frontiers | Frontiers in Bioengineering and Biotechnology

ORIGINAL RESEARCH
published: 04 August 2022
doi: 10.3389/fbioe.2022.890132

Multi-Task Learning Based on Stochastic Configuration Neural Networks

Xue-Mei Dong, Xudong Kong and Xiaoping Zhang*

Collaborative Innovation Center of Statistical Data Engineering, Technology & Application, School of Statistics and Mathematics, Zhejiang Gongshang University, Hangzhou, China

When the human brain learns multiple related or continuous tasks, it will produce knowledge sharing and transfer. Thus, fast and effective task learning can be realized. This idea leads to multi-task learning. The key of multi-task learning is to find the correlation between tasks and establish a fast and effective model based on these relationship information. This paper proposes a multi-task learning framework based on stochastic configuration neural networks. It organically combines the idea of the classical parameter sharing multi-task learning with that of constraint sharing configuration in stochastic configuration neural networks. Moreover, it provides an efficient multi-kernel function selection mechanism. The convergence of the proposed algorithm is proved theoretically. The experiment results on one simulation data set and four real life data sets verify the effectiveness of the proposed algorithm.

Keywords: multi-task learning, neural networks, stochastic configuration, knowledge sharing and transfer, supervised mechanism

OPEN ACCESS

Edited by:
Gongfa Li,
Wuhan University of Science and Technology, China

Reviewed by:
Inci M Baytas,
Boğaziçi University, Turkey
Tiantian He,
Agency for Science, Technology and Research (A∗STAR), Singapore

****Correspondence:***
Xudong Kong
simon_kong@163.com

Specialty section:
This article was submitted to Bionics and Biomimetics, a section of the journal Frontiers in Bioengineering and Biotechnology

Received: *05 March 2022*
Accepted: *16 June 2022*
Published: *04 August 2022*

Citation:
Dong X-M, Kong X and Zhang X (2022) Multi-Task Learning Based on Stochastic Configuration Neural Networks. Front. Bioeng. Biotechnol. 10:890132. doi: 10.3389/fbioe.2022.890132

1 INTRODUCTION

In supervised machine learning, we often encounter situations that establishing models for several related tasks, such as searching cancer sites, identifying cancer types, judging cancer stages and so on, based on cancer image data. Generally, these tasks are undertaken separately, which we refer to as single-task supervised learning (STSL) in traditional machine learning (Ben-David and Schuller, 2003). These models do not consider the correlation among multiple tasks so some common information in model parameters or data features is lost. In particular, when the training sample size of a single task is insufficient, it is difficult for STSL to capture enough information, which results in poor generalization performance. Multi-task supervised learning (MTSL) provides a solution for such a situation. It improves the performance of each task by setting shared representations among related tasks (Baxter, 2000; Argyriou et al., 2007; Liu et al., 2017). In a sense, a very important reason why human beings can learn based on a small number of samples is that human beings can make full use of various senses to obtain enough information and synthesize relevant information. MTSL is one of the ways to realize this idea.

Classical MTSL can be roughly divided into two categories, namely, MTSL based on constraint sharing and MTSL based on parameter sharing. In relation to the first method, Argyriou et al. (2008) proposed the MTL-L_{21} based on regularization strategies, that was achieved by adding a regularization term for all the tasks' objective function coefficients on the cost function. But this method performs poorly when data features have the problem of collinearity. To reduce the impact of this problem, Chen et al. (2012) added a quadratic regularization term for all the tasks' objective function coefficients based on MTL-L_{21}. In 2015, Duong et al. (2015) used L_2 distance to regularize the parameters in their multi-task neural networks, so that each task has similar but different model parameters. In 2017, Yang and Hospedales (2017) used the trace norm to implement Duong's model. In 2019, Oliveira et al. (2019) attempted to conceive a group LASSO with asymmetric transference formulation in multi-task learning, looking for the best of both worlds in a framework that admits the overlap of groups. Since all of these MTSL methods need to learn sparse features, their performance is not ideal when the data has few features. The MTSL methods based on parameter sharing (Caruana, 1997; Jacob et al., 2009) are not affected by this problem. In 1997, Caruana (Caruana, 1997) proposed a MTSL method (MTL) based on backpropagation neural networks. He mirrored the correlation information by sharing the input and hidden layer neurons among different tasks. In 1998, Lecun et al. (1998) used convolutional neural networks, named as LeNet-5, for document recognition on the basis of MTL. Their results clearly demonstrated the advantages of training a recognizer at the word level, rather than training it on presegmented, hand-labeled, isolated characters. In 2018, Ma et al. (2018) proposed multi-gate mixture-of-experts (MMoE), which adapted the mixture-of-experts (MoE) structure to multi-task learning by sharing the expert submodels across all tasks, while also had a gating network trained to optimize each task. In 2021, Zhang et al. (Zhang et al., 2021) developed a programming framework, AutoMTL, which generates compact multi-task models given an arbitrary input backbone convolutional neural network model and a set of tasks. However, these methods have high computational complexity and poor learning performance when the training samples are insufficient.

To address the aforementioned problems, this paper proposes a MTSL method based on a constraint sharing framework of stochastic configuration networks (SCNs) proposed by Wang et al. (Wang and Li, 2017a; Wang and Li, 2017b) Instead of the complex gradient descent method for solving the weight parameters of hidden layer nodes in general neural networks, SCNs use a supervision mechanism to stochastically configure these parameters. This stochastic configuration mechanism greatly reduces the computational complexity. Inspired by this idea, we establish a multi-task supervised learning algorithm based on stochastic configuration radial basis networks (MTSL-SCRBN). The main contributions of this study are as follows.

1. We combine constraint sharing of SCNs and parameter sharing of MTSL organically. The shared parameters are stochastically configured under certain constraint, which has low computational complexity. At the same time, to improve learning performance, the radial basis functions (Powell, 1987; Broomhead and Lowe, 1988) with different scale parameters are used as the basis functions to replace the original sigmoid functions of SCNs.
2. Two types of difficult to choice hyper parameters of the proposed model, the scale parameters and the centers of the radial basis functions, are stochastically configured during the learning process.

The rest of the paper is organized as follows. In **Section 2**, we briefly review MTL-L_{21}, MTEN, MTL, and SCNs. **Section 3** details our proposed algorithm MTSL-SCRBN and proves its convergence. The experimental results of these algorithms on one simulation data set and five real data sets are detailed in **Section 4**. **Section 5** summarizes this paper.

