

Received May 29, 2020, accepted July 12, 2020, date of publication July 17, 2020, date of current version July 29, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3009988

# FSPMTL: Flexible Self-Paced Multi-Task Learning

LIJIAN SUN<sup>ID</sup> AND YUN ZHOU<sup>ID</sup>, (Member, IEEE)

Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha 410072, China

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

This work was supported in part by the National Natural Science Foundation of China under Grant 61703416, in part by the Natural Science Foundation of Hunan Province under Grant 2018JJ3614, and in part by 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, SPL is introduced as a 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 models.

**INDEX TERMS** Multi-task learning, self-paced learning, 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 scenarios, 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 among different tasks and different learning samples in the learning process.

To address this deficiency, the third category of MTL [13], which is called Self-Paced Multi-Task Learning, has been developed recently. 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

The associate editor coordinating the review of this manuscript and approving it for publication was Xinyu Du<sup>ID</sup>.

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 obtains sample difficulty levels to select samples of each task. Secondly, it selects the samples of different difficulty levels and uses the state-of-the-art MTL models to train data iteratively until 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 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 and Iii [20] proposed the GO-MTL model to selectively share the information across the tasks. Subsequently, based on previous models, Jeong and Jun [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 prediction performance of the model.

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-to-Teach* (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 prior knowledge. 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 and Carbonell [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 difficulty levels of both tasks and instances per task,

and the *spMMTL* [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 FSPMTL to solve this problem. In the first stage, we use the SPL model to obtain the sample difficulty matrix  $\mathbf{E}$ , whose elements show the difficulties of the samples in each task. In the second stage, we select new training samples through the coefficient matrix  $\mathbf{E}$ , so that we could update the optimized MTL model to obtain the coefficient matrix  $\mathbf{W}$ . More details are discussed in the next section.

## III. FLEXIBLE SELF-PACED MULTI-TASK LEARNING

In this section, we introduce our FSPMTL model based on the BSPL model [15] and the VSTG-MTL model [21]. The details are shown below.

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

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

$$y_t^n = f(\mathbf{w}_t^T \mathbf{x}_t^n) \quad (1)$$

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

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

The optimization function for this problem is,

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

where  $L(\cdot, \cdot)$  is the empirical loss function, which is the logistic loss  $\sum_{n=1}^{N_t} \log(1 + \exp(-y_t^n \mathbf{v}_t^T \mathbf{U}^T \mathbf{x}_t^n))$  for a binary

classification problem;  $\|U\|_1 = \sum_{d=1}^D \sum_{m=1}^M |u_{dm}|$  is the  $\ell_1$  norm;  $\|U\|_{1,\infty} = \sum_{d=1}^D \|u^d\|_\infty$  is the  $\ell_{1,\infty}$  norm;  $\|v_l\|_k^{sp}$  is the  $k$ -support norm; where  $\gamma_1, \gamma_2$ , and  $\mu$  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  $E$  of the training samples. Assume that the training data can be divided into  $L$  levels based on the difficulty of the data samples,  $E$  is a three-dimensional matrix of  $\mathbb{R}^{L \times T \times N_t}$ , whose row vector is denoted as  $E = [e_1, e_2, \dots, e_L]^T$  and whose column vector is denoted as  $E = [e^1, e^2, \dots, e^T]$ . Each element  $e_l^t$  in  $E$  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  $\theta_l^t$ , which is the parameter of the decision function  $g$ , and the latent sample difficulty variable  $e_l^t = [e_l^{t,1}, \dots, e_l^{t,N_t}]$  by minimizing:

$$\min_{\theta_l^t, e_l^t} \sum_{n=1}^{N_t} e_l^{t,n} L(y_l^n; g(x_l^n, \theta_l^t)) + \mu R(\theta_l^t) - \sum_{k=1}^K \sum_{x_l^{nk} \in Cl_k} \lambda_k e_l^{t,nk} \tag{3}$$

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

$$e_l^{t,n*} = \begin{cases} 1, & \text{if } L_k(y_l^n; g(x_l^n, \theta_l^t)) < \lambda_k \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where  $L_k(y_l^n; g(x_l^n, \theta_l^t))$  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  $e_l$  of  $E$  represents the sample difficulty vector of the training samples. Therefore, we can train the model parameters ( $E, W$ ) with the following strategies.

