104
[16] T. Evgeniou and M. Pontil, "Regularized multi–task learning," in Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2004, pp. 109–117.
[17] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in Advances in neural information processing systems, 2007, pp. 41–48.
[18] Y. Zhou, J. Wang, C. Zhu, and W. Zhang, "Multiple dags learning with non-negative matrix factorization," in Proceedings of the 3rd Workshop on Advanced Methodologies for Bayesian Networks, AMBN 2017, Kyoto, Japan, September 20-22, 2017, 2017, pp. 81–92. [Online]. Available: http://proceedings.mlr.press/v73/zhou17a.html
[19] D. Zhang, J. Han, L. Yang, and D. Xu, "SPFTN: A joint learning framework for localizing and segmenting objects in weakly labeled videos," IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 2, pp.

475–489, 2020. [Online]. Available: https://doi.org/10.1109/TPAMI.2018.2881114

[20] A. Kumar and H. D. Iii, "Learning task grouping and overlap in multi-task learning," Computer Science, vol. 2, 2012.

[21] J. Jeong and C. Jun, "Variable selection and task grouping for multi-task learning," in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018, 2018, pp. 1589–1598. [Online]. Available: https://doi.org/10.1145/3219819.3219992

[22] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009, 2009, pp. 41–48. [Online]. Available: https://doi.org/10.1145/1553374.1553380

[23] C. Gong, "Exploring commonality and individuality for multi-modal curriculum learning," in Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA, 2017, pp. 1926–1933. [Online]. Available: http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14205

[24] C. Gong, J. Yang, and D. Tao, "Multi-modal curriculum learning over graphs," ACM Trans. Intell. Syst. Technol., vol. 10, no. 4, pp. 35:1–35:25, 2019. [Online]. Available: https://doi.org/10.1145/3322122

[25] C. Gong, D. Tao, S. J. Maybank, W. Liu, G. Kang, and J. Yang, "Multi-modal curriculum learning for semi-supervised image classification," IEEE Trans. Image Process., vol. 25, no. 7, pp. 3249–3260, 2016. [Online]. Available: https://doi.org/10.1109/TIP.2016.2563981

[26] M. P. Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," in Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems 2010. Proceedings of a meeting held 6-9 December 2010, Vancouver, British Columbia, Canada., 2010, pp. 1189–1197. [Online]. Available: http://papers.nips.cc/paper/3923-self-paced-learning-for-latent-variable-models

[27] C. Gong, D. Tao, J. Yang, and W. Liu, "Teaching-to-learn and learning-to-teach for multi-label propagation," in Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA, 2016, pp. 1610–1616. [Online]. Available: http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11780

[28] C. Gong, D. Tao, W. Liu, L. Liu, and J. Yang, "Label propagation via teaching-to-learn and learning-to-teach," IEEE Trans. Neural Networks Learn. Syst., vol. 28, no. 6, pp. 1452–1465, 2017. [Online]. Available: https://doi.org/10.1109/TNNLS.2016.2514360

[29] K. Ghasedi, X. Wang, C. Deng, and H. Huang, "Balanced self-paced learning for generative adversarial clustering network," in IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, 2019, pp. 4391–4400. [Online]. Available: http://openaccess.thecvf.com/content_CVPR_2019/html/Ghasedi_Balanced_Self-Paced_Learning_for_Generative_Adversarial_Clustering_Network_CVPR_2019_paper.html

[30] K. Wang, Y. Wang, Q. Zhao, D. Meng, and Z. Xu, "Splboost: An improved robust boosting algorithm based on self-paced learning," CoRR, vol. abs/1706.06341, 2017. [Online]. Available: http://arxiv.org/abs/1706.06341

[31] L. Han, D. Zhang, D. Huang, X. Chang, J. Ren, S. Luo, and J. Han, "Self-paced mixture of regressions," in Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017, 2017, pp. 1816–1822. [Online]. Available: https://doi.org/10.24963/ijcai.2017/252

[32] D. Zhang, J. Han, L. Zhao, and D. Meng, "Leveraging prior-knowledge for weakly supervised object detection under a collaborative self-paced curriculum learning framework," Int. J. Comput. Vis., vol. 127, no. 4, pp. 363–380, 2019. [Online]. Available: https://doi.org/10.1007/s11263-018-1112-4

[33] L. Jiang, D. Meng, Q. Zhao, S. Shan, and A. G. Hauptmann, "Self-paced curriculum learning," in Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA., 2015, pp. 2694–2700. [Online]. Available: http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9750

LIJIAN SUN received the bachelor's degree from National University of Defense Technology (NUDT) of China, in 2018. He is currently pursuing the Master degree in Science and Technology on Information Systems Engineering Laboratory at the National University of Defense Technology. His research interests include self-paced learning and multi-task learning.

YUN ZHOU received his Ph.D. degree in computer science from the Queen Mary, University of London in 2015. He is currently an Assistant Professor in Science and Technology on Information Systems Engineering Laboratory at the National University of Defense Technology, Changsha, China. His research focuses on Machine Learning and Decision Support Systems. He applies these techniques to a wide range of real-world problems. He has published several papers in reputed journals and conferences in this area, including INFOCOM, IJCAI, IJAR, UAI, and PGM.