2 RELATED WORK

Firstly, we introduce some notations. Suppose that there are M supervised learning tasks. The samples of the m-th task are given by,

$$\left\{\left(\mathbf{x}_1^m, y_1^m\right), \ldots, \left(\mathbf{x}_{N_m}^m, y_{N_m}^m\right)\right\}, \tag{1}$$

where $\mathbf{x}_i^m = [x_{i,1}^m, \ldots, x_{i,d}^m]^{\mathrm{T}} \in R^d$, $y_i^m \in R$, $i = 1, \ldots, N_m$, $m = 1, \ldots, M$, and T means transpose transform.

2.1 Multi-Task Learning Methods Based on Constraint Sharing

Inspired by group sparsity, Argyrios et al. (Argyriou et al., 2008) proposed the MTL-L_{21} method to learn the correlation among multiple tasks under a regularization strategy. It can be described as the following optimization problem,

$$V^*_{MTL-L_{21}} = \underset{V\in R^{d\times M}}{\arg\min} \sum_{m=1}^{M} \|X^m\mathbf{v}^m - \mathbf{y}^m\|_F^2 + \lambda\|V\|_{2,1},$$

where $X^m = [\mathbf{x}_1^m, \ldots, \mathbf{x}_{N_m}^m]^{\mathrm{T}} \in R^{N_m\times d}$, $\mathbf{y}^m = [y_1^m, \ldots, y_{N_m}^m]^{\mathrm{T}} \in R^{N_m}$, $\|\cdot\|_F$ is the Frobenius norm, $V = [\mathbf{v}^1, \ldots, \mathbf{v}^M]$ on behalf of the model coefficient matrix, $\mathbf{v}^m = [v_1^m, \ldots, v_d^m]^{\mathrm{T}}$ represents the m-th column of V, which is the coefficient vector of the m-th task. λ represents the regularization coefficient and $\|V\|_{2,1} = \sum_{i=1}^{d}\sqrt{\sum_{m=1}^{M}(v_i^m)^2}$. For the input $\tilde{\mathbf{x}}^m$ of the m-th task, MTL-L_{21} gives the predicted value $f(\tilde{\mathbf{x}}^m) = \tilde{\mathbf{x}}^{m^{\mathrm{T}}}\mathbf{v}^{m*}_{MTL-L_{21}}$.

When data features have the problem of collinearity, MTL-L_{21} will have an unstable prediction performance. Xi Chen et al. (Chen et al., 2012) proposed the MTEN method by adding another quadratic regularization term for the objective function coefficients of all tasks on the basis of MTL-L_{21}. It can be described as the following optimization problem,

$$\begin{aligned} V^*_{MTEN} = \underset{V\in R^{d\times M}}{\arg\min} \sum_{m=1}^{M} \frac{1}{2n}\|X^m\mathbf{v}^m - \mathbf{y}^m\|_F^2 + \lambda\rho\|V\|_{2,1} \\ + \frac{\lambda(1-\rho)}{2}\|V\|_F^2, \end{aligned}$$

where $\rho \in [0, 1]$ represents the elastic net mixing parameter. For the input $\tilde{\mathbf{x}}^m$ of the m-th task, MTEN gives the predicted value $f(\tilde{\mathbf{x}}^m) = \tilde{\mathbf{x}}^{m^{\mathrm{T}}}\mathbf{v}^{m*}_{MTEN}$.

In the case of insufficient data features and data size, the two algorithms MTL-L_{21} and MTEN cannot obtain enough information by learning sparse features, which leads to poor prediction performance.

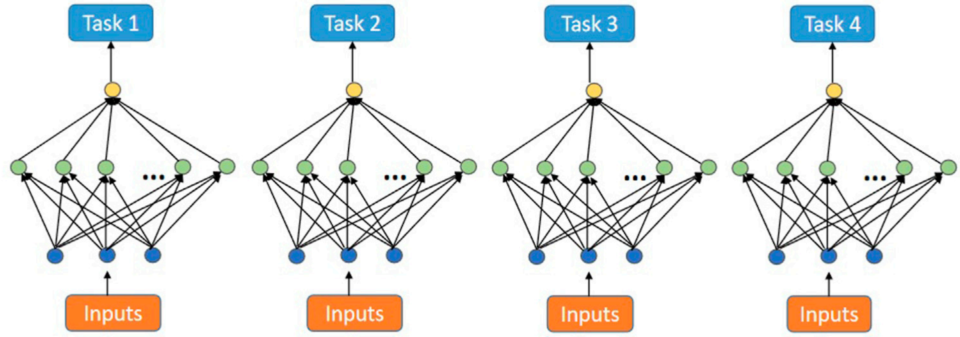

FIGURE 1 | Single-task backpropagation nets.

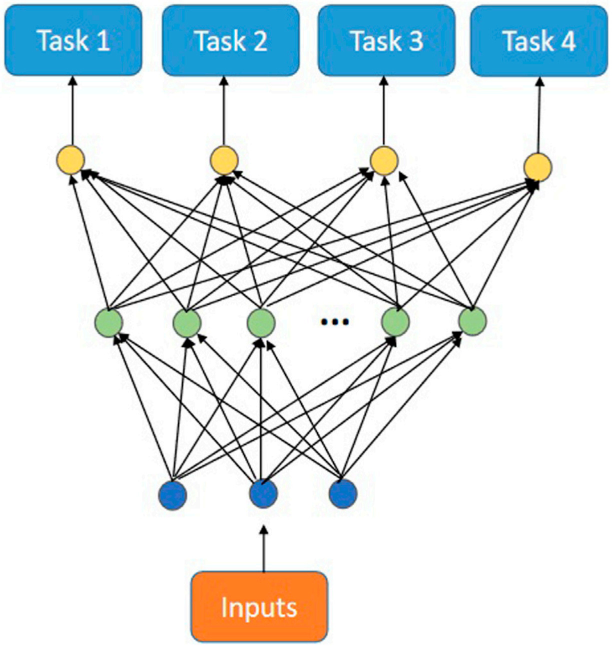

FIGURE 2 | The multi-task backpropagation net.

2.2 Multi-Task Learning Methods Based on Parameter Sharing

Caruana (1997) implemented MTSL on backpropagation nets by sharing input and hidden layer neurons among different tasks. Essentially, this method optimizes the choice of function space by the correlation among tasks and obtains better internal weight parameters.