The update process of parameters in the model is generally completed in two stages. In the first stage, by solving the equation 3, we can obtain the sample difficulty coefficient  $e_l^t$  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  $e_l$ , the VSTG-MTL model is used to train  $W_l$  by solving the

equation 2. Iteratively, when training processes of all  $L$  levels are completed, the final training coefficient matrix  $W$  is obtained by calculating the expectation of  $W_1$  to  $W_L$ . Specific calculations are shown in Step 3 and 4 as follows.

Specifically, the updating process of the model can be divided into four steps. When  $L, \theta_l, U_l^{init}$  and  $V_l^{init}$  are initialized, the following four steps will be iteratively completed.

*Step 1: Fix  $\theta_l^t$ , update  $e_l^t$ .*

First, for the  $t$ -th task, we fix  $\theta_l^t$  and then solve the following problem to update  $e_l^t$ :

$$\begin{aligned} e_l^{t*} &= \arg \min \sum_{n=1}^{N_t} e_l^{t,n} L(y_l^n; g(x_l^n, \theta_l^t)) - \sum_{k=1}^K \sum_{x_l^{nk} \in Cl_k} \lambda_k e_l^{t,nk} \\ &= \arg \min \sum_{k=1}^K \sum_{x_l^{nk} \in Cl_k} e_l^{t,nk} (L(y_l^{nk}; g(x_l^{nk}, \theta_l^{t,nk})) - \lambda_k) \end{aligned} \tag{5}$$

*Step 2: Fix  $e_l^t$ , update  $\theta_{l+1}^t$ .*

For the  $t$ -th task, we fix  $e_l^t$  and update  $\theta_{l+1}^t$  by solving:

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

We use the gradient descent algorithm to update the equation above, then increase  $\lambda_k$  and return to the Step 1 until all the instances are selected. Then, we can get the sample difficulty coefficient vector  $e_l^t$  of the  $t$ -th task.

When all tasks are executed, we can obtain  $e_l = [e_l^1, e_l^2, \dots, e_l^T]$  and  $E = [e_1, e_2, \dots, e_L]^T$ .

*Step 3: Fix  $e_l$ , update  $U_l$ .*

For each level of training tasks, we select the training sample through  $e_l$ , i.e.

$$X_l = X \quad (e_l \text{ is equal to } 1) \tag{7}$$

Similarly, we select the label  $y_l$  corresponding to  $X_l$ . Then, we update  $U_l$  with an alternating direction method of multipliers and an early stopping. The objective function is as follows:

$$U_l = \arg \min_{U_l} \left( \sum_{t=1}^T \frac{1}{N_j} L(y_l^t, X_l^t U_l v_l^t) + \gamma_1 \|U_l\|_1 + \gamma_2 \|U_l\|_{1,\infty} \right) \tag{8}$$

*Step 4: Fix  $U_l$ , update  $V_l$ .*

We use accelerated proximal gradient descent to solve the following equation and  $V_l = [v_l^1, \dots, v_l^T]$ :

$$v_l^t = \arg \min_{v_l^t} \left( \sum_{t=1}^T \frac{1}{N_j} L(y_l^t, X_l^t U_l v_l^t) + \mu \sum_{t=1}^T (\|v_l^t\|_k^{sp})^2 \right) \tag{9}$$

Next, the model iteratively executes the step 3 and 4 until the  $U_l$  and  $V_l$  coverage. After that, we calculate  $W_l = U_l V_l$ .