Figure 1 shows the process of traditional backpropagation nets to deal with four related tasks. This method ignores the information among related tasks. Especially in the case of insufficient data samples, these models may have problems such as over-fitting.

Figure 2 shows the multi-task backpropagation net (MTL) conceived by Caruana. In MTL, each task shares input and hidden layer neurons.

Compared with the data form given in **Eq. 1**, the data form suitable for MTL is that different tasks have the same input,

$$X^1 = \ldots = X^M = X = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^{\mathrm{T}} \in R^{N\times d},$$

and the output is,

$$Y = [\mathbf{y}^1, \ldots, \mathbf{y}^M] \in R^{N\times M}, \mathbf{y}^m = [y_1^m, \ldots, y_N^m]^{\mathrm{T}} \in R^N.$$

MTL can be described as follows

$$(\boldsymbol{\beta}^*, W^*, \mathbf{b}^*) = \underset{\substack{\boldsymbol{\beta}\in R^{S\times M} \\ W \in R^{S\times d} \\ \mathbf{b}\in R^S}}{\arg\min} \sum_{m=1}^{M} \left\| \mathbf{y}^m - \sum_{j=1}^{S} \mathbf{g}_j \beta_j^m \right\|,$$

where S is the number of hidden layer nodes, β_j^m is the external weight parameter of the m-th task in the j-th hidden layer node. $\mathbf{g}_j \coloneqq \mathbf{g}_j(X) = [g(\mathbf{x}_1^{\mathrm{T}}\mathbf{w}_j + b_j), \ldots, g(\mathbf{x}_N^{\mathrm{T}}\mathbf{w}_j + b_j)]^{\mathrm{T}}$ (here g represents the sigmoid function), $\mathbf{w}_j = [w_{j,1}, \ldots, w_{j,d}]^{\mathrm{T}}$ and b_j represent the internal weight parameter vector and the bias internal weight parameter shared by the backpropagation net. $\boldsymbol{\beta} \in R^{N\times M}$, $W \in R^{d\times N}$, $\mathbf{b} \in R^N$ are the corresponding parameter matrix. For the input $\tilde{\mathbf{x}}^m$ of the m-th task, MTL gives the predicted value $f(\tilde{\mathbf{x}}^m) = \sum_{j=1}^{S} g(\tilde{\mathbf{x}}^{m^{\mathrm{T}}}\mathbf{w}_j^* + b_j^*)\beta_j^{m*}$.

From a mathematical point of view, the essence of the backpropagation net is the gradient descent algorithm. In single-task supervised learning, the backpropagation net may fall into local optimum. However, in multi-task supervised learning, the local optimum of different tasks is in different positions, and the interaction among tasks can help the hidden layer to escape from local optimums (Caruana, 1997).

2.3 Stochastic Configuration Networks

Wang and Li (Wang and Li, 2017a; Wang and Li, 2017b) proposed supervised stochastic configuration networks, and implemented SCNs using three algorithms SC-i (i=I,II,III). SC-i starts with a small network structure (Tin-Yan Kwok and Dit-Yan Yeung, 1997), and uses a supervision mechanism to add hidden layer neurons until the model meets a predetermined error criterion. Since SC-III performs the best of the three algorithms, we next describe the implementation of the SC-III algorithm.

Suppose a SC-III with $L - 1$ hidden layer nodes has already been constructed, that is,

$$f^{L-1}(\mathbf{x}^{\mathrm{T}}) = \sum_{j=1}^{L-1} g\left(\mathbf{x}^{\mathrm{T}}\mathbf{w}_j^* + b_j^*\right)\beta_j^{L-1}, \; (L = 2, 3, \ldots, f^0 \in [0, 0, \ldots, 0] \in R^q),$$

where $\beta_j^{L-1} = [\beta_{j,1}^{L-1}, \ldots, \beta_{j,q}^{L-1}] \in R^q$ represents the optimal external weight parameter of the j-th hidden layer node, $\mathbf{w}_j^*$ and b_j^* represent the optimal internal weight parameters of the j-th hidden layer node.

For training data set $X = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^{\mathrm{T}} \in R^{N\times d}$, $Y = [\mathbf{y}^1, \ldots, \mathbf{y}^q] \in R^{N\times q}$, where $\mathbf{x}_i = [x_{i,1}, \ldots, x_{i,d}]^{\mathrm{T}}$. Let $f^{L-1} := f^{L-1}(X) = \sum_{j=1}^{L-1} \mathbf{g}_j(X)\beta_j^{L-1}$ and $\boldsymbol{e}^{L-1} := Y - f^{L-1} := [\mathbf{e}_1^{L-1}, \ldots, \mathbf{e}_q^{L-1}]$, which is the residual error matrix of the $(L - 1)$-th hidden layer node. If $\|\boldsymbol{e}^{L-1}\|_F$ does not meet the predetermined error criteria, SC-III needs to generate a new hidden layer node, that is, stochastically configure internal weight parameters $\mathbf{w}_L$, b_L from an uniform distribution $U^{d+1}[-\Phi, \Phi]$, $\Phi > 0$. For new variables,

$$\xi_n^L = \frac{\langle \mathbf{e}_n^{L-1}, \mathbf{g}_L \rangle^2}{\|\mathbf{g}_L\|_2^2} - (1 - r - \mu_L)\|\mathbf{e}_n^{L-1}\|_2^2, \; n = 1, \ldots, q,$$

if $\min_n(\xi_n^L) \geq 0$, then $\mathbf{w}_L$, b_L are considered to meet the condition, otherwise $\mathbf{w}_L$, b_L need to be configured again. With the qualified internal weight parameters $\mathbf{w}_L^*$ and b_L^*, SC-III obtains the optimal external weight parameter vector by the following optimization problem,

$$\boldsymbol{\beta}^L = \underset{\beta \in R^{L\times q}}{\arg\min} \left\| Y - \sum_{j=1}^{L} \mathbf{g}_j \beta_j \right\|_F^2.$$

The leading model $f^L(\mathbf{x}) = \sum_{j=1}^{L} g(\mathbf{x}^{\mathrm{T}}\mathbf{w}_j^* + b_j^*)\beta_j^L$ will have an improved residual error. Repeat the above steps to add hidden layer nodes until the residual error meets the predetermined error criteria.