Finally, the training process will repeat  $L$  times until the samples of all  $L$  levels are trained, then the expectation value

**TABLE 1. The flexible self-paced multi-task learning (FSPMTL) model.**

<b>Algorithm 1</b> Flexible Self-Paced Multi-Task Learning (FSPMTL)	
<b>Input:</b>	Dataset $\mathbf{X}$ and $\mathbf{y}$
	The number of data difficulty levels $L$
<b>Output:</b>	Coefficient matrix $\mathbf{W}$
1.	<b>for</b> $t=1:\mathcal{T}$ <b>do</b>
2.	<b>for</b> $l=1:L$ <b>do</b>
3.	select <i>SPL</i> model to train to get $e_l^t$
4.	<b>end for</b>
5.	<b>end for</b>
6.	$e_t = [e_1^t, e_2^t, \dots, e_L^t]^T$
7.	$\mathbf{E} = [e_1, e_2, \dots, e_L]^T$
8.	<b>for</b> $l=1:L$ <b>do</b>
9.	select training samples $\mathbf{X}_l$ and $\mathbf{y}_l$ from $\mathbf{X}$ and $\mathbf{y}$ through $e_l$
10.	select <i>MTL</i> model for training based on $\mathbf{X}_l$ and $\mathbf{y}_l$ to obtain $\mathbf{W}_l$
11.	<b>end for</b>
12.	$\mathbf{W} = \mathbb{E}[\mathbf{W}_1, \dots, \mathbf{W}_L]$
13.	<b>return</b> $\mathbf{W}$

of  $\mathbf{W}_1, \dots, \mathbf{W}_L$  could be seemed as the final coefficient matrix  $\mathbf{W}$ .

#### D. FLEXIBLE 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 model and shown in Algorithm 1.

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 model is depended on the convergence of the *SPL* and *MTL* parts, which means the FSPMTL model 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 model 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 model is implemented based on the VSTG-MTL model and named as FSP-VSTG-MTL. Here, we compared our FSP-VSTG-MTL model 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 into the product of two low rank matrices in the model. This model would simultaneously perform variable selection and learn an overlapping group structure among learning tasks [21].
- **spMMTL method:** spMMTL is the acronym of Self-Paced Mean Regularized Multi-task Learning, and the model picks up the simple tasks based on the distance the model parameter vector of each task [14].

The parameter initialization of the FSP-VSTG-MTL model is divided into two parts. For the BSPL part, we selected half of the data points during the first iteration and then updated the  $\lambda_k$  with rise of 10% in the next iteration, that is,  $L = 6$ . Then we initially set  $e_l^{1:n} = 1 (n = 1, \dots, N_l)$  and ran the corresponding classification algorithm for 5 iterations to obtain an estimate of  $\theta_l$ . For the VSTG-MTL part, we selected the hyper-parameters of the VSTG-MTL model with the best performance according to the previous works. Specifically, we set the third regularization parameter  $\mu$  to be equal to the first regularization parameter  $\gamma_1$ . The regularization parameters are selected from the search grid  $\{2^{-10}, \dots, 2^3\}$ . Initial estimates of the matrix  $\mathbf{W}_l^{init}$  is implemented by logistic regression algorithm. The initial estimates of  $\mathbf{U}_l^{init}$  and  $\mathbf{V}_l^{init}$  are given by singular value decomposition of  $\mathbf{W}_l^{init}$ .

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 coefficient matrix. Then, we made predictions on the testing set to get its final classification effect. We ensured 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