3 MULTI-TASK SUPERVISED LEARNING BASED ON STOCHASTIC CONFIGURATION RADIAL BASIS NETWORKS

In this section, we introduce the proposed MTSL-SCRBN algorithm.

3.1 Model Introduction

In order to combine SCNs and MTL organically, we need to change the data form given in **Eq. 1**. In MTSL-SCRBN, first, we require each task to have a same number of samples, namely, $N_1 = \ldots = N_M =: N$. (If the number of samples for each task is different, this requirement can be achieved by random sampling.) Then we merge the input data of different tasks into a new input data, that is, the i-th new input data is $\mathbf{X}_i^{\mathrm{T}} := (\mathbf{x}_i^{1^{\mathrm{T}}}, \ldots, \mathbf{x}_i^{M^{\mathrm{T}}}) \in R^{1\times Md}$, where $\mathbf{x}_i^m \in R^d$ is the i-th input of the m-th task, for $i = 1, \ldots, N$ and $m = 1, \ldots, M$. The corresponding i-th new output data is $\mathbf{y}_i := (y_i^1, \ldots, y_i^M)^T$, where y_i^m is the original output of $\mathbf{x}_i^m$ in the m-th task. The goal of our MTSL-SCRBN is to establish an appropriate model from R^{Md} to R^M based on these data $\{(\mathbf{X}_i, \mathbf{y}_i)_{i=1}^N\}$.

In order to obtain good learning performance, we use the following radial basis function $k_\sigma(\mathbf{x}, \mathbf{x}')$ as model's basis function,

$$k_\sigma(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right),$$

where $\mathbf{x}$ is the input, $\mathbf{x}'$ represents the center and σ is the scale parameter.

Suppose $f^0 = [0, 0, \ldots, 0] \in R^M$, for $L = 2, 3, \ldots$, it is assumed that a MTSL-SCRBN with $L - 1$ hidden layer nodes has already been constructed as follows,

$$f^{L-1}(\mathbf{X}) = \sum_{j=1}^{L-1} \left[k_{\sigma_j^*}(\mathbf{x}^1, \mathbf{x}^{j*}), \ldots, k_{\sigma_j^*}(\mathbf{x}^M, \mathbf{x}^{j*})\right]\boldsymbol{\beta}_j^{L-1},$$

where $\mathbf{X}^{\mathrm{T}} = (\mathbf{x}^{1^{\mathrm{T}}}, \ldots, \mathbf{x}^{M^{\mathrm{T}}})$ is a new input formed by the inputs of M tasks, and $\boldsymbol{\beta}_j^{L-1} = [\beta_j^{1,L-1}, \ldots, \beta_j^{M,L-1}] \in R^{M\times M}$,where $\beta_j^{m,L-1} \in R^M$ represents the optimal external weight parameter vector of the m-th task in the j-th hidden layer node, $\mathbf{x}^{j\star}$ and $\sigma_j^\star$ are the optimal center and the optimal scale parameter of the radial basis function in the j-th hidden layer node, respectively. Different from the traditional learning of radial basis neural network, in our MTSL-SCRBN, the optimal centers and the optimal scale parameters at each step are randomly assigned by a shared supervision mechanism given in the following. This is simple to implement and easy to obtain a learning model with good performance.

Denote $\mathbf{k}_j^m := [k_{\sigma_j^*}(\mathbf{x}_1^m, \mathbf{x}^{j*}), \ldots, k_{\sigma_j^*}(\mathbf{x}_N^m, \mathbf{x}^{j*})]^{\mathrm{T}}, K_j := [\mathbf{k}_j^1, \ldots, \mathbf{k}_j^M] \in R^{N\times M}$ and $f^{L-1} := \sum_{j=1}^{L-1} K_j\boldsymbol{\beta}_j^{L-1}$, $Y = [\mathbf{y}_1, \ldots, \mathbf{y}_N]^{\mathrm{T}} \in R^{N\times M}$. Then, let $\boldsymbol{e}^{L-1}{:=}Y - f^{L-1}{:=}[\mathbf{e}^{1,L-1}, \ldots, \mathbf{e}^{M,L-1}]$ be the residual error matrix of the $(L - 1)$-th hidden layer node. If $\|\boldsymbol{e}^{L-1}\|_F$ does not meet the predetermined error criteria, MTSL-SCRBN needs to generate a new hidden layer node, that is, stochastically configure the scale parameter σ_L from $U[0, \Omega]$, $\Omega > 0$ and the center of the radial basis function $\mathbf{x}^L$ from $\{\mathbf{x}_i^m\colon i = 1, \ldots, N, \; m = 1, \ldots, M\}$.

Similar to that in SCNs, we introduce a variable $\xi^{m,L}$ in our multi-task learning case as follows,

$$\xi^{m,L} = \frac{1}{M}\frac{\langle \mathbf{e}^{m,L-1}, \mathbf{k}_L^m \rangle^2}{\|\mathbf{k}_L^m\|_2^2} - \left(1 - r - \mu_{L,r}\right)\|\mathbf{e}^{m,L-1}\|_2^2.$$

Here $\mathbf{k}_L^m = [k_{\sigma_L}(\mathbf{x}_1^m, \mathbf{x}^L), \ldots, k_{\sigma_L}(\mathbf{x}_N^m, \mathbf{x}^L)]^{\mathrm{T}}, 0 < r < 1$ is a given constant and $\mu_{L,r} = \frac{1-r}{L+1}$.

If $\sum_{m=1}^{M} \xi^{m,L} \geq 0$, then σ_L, $\mathbf{x}^L$ are considered to meet the condition, otherwise, σ_L, $\mathbf{x}^L$ need to be configured again. With the qualified parameters $\sigma_L^\star$ and $\mathbf{x}^{L*}$, MTSL-SCRBN obtains the optimal external weight parameter vector by the following optimization problem,

$$\boldsymbol{\beta}^L = \underset{\beta \in R^{LM\times M}}{\arg\min} \left\| Y - \sum_{j=1}^{L} K_j\boldsymbol{\beta}_j \right\|_F^2.$$

The leading model,

$$f^L(\mathbf{X}) = \sum_{j=1}^{L} \left[k_{\sigma_j^*}(\mathbf{x}^1, \mathbf{x}^{j*}), \ldots, k_{\sigma_j^\star}(\mathbf{x}^M, \mathbf{x}^{j*}))\right]\boldsymbol{\beta}_j^L,$$

TABLE 1 | Parameter description.