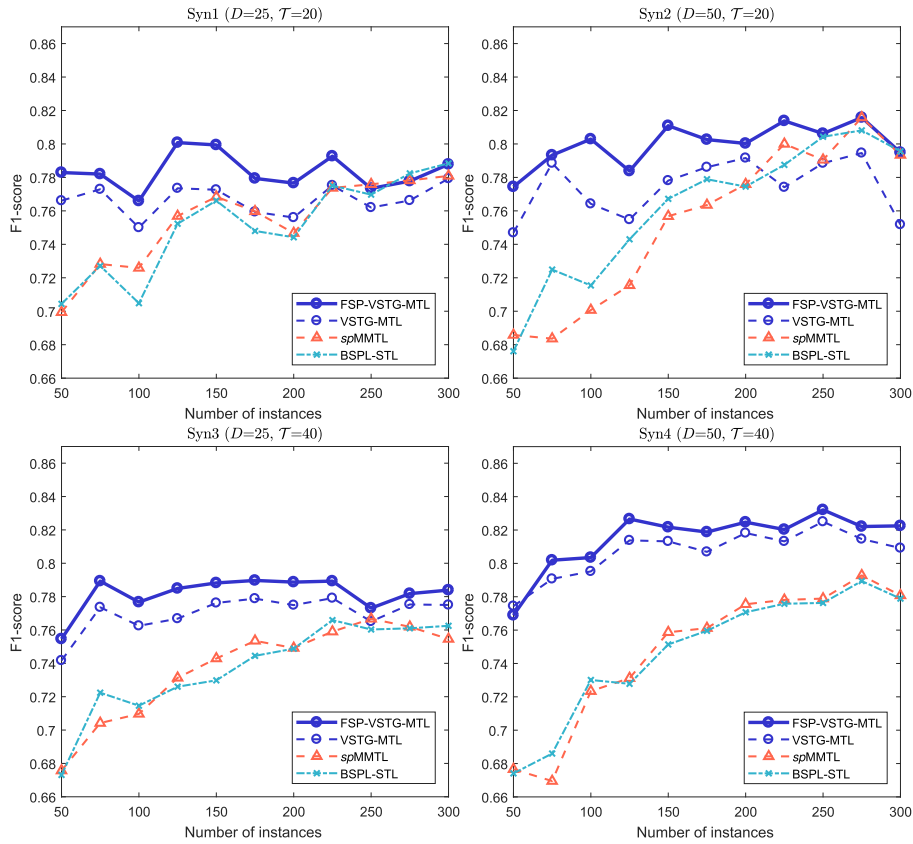
We generated four synthetic datasets as follows, which have different number of  $D$  dimensional variables and  $\mathcal{T}$  tasks. The instance  $\mathbf{x}_i^n$  is sampled from a *Standard Normal Distribution*  $N(0,1)$ , and the response is  $y_i^n = \text{sign}(\mathbf{w}_i^T \mathbf{x}_i^n +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

###### 1) SYN1

Syn1 has 25 dimensional variables and 20 tasks. For  $r = 1, \dots, M$ , the latent basis  $\mathbf{u}_r$  only has non-zero 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  $r = 1, \dots, M$ , the weighting vectors  $\mathbf{v}_{4r-3}, \dots, \mathbf{v}_{4r}$  only have nonzero values on the  $r$ -th and  $(r + 1)$ -th components. The last four weighting vectors  $\mathbf{v}_{4M-3}, \dots, \mathbf{v}_{4M}$  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.

###### 2) SYN2

Syn2 has 50 dimensional variables and 20 tasks. For  $r = 1, \dots, M$ , the latent basis  $\mathbf{u}_r$  only has non-zero values from the  $(8r - 7)$ -th to the  $(8r + 8)$ -th components. Similarly,  $\mathbf{V}$  is



**FIGURE 1. Results on four synthetic datasets with different number of instances, the dark blue solid line represents our proposed FSP-VSTG-MTL model.**

generated in the same way as Syn1. The nonzero values are generated by the same distribution as that used in Syn1.