Algorithms	Parameters	Parameters' Range
MTSL-SCRBN	RBF scale σ	$\sigma \sim U[0, \Omega]$, $\Omega \in (0, 100]$
MTEN	Regularization parameters λ, the elastic net mixing parameter ρ	$\lambda \in \{10^t, t = -6, -5.5, \ldots, 6\}$, $\rho \in [0, 1]$
SVM	RBF scale σ, Penalty parameter C	$\sigma \in \{2^{-5}, 2^{-4}, \ldots, 2^4, 2^5\}$, $C \in \{10^t, t = -4, -3, \ldots, 3, 4\}$
SC-III	Internal weight parameters **w**, *b*	$(\mathbf{w}, b) \sim U^{d+1}[-\Phi, \Phi]$, $\Phi \in \{1, 5, 15, 30, 50, 100, 150, 200\}$
DMTRL	Factorisation method parameters	{*LAF*, *Tucker*, *TT*}
MMoE	Units *u*, NumExperts *num*	$u \in \{5, 6, 7, \ldots, 19, 20\}$, $num \in \{5, 6, 7, \ldots, 19, 20\}$
AUTOMTL	Weight *LR*, Policy *p*, Decay *d*, Iteration *iter*	$LR = 0.001$, $p = 0.01$, $d = 0.5$, $iter = 400$

will have an improved residual error. Repeat the above steps to add hidden layer nodes until the residual error meets the predetermined error criteria.

The above implementation process of the proposed MTSL-SCRBN algorithm is described as follows.

Algorithm 1. The MTSL-SCRBN algorithm

```
Set maximum number of hidden layer nodes $L_{max}$, maximum tolerance error $\epsilon$,
maximum number of iterations $I_{max}$, maximum number of updates $U_{max}$;
Choose a set of positive scalars $\Pi = \{\Omega_{min} : \Delta\Omega : \Omega_{max}\}$.
1.Initialize $e^{m,0} := [y_1^m, \ldots, y_N^m]^{\mathrm{T}}, (m = 1, \ldots, M), 0 < r < 1$,
        empty sets $\Xi$ and $\Psi$;
2.While $L \le L_{max}$ AND $\sqrt{\frac{1}{N \times M}\sum_{m=1}^{M}\|e^{m,0}\|_2^2} > \epsilon$, Do
3.    For $\Omega \in \Pi$, Do
4.        While $i \le I_{max}$ AND $u \le U_{max}$, Do
5.            Randomly choose $\sigma_L$ from $U[0, \Omega], \Omega > 0$;
6.            Randomly choose $\mathbf{x}^L$ from $\{\mathbf{x}_1^m, \ldots, \mathbf{x}_N^m\}_{m=1}^{M}$;
7.            Set $\mu_{L,r} = \frac{1-r}{L+1}$, calculate $\xi^{m,L}$;
8.            If $\sum_{m=1}^{M} \xi^{m,L} \ge 0$
9.                Save $\sigma_L, \mathbf{x}^L$ in $\Psi$, $\sum_{m=1}^{M} \xi^{m,L}$ in $\Xi$;
10.           Else $i := i + 1$, return to step 5
11.           End If
12.       End While
13.       If $\Xi$ is empty
14.           Randomly choose $\tau$ from $U[0, 1 - r]$,
              update $r := r + \tau, u := u + 1$,
              return to step 5;
15.       Else Find the largest $\sum_{m=1}^{M} \xi^{m,L*}$ from $\Xi$ and
              the corresponding $\sigma_L^*, \mathbf{x}^{L*}$ from $\Psi$,
              calculate $K_L = [\mathbf{k}_L^1, \ldots, \mathbf{k}_L^M]$, set $\mathbf{K}_L = [K_1, \ldots, K_L]$;
16.           Break (return to step 19)
17.       End If
18.   End For
19.   Calculate $\boldsymbol{\beta}^L = [\boldsymbol{\beta}_1^L, \ldots, \boldsymbol{\beta}_L^L]^{\mathrm{T}} := \mathbf{K}_L^{\dagger} Y$;
20.   Calculate $\boldsymbol{e}^L = \mathbf{K}_L \boldsymbol{\beta}^L - Y$;
21.   Update $\boldsymbol{e}^0 := \boldsymbol{e}^L; L := L + 1$;
22.End While
23.Return $\boldsymbol{\beta}_1^L, \ldots, \boldsymbol{\beta}_L^L, \sigma_1^*, \ldots, \sigma_L^*, \mathbf{x}^{1*}, \ldots, \mathbf{x}^{L*}$.
```

Notice that in step 19, we calculate the parameter matrix based on the standard least squares method,

$$\boldsymbol{\beta}^L = \underset{\boldsymbol{\beta} \in R^{LM \times M}}{\arg\min} \left\| Y - \sum_{j=1}^{L} K_j \boldsymbol{\beta}_j \right\|_F^2 = \mathbf{K}_L^{\dagger} Y,$$

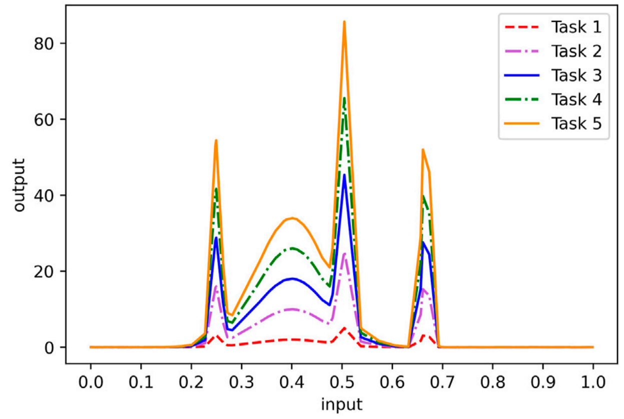

FIGURE 3 | Distribution of the simulation data set.

where $\mathbf{K}_L^{\dagger}$ is the Moore-Penrose generalized inverse (Lancaster and Tismenetsky, 1985) of $\mathbf{K}_L$. The setting of $\mu_{L,r}$ in step 7 and the updating idea of r in step 14 can be referred to literature (Wang and Li, 2017a).

3.2 The Convergence Theorem of the MTSL-SCRBN Algorithm

We extend the method in (Wang and Li, 2017a) to the multi-task learning framework of this paper and prove the convergence of the proposed algorithm.