3) SYN3

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

4) SYN4

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

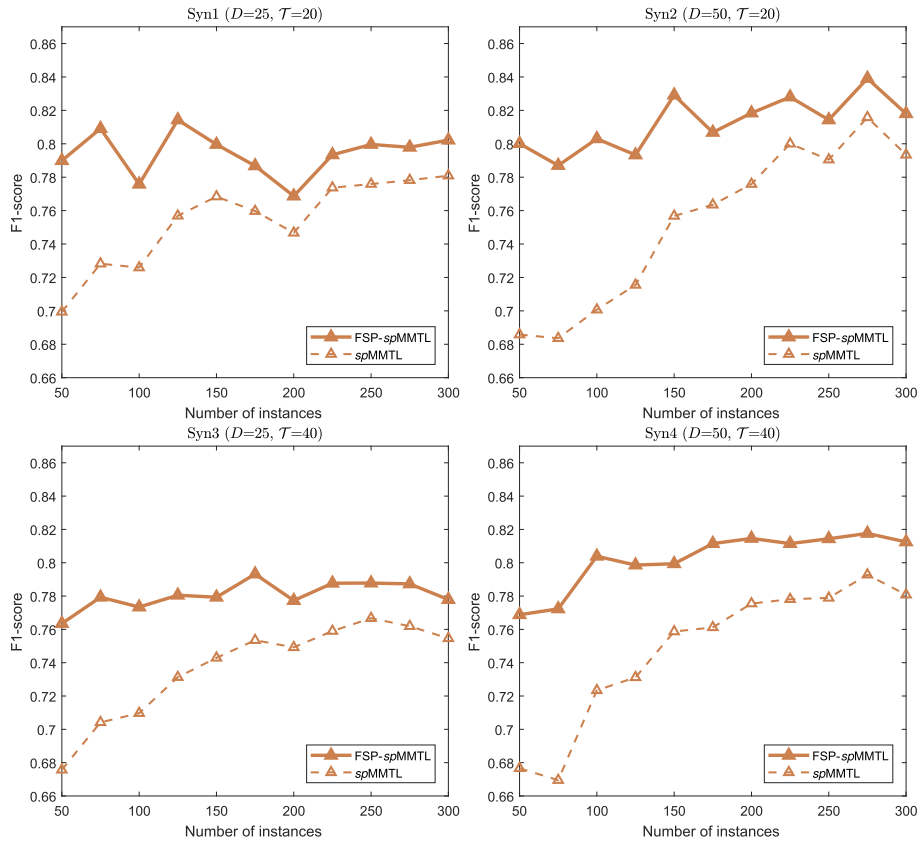
5) VARYING NUMBER OF INSTANCES

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 learning (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 instances, but also depends on the number of dimensions of the instance. In addition, compared to the state-of-the-art models (VSTG-MTL,  $spMMTL$ ), our proposed FSP-VSTG-MTL model is better off in most settings. It's worth noting that the performance of our model is 0.022, 0.0328, 0.0155 and 0.01 higher than that of the VSTG-MTL model on four synthetic datasets on average, respectively. Besides, to verify the flexibility of our proposed framework, we extended  $spMMTL$  with our FSPMTL framework to see if the model performance could be further improved.

6) FLEXIBILITY TEST OF THE FSPMTL FRAMEWORK

To further verify the flexibility of our framework, we extended the FSPMTL framework into the  $spMMTL$  model and named it FSP- $spMMTL$  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- $spMMTL$  model is better than the  $spMMTL$  model with 0.0535, 0.0792, 0.059 and 0.0727 on four synthetic



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

datasets on average, respectively, which demonstrates that our FSPMTL framework has achieved excellent results. Therefore, adding the FSPMTL framework during the training process can effectively improve the performance of the model.

#### 7) ROBUSTNESS TEST OF THE FSPMTL FRAMEWORK

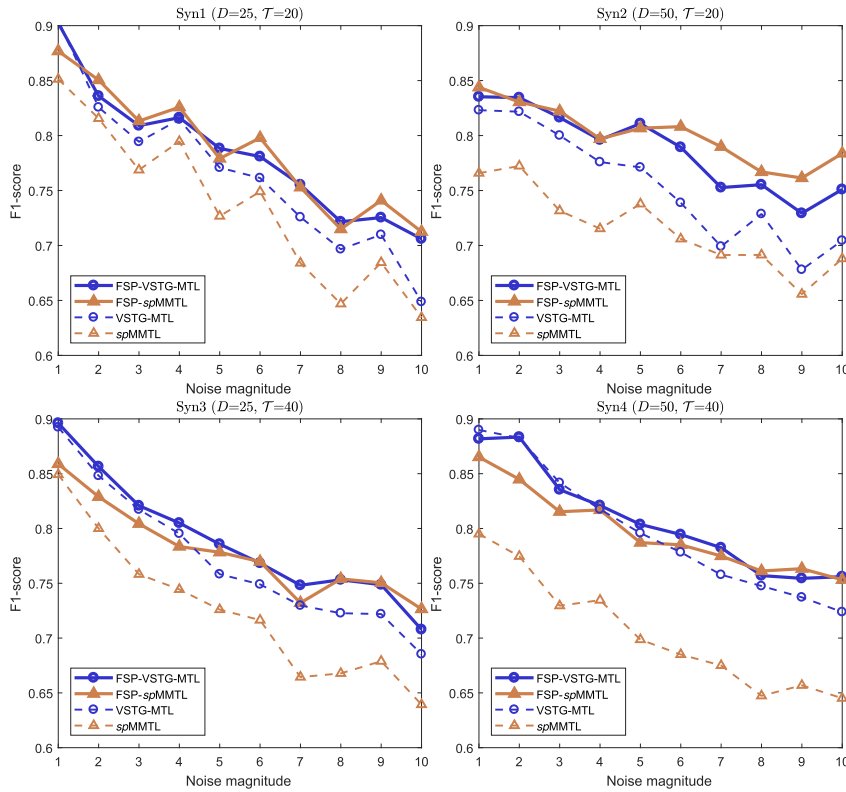
If we change the noise level of instances in the datasets, the learning difficulty of instances will change as well. Here, we varied the *Normal Distribution* of  $\sigma_t^n$  from  $N(0,1)$  to  $N(0,10)$ , increasing the variance by one each time. Meanwhile, we set 100 instances for each task. Figure 3 shows that performances of models 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  $\sigma_t^n$  is 1-5, on average, FSP-VSTG-MTL only performs better than VSTG-MTL with 0.0103, 0.0254, 0.0128 and  $-0.0005$  and FSP-spMMTL performs better than spMMTL with 0.0476, 0.1014, 0.0454 and 0.1064 on four synthetic datasets above. However, with the rise of the noise, the advantage of the FSPMTL framework is emerging. Specifically, when the variance of  $\sigma_t^n$  is 6-10, averagely, FSP-VSTG-MTL performs better than VSTG-MTL with 0.0415, 0.0644, 0.0326 and 0.0267 and FSP-spMMTL performs better than spMMTL with 0.0942, 0.1391, 0.1087 and 0.1597. The results demonstrate the superiority and flexibility of the FSPMTL framework.