Theorem 1. Assume that there are some $p_{\mathbf{k}} \in R^+$, satisfying $0 < \|\mathbf{k}_j^m\|_2 < p_{\mathbf{k}}$. Given $0 < r < 1$ and a non-negative real value sequence $\{\mu_L\}$ with $\lim_{L \to +\infty} \mu_L = 0$ and $\mu_L \le (1 - r)$. For $L = 2, 3 \ldots$, denoted by

$$\delta^L = \sum_{m=1}^{M} \delta^{m,L},$$

$$\delta^{m,L} = (1 - r - \mu_L)\|\mathbf{e}^{m,L-1}\|_2^2.$$

If the basis function $\mathbf{k}_L^m$ is generated to satisfy the following inequality,

$$\sum_{m=1}^{M} \langle \mathbf{e}^{m,L-1}, \boldsymbol{k}_L^m \rangle^2 \ge \frac{M^2}{2M - 2} p_{\mathbf{k}}^2 \delta^L, \tag{2}$$

and the external weight parameter vector is evaluated by,

TABLE 2 | The results of MTSL-SCRBN, SC-III and SVM on the simulation data set.

Task	MTSL-SCRBN		SC-III		SVM	
	training	test	training	test	training	test
Task 1	0.1080	**0.2589**	0.1735	0.4264	0.6624	0.8129
Task 2	0.5320	**1.2773**	0.8595	2.1120	3.4796	4.2842
Task 3	0.9589	**2.2985**	1.5501	3.7974	6.0838	7.6899
Task 4	1.3853	**3.3163**	2.2385	5.4810	8.8058	11.1654
Task 5	1.8116	**4.3383**	2.9267	7.1716	11.5199	14.6416

The results with the minimum test errors are marked in bold.

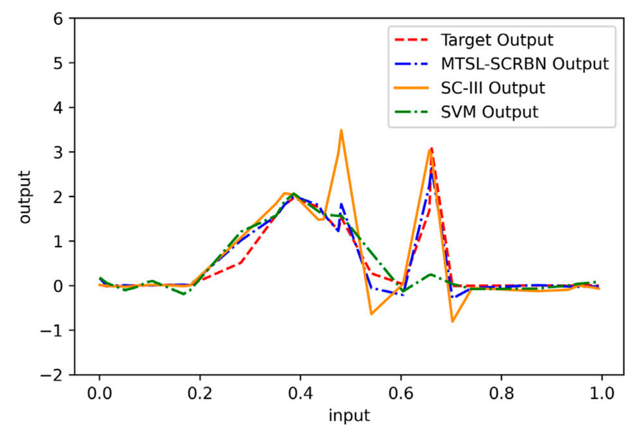

FIGURE 4 | Prediction performance of MTSL-SCRBN, SC-III and SVM on Task 1 of simulation data set.

$$\beta^L = \left[\boldsymbol{\beta}_1^L, \ldots, \boldsymbol{\beta}_L^L\right]^{\mathrm{T}} = \underset{\beta \in R^{LM\times M}}{\arg\min} \left\| Y - \sum_{j=1}^{L} K_j \boldsymbol{\beta}_j \right\|_F^2 .$$

Then, we have $\lim_{L\to+\infty} \|Y - f^L\|_F = 0$.

Proof of Theorem 1. Define intermediate val*ues*

$$\tilde{\beta}_L^{m,L} = \left[\frac{\langle \boldsymbol{e}^{m,L-1}, \boldsymbol{k}_L^1 \rangle}{M\|\boldsymbol{k}_L^1\|_2^2}, \ldots, \frac{\langle \boldsymbol{e}^{m,L-1}, \boldsymbol{k}_L^M \rangle}{M\|\boldsymbol{k}_L^M\|_2^2} \right]^{\mathrm{T}},$$

and $\tilde{\boldsymbol{e}}^L = \boldsymbol{e}^{L-1} - K_L \tilde{\boldsymbol{\beta}}_L^L$, *with* $\tilde{\boldsymbol{\beta}}_L^L = [\tilde{\beta}_L^{1,L}, \ldots, \tilde{\beta}_L^{M,L}], \boldsymbol{e}^0 = Y$.

It is clear that $\|\boldsymbol{e}^L\|_F^2 \le \|\tilde{\boldsymbol{e}}^L\|_F^2 \le \|\boldsymbol{e}^{L-1}\|_F^2 \le \|\tilde{\boldsymbol{e}}^{L-1}\|_F^2$, *where* $L = 2, 3, \ldots$. *So* $\{\|\boldsymbol{e}^{L-1}\|_F^2\}$ *is monotonically decreasing and convergent. Hence, we have,*

$$\begin{aligned}
& \|\boldsymbol{e}^L\|_F^2 - (r+\mu_L)\|\boldsymbol{e}^{L-1}\|_F^2 \\
\le\ & \|\tilde{\boldsymbol{e}}^L\|_F^2 - (r+\mu_L)\|\boldsymbol{e}^{L-1}\|_F^2 \\
=\ & \sum_{m=1}^{M}\left(\langle \boldsymbol{e}^{m,L-1} - K_L\tilde{\beta}_L^{m,L}, \boldsymbol{e}^{m,L-1} - K_L\tilde{\beta}_L^{m,L} \rangle - (r+\mu_L)\langle \boldsymbol{e}^{m,L-1}, \boldsymbol{e}^{m,L-1} \rangle \right) \\
=\ & (1-r-\mu_L)\|\boldsymbol{e}^{L-1}\|_F^2 - \sum_{m=1}^{M}\left(2\langle \boldsymbol{e}^{m,L-1}, K_L\tilde{\beta}_L^{m,L} \rangle - \langle K_L\tilde{\beta}_L^{m,L}, K_L\tilde{\beta}_L^{m,L} \rangle \right) \\
\le\ & (1-r-\mu_L)\|\boldsymbol{e}^{L-1}\|_F^2 - \frac{2M-2}{M^2}\sum_{m=1}^{M}\left(\frac{\langle \boldsymbol{e}^{m,L-1}, \boldsymbol{k}_L^1 \rangle^2}{\|\boldsymbol{k}_L^1\|_2^2} + \cdots + \frac{\langle \boldsymbol{e}^{m,L-1}, \boldsymbol{k}_L^M \rangle^2}{\|\boldsymbol{k}_L^M\|_2^2} \right) \\
\le\ & (1-r-\mu_L)\|\boldsymbol{e}^{L-1}\|_F^2 - \frac{2M-2}{M^2}\sum_{m=1}^{M}\frac{\langle \boldsymbol{e}^{m,L-1}, \boldsymbol{k}_L^m \rangle^2}{\|\boldsymbol{k}_L^m\|_2^2} \\
=\ & \delta^L - \frac{2M-2}{M^2}\sum_{m=1}^{M}\frac{\langle \boldsymbol{e}^{m,L-1}, \boldsymbol{k}_L^m \rangle^2}{\|\boldsymbol{k}_L^m\|_2^2} \\
\le\ & \delta^L - \frac{2M-2}{M^2}\frac{\sum_{m=1}^{M}\langle \boldsymbol{e}^{m,L-1}, \boldsymbol{k}_L^m \rangle^2}{p_{\boldsymbol{k}}^2} \\
\le\ & 0.
\end{aligned}$$