#### 8) VISUALIZATION OF THE SELECTED SAMPLES

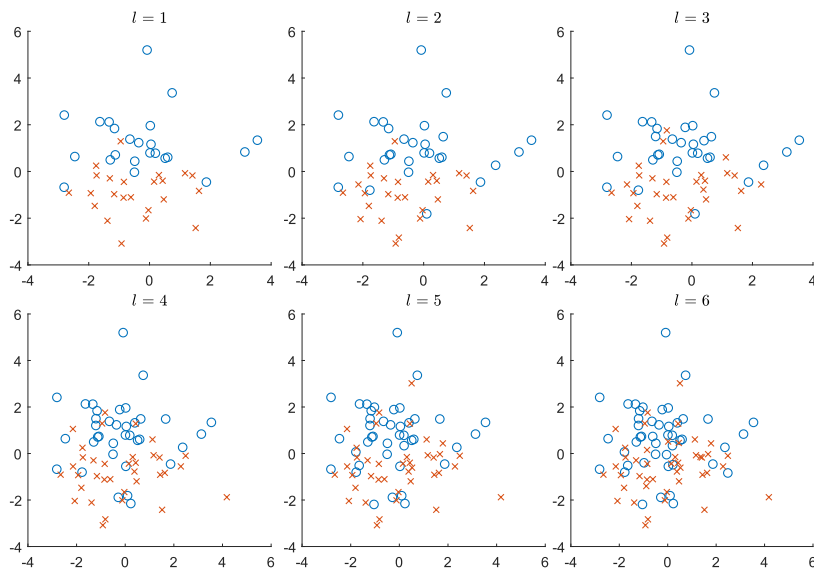
In order to make our experiment clearer, we took the first task as an example to visualize the selected samples with our FSP-VSTG-MTL model. When the task contains 100 instances, we first selected 50% of samples ( $l = 1$ ), and then increased 10% of samples in the next iteration. For the selected samples, 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 and negative samples. With the increase the  $l$ , the number of the selected samples increases gradually and the overlap area of samples in in two dimensions also increases gradually, which means the selected samples are difficult to learn.

#### 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 students' test grades.



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



**FIGURE 4.** The 2-D distribution of the selected samples in different difficulty levels with our FSP-VSTG-MTL model. Here “o” represents positive samples and “x” represents negative samples in the classification problem. When  $l = 1$ , there is a clear boundary between positive samples (top) and negative samples (bottom). With the increase the  $l$ , the number of the selected samples increases gradually and the overlap area of samples in in two dimensions also increases gradually, which means the selected samples are difficult to learn.