Then, the following inequality holds,

$$\|\boldsymbol{e}^L\|_F^2 \le r\|\boldsymbol{e}^{L-1}\|_F^2 + \gamma_L, \left(\gamma_L = \mu_L\|\boldsymbol{e}^{L-1}\|_F^2 \ge 0\right).$$

Since $\lim_{L\to+\infty}\mu_L = 0$, *and* $0 < r < 1$, *we have* $\lim_{L\to+\infty}\|\boldsymbol{e}^L\|_F^2 = 0$, *and* $\lim_{L\to+\infty}\|\boldsymbol{e}^L\|_F = 0$.

Remark 1. Unl*ike SC-III, we relax the condition for the configuration parameters in the formula* (*). *SC-III requires each task to meet the inequality conditions, but MTSL-SCRBN only requires the sum of all tasks to satisfy the inequality condition. The rationality of this condition will also be verified in the experiment results of next section.*

4 EXPERIMENT RESULTS

In order to show the effectiveness of the proposed algorithm, this section uses the classical STSL algorithms SVM (Cortes and Vapnik, 1995), SC-III (Wang and Li, 2017a) and seven MTSL algorithms MTSL-SCRBN, MTL (Caruana, 1997), MTEN (Chen et al., 2012), DMTRL (Yang and Hospedales, 2017), MMoE (Ma et al., 2018), GAMTL (Oliveira et al., 2019), AUTOMTL (Zhang et al., 2021) to perform comparative experiments. All calculations are conducted using Python 3.6.5 on a computer with 2.60 GHz CPU and 8 GB RAM. The input features are scaled into $[-1, 1]$ and the output remains unchanged. All the results reported in this paper take averages over 20 independent trials, except for the SVM and MTEN algorithms, which have fixed experiment results. The accuracy (ACC) and root mean square error (RMSE) are chosen as the classification and regression evaluation indicators, where

$$RMSE = \left[\frac{1}{N\times M} \sum_{m=1}^{M}\sum_{i=1}^{N} \left(\hat{y}_i^m - y_i^m\right)^2 \right]^{\frac{1}{2}},$$

with y_i^m and $\hat{y}_i^m$ representing the target output and the learner's output of $i-$ th sample for task m respectively.

For different data sets, some algorithms used in the following experiments can stochastically configure hyperparameters within specified ranges or determine parameters by cross-validation. **Table 1** gives the specific selection range of each parameter.

4.1 Experimental Results and Analysis on Simulated Data

The simulation data set selected in this paper is generated by the following five functions, which we refer as five tasks,

TABLE 3 | The results of MTSL-SCRBN, MTL, MTEN on the simulation data set.

MTSL-SCRBN		MTL		MTEN	
training	test	training	test	training	test
1.2836 (0.1763)	**2.6383** (0.0929)	6.7430 (0.0336)	5.1472 (0.0187)	7.8274	6.2219

The results with the minimum test errors are marked in bold.

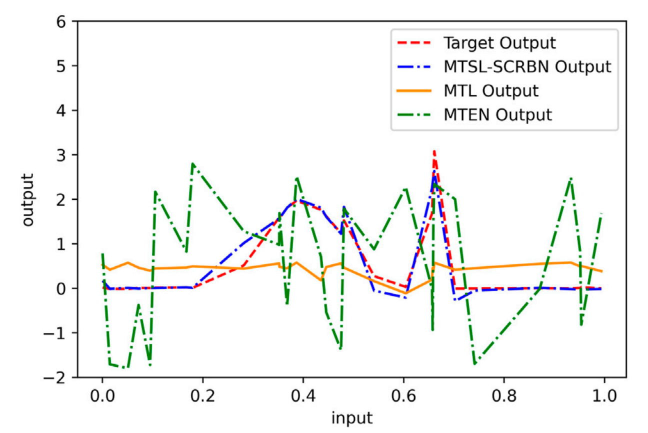

FIGURE 5 | Prediction performance of MTSL-SCRBN, MTL, MTEN on Task 1 of simulation data set.

$$
\begin{aligned}
&\text{Task 1: } f_1(x) = 2e^{-(10x-4)^2} + 5e^{-(80x-40)^2} + 3e^{-(80x-20)^2} + 4e^{-(90x-60)^2}\\
&\text{Task 2: } f_2(x) = 10e^{-(10x-4)^2} + 25e^{-(80x-40)^2} + 15e^{-(80x-20)^2} + 20e^{-(90x-60)^2}\\
&\text{Task 3: } f_3(x) = 18e^{-(10x-4)^2} + 45e^{-(80x-40)^2} + 27e^{-(80x-20)^2} + 36e^{-(90x-60)^2}.\\
&\text{Task 4: } f_4(x) = 26e^{-(10x-4)^2} + 65e^{-(80x-40)^2} + 39e^{-(80x-20)^2} + 52e^{-(90x-60)^2}\\
&\text{Task 5: } f_5(x) = 34e^{-(10x-4)^2} + 85e^{-(80x-40)^2} + 51e^{-(80x-20)^2} + 68e^{-(90x-60)^2}
\end{aligned}
$$

Figure 3 depicts the distributions of five functions on [0, 1]. As we can see, when the independent variables of the five functions are the same, the function values follow similar trends. Therefore, learning the function values of five functions with the same independent variable can be regarded as a multi-task learning. Here, we independently extract 100 one-dimensional input data from the same uniform distribution, then calculate the corresponding function values according to these five functions, and add white Gaussian noise with a standard deviation of 0.01 to form 100 five-dimensional output data. In the following experiments, we randomly select 70% of the data as training data and 30% of the data as test data. From the figures of the five functions, it can be seen that only 70 training samples are not enough to achieve good single-task learning results. We verify this point by experimenting with single-task and multi-task algorithms.