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 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 three 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 0.0053 and 0.0139 on average respectively. To sum up, by incorporating the self-paced learning regime into MTL, our method is effective in these experiments.

**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	<i>sp</i> MMTL	FSP- <i>sp</i> MMTL
F1-score	0.6269±0.0133	0.7234±0.0097	<b>0.7272±0.0103</b>	0.7138±0.0104	0.7234±0.0108

## 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 embed different MTL models into SPL models flexibly. Compared with the traditional model, extensive experiments show that our FSPMTL framework not only effectively improves the performance of the model, but also increases the flexibility and robustness. For future work, we would like to introduce the prior knowledge in the framework and apply this method in real-world applications.

## REFERENCES

- R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, no. 1, pp. 41–75, 1997, doi: [10.1023/A:1007379606734](https://doi.org/10.1023/A:1007379606734).
- 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>
- Z. Kang, K. Grauman, and F. Sha, "Learning with whom to share in multi-task feature learning," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, Bellevue, WA, USA, Jun./Jul. 2011, pp. 521–528. [Online]. Available: [https://icml.cc/2011/papers/344\\_icmlpaper.pdf](https://icml.cc/2011/papers/344_icmlpaper.pdf)
- J. Pu, Y. Jiang, J. Wang, and X. Xue, "Multiple task learning using iteratively reweighted least square," in *Proc. 23rd Int. Joint Conf. Artif. Intell. (IJCAI)*, Beijing, China, Aug. 2013, pp. 1607–1613 [Online]. Available: <http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6857>
- J. Pu, J. Wang, Y.-G. Jiang, and X. Xue, "Multiple task learning with flexible structure regularization," *Neurocomputing*, vol. 177, pp. 242–256, Feb. 2016, doi: [10.1016/j.neucom.2015.11.029](https://doi.org/10.1016/j.neucom.2015.11.029).
- S. Zhong, J. Pu, Y.-G. Jiang, R. Feng, and X. Xue, "Flexible multi-task learning with latent task grouping," *Neurocomputing*, vol. 189, pp. 179–188, May 2016.
- A. Argyriou, T. Evgeniou, and M. Pontil, "Convex multi-task feature learning," *Mach. Learn.*, vol. 73, no. 3, pp. 243–272, Dec. 2008.
- Y. Zhang and D.-Y. Yeung, "Transfer metric learning by learning task relationships," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, Washington, DC, USA, Jun. 2010, pp. 1199–1208, doi: [10.1145/1835804.1835954](https://doi.org/10.1145/1835804.1835954).
- Y. Yang, J. Yang, J. Yan, S. Liao, D. Yi, and S. Z. Li, "Salient color names for person re-identification," in *Proc. 13th Eur. Conf. (ECCV)*, Zürich, Switzerland, Sep. 2014, pp. 536–551, doi: [10.1007/978-3-319-10590-1\\_35](https://doi.org/10.1007/978-3-319-10590-1_35).
- S. Kim and E. P. Xing, "Tree-guided group lasso for multi-task regression with structured sparsity," in *Proc. ICML*, 2009, p. 1.
- A. Schwaighofer, V. Tresp, and K. Yu, "Learning Gaussian process kernels via hierarchical Bayes," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, Vancouver, BC, Canada, Dec. 2004, pp. 1209–1216. [Online]. Available: <http://papers.nips.cc/paper/2595-learning-gaussian-process-kernels-via-hierarchical-bayes>
- R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *J. Mach. Learn. Res.*, vol. 6, pp. 1817–1853, Nov. 2005.
- C. Li, J. Yan, F. Wei, W. Dong, Q. Liu, and H. Zha, "Self-paced multi-task learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, Feb. 2017, pp. 2175–2181. [Online]. Available: <http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14535>
- K. Murugesan and J. Carbonell, "Self-paced multitask learning with shared knowledge," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Melbourne, VIC, Australia, Aug. 2017, pp. 2522–2528, doi: [10.24963/fjcai.2017/351](https://doi.org/10.24963/fjcai.2017/351).
- Y. Ren, P. Zhao, Z. Xu, and D. Yao, "Balanced self-paced learning with feature corruption," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Anchorage, AK, USA, May 2017, pp. 2064–2071, doi: [10.1109/IJCNN.2017.7966104](https://doi.org/10.1109/IJCNN.2017.7966104).
- T. Evgeniou and M. Pontil, "Regularized multi-task learning," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, New York, NY, USA: ACM, 2004, pp. 109–117.
- A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 41–48.
- Y. Zhou, J. Wang, C. Zhu, and W. Zhang, "Multiple dags learning with non-negative matrix factorization," in *Proc. 3rd Workshop Adv. Methodologies Bayesian Netw. (AMB/N)*, Kyoto, Japan, Sep. 2017, pp. 81–92. [Online]. Available: <http://proceedings.mlr.press/v73/zhou17a.html>
- 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, Feb. 2020, doi: [10.1109/TPAMI.2018.2881114](https://doi.org/10.1109/TPAMI.2018.2881114).
- A. Kumar and H. D. Iii, "Learning task grouping and overlap in multi-task learning," *Comput. Sci.*, vol. 2, pp. 1723–1730, Jun. 2012.
- J.-Y. Jeong and C.-H. Jun, "Variable selection and task grouping for multi-task learning," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, London, U.K., Jul. 2018, pp. 1589–1598, doi: [10.1145/3219819.3219992](https://doi.org/10.1145/3219819.3219992).
- Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. 26th Annu. Int. Conf. Mach. Learn. (ICML)*, Montreal, QC, Canada, Jun. 2009, pp. 41–48, doi: [10.1145/1553374.1553380](https://doi.org/10.1145/1553374.1553380).
- C. Gong, "Exploring commonality and individuality for multi-modal curriculum learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, Feb. 2017, pp. 1926–1933. [Online]. Available: <http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14205>
- 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, doi: [10.1145/3322122](https://doi.org/10.1145/3322122).
- 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, Jul. 2016, doi: [10.1109/TIP.2016.2563981](https://doi.org/10.1109/TIP.2016.2563981).
- M. P. Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 2010, pp. 1189–1197. [Online]. Available: <http://papers.nips.cc/paper/3923-self-paced-learning-for-latent-variable-models>
- C. Gong, D. Tao, J. Yang, and W. Liu, "Teaching-to-learn and learning-to-teach for multi-label propagation," in *Proc. 13th AAAI Conf. Artif. Intell.*, Phoenix, AZ, USA, Feb. 2016, pp. 1610–1616. [Online]. Available: <http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/11780>
- C. Gong, D. Tao, W. Liu, L. Liu, and J. Yang, "Label propagation via teaching-to-learn and learning-to-teach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 6, pp. 1452–1465, Jun. 2017, doi: [10.1109/TNNLS.2016.2514360](https://doi.org/10.1109/TNNLS.2016.2514360).
- K. Ghasedi, X. Wang, C. Deng, and H. Huang, "Balanced self-paced learning for generative adversarial clustering network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, Jun. 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](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," *IEEE Trans. Syst., Man, Cybern.*, pp. 1–15, Dec. 2019, doi: [10.1109/TCYB.2019.2957101](https://doi.org/10.1109/TCYB.2019.2957101).
- [31] L. Han, D. Zhang, D. Huang, X. Chang, J. Ren, S. Luo, and J. Han, "Self-paced mixture of regressions," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Melbourne, VIC, Australia, Aug. 2017, pp. 1816–1822, doi: [10.24963/ijcai.2017/252](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, Apr. 2019, doi: [10.1007/s11263-018-1112-4](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 *Proc. 29th AAAI Conf. Artif. Intell.*, Austin, TX, USA, Jan. 2015, pp. 2694–2700. [Online]. Available: <http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9750>



**LIJIAN SUN** received the bachelor degree from the National University of Defense Technology, China, in 2018, where he is currently pursuing the master degree with the Science and Technology on Information Systems Engineering Laboratory. His research interests include self-paced learning and multi-task learning.



**YUN ZHOU** (Member, IEEE) received the Ph.D. degree in computer science from the Queen Mary University of London, in 2015. He is currently an Assistant Professor with the Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology, Changsha, China. He applies the below 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. His research interests include machine learning and probabilistic graphical models.

• • •