Firstly, we compare the learning performance of the proposed MTSL-SCRBN with other two STSL algorithms, SVM and SC-III, on five tasks. For MTSL-SCRBN, these five tasks are combined to learn together. The training and test RMSEs on five tasks for these three methods are given in **Table 2**. Clearly, the proposed multi-task learning model can product better performance on each task than the two STSL models, which only use 70 samples to learn each task independently. Furthermore, we show the learning effects of the three algorithms on Task 1 in **Figure 4**. It is can be seen that the proposed MTSL-SCRBN has good learning performance where the data changes dramatically.

Next, the comparison results of three MTSL algorithms, MTSL-SCRBN, MTL, MTEN, on the simulation data set are recorded in **Table 3** and **Figure 5**. In **Table 3**, the values in parentheses represent the standard deviations of 20 experiments' results. According to these results, compared with MTEN and MTL, the proposed MTSL-SCRBN has better approximation ability.

4.2 Experimental Results and Analysis on Benchmark Datasets

This subsection further compares seven MTSL algorithms on four benchmark datasets. They are MTL, MTEN, DMTRL, MMoE, GAMTL, AUTOMTL and the proposed MTSL-SCRBN. According to the characteristics of data sets and algorithms, different algorithms will be selected for comparative analysis on different data sets. The four benchmark datasets include three regression problems on the stock portfolio performance data set, the bionic robot data set SARCOS and the School data set, one classification problem on the Mnist data set from Yann LeCun[1] The basic information of the four datasets are summarized in **Table 4**.

Firstly, we compare the performance of MTL, MTEN, DMTRL, MMoE, GAMTL and MTSL-SCRBN on different sizes of three regression data sets. For the three data sets, we randomly choose 15/30/3,500 samples outside the training set as test set, respectively. At the same time, we select 5 tasks, all of which have more than 230 samples, from 139 tasks in School data set. **Table 5** below shows the specific experiment results. As we can see, the performance of each algorithm tends to be better and more stable with an increasing number of training samples. Furthermore, the proposed MTSL-SCRBN algorithm exhibits good performance even with a small number of training observations.

Then, in order to further verify the performance of MTSL-SCRBN for classification cases, we compare the results of MTSL-SCRBN, DMTRL, MMoE, GAMTL and AUTOMTL on the Mnist data set. We randomly choose 50/100/150 samples from each task in the Mnist data set as training set, and 10000 samples in the remaining samples as test set. Considering that this is a high

[1] These four datasets can be obtained from http://archive.ics.uci.edu/ml/datasets/Stock+portfolio+performance, http://gaussianprocess.org/gpml/data, http://bristol.ac.uk/cmm/learning/support/datasets and http://yann.lecun.com/exdb/mnist/, respectively.

TABLE 4 | Descriptions of benchmark datasets.

Data Set	Size	Feature Number	Task Number
Stock	63	6	6
SARCOS	48933	21	7
School	15362	8	139
Mnist	70000	28*28	10

TABLE 5 | The comparison results of six MSTL algorithms on three data sets.

Data set	Size	MTSL-SCRBN		MTL		MTEN		DMTRL		MMoE		GAMTL	
		training	test	training	test	training	test	training	test	training	test	training	test
Stock		0.0796	**0.1105**	0.1524	0.3016			0.1155	0.1397	0.1441	0.1649	0.0885	0.1536
	10	(0.0039)	(0.0040)	(0.0619)	(0.0667)	0.1370	0.1387	(0.0268)	(0.0089)	(0.0045)	(0.0040)	(0.0072)	(0.0018)
		0.0661	**0.0904**	0.1149	0.2207		0.0973	0.1008	0.1362	0.1577	0.0811	0.1342	
	20	(0.0050)	(0.0069)	(0.0158)	(0.0754)	0.1313	0.1379	(0.0188)	(0.0112)	(0.0091)	(0.0057)	(0.0074)	(0.0065)
		0.0542	**0.0793**	0.1055	0.1712		0.0839	0.0975	0.1282	0.1505	0.0769	0.1186	
	30	(0.0054)	(0.0078)	(0.0117)	(0.0366)	0.1380	0.1339	(0.0156)	(0.0078)	(0.0019)	(0.0010)	(0.0033)	(0.0022)
Sarcos		2.3921	**3.6667**	5.1595	5.9636		4.8021	5.4138	3.5679	4.3704	3.6894	4.5503	
	700	(0.0221)	(0.0364)	(0.0395)	(0.0895)	4.2165	4.2680	(0.1312)	(0.1128)	(0.0165)	(0.0112)	(0.0315)	(0.0489)
		2.2732	**3.1740**	4.7317	5.6063		4.2588	4.7137	2.7236	3.4125	2.9128	3.6810	
	1,400	(0.0250)	(0.0179)	(0.0342)	(0.0808)	4.1334	4.2281	(0.0610)	(0.1084)	(0.0100)	(0.0101)	(0.0147)	(0.0286)
		2.1128	**2.9291**	4.5611	5.0693		3.4181	4.0180	2.6235	2.9898	2.7503	3.0137	
	2,100	(0.0255)	(0.0392)	(0.0370)	(0.0899)	4.1552	4.1818	(0.0683)	(0.0639)	(0.0115)	(0.0123)	(0.0239)	(0.0317)
School		8.9122	**12.1368**	11.9488	13.6566		12.0246	13.2983	11.9107	13.5805	11.7439	12.3742	
	100	(0.1221)	(0.0634)	(0.1356)	(0.1284)	12.1952	13.9454	(0.1411)	(0.1155)	(0.1286)	(0.2422)	(0.1280)	(0.1560)
		8.4406	**11.5972**	11.4453	13.3710		11.7838	13.1285	11.6480	13.2517	11.3313	11.7067	
	150	(0.0736)	(0.0828)	(0.1141)	(0.1089)	11.7394	13.5306	(0.1562)	(0.1371)	(0.0907)	(0.1108)	(0.0899)	(0.0249)
		7.6262	**10.6699**	11.1923	12.9307		11.4090	12.9873	11.3628	12.9779	10.8133	11.3346	
	200	(0.1177)	(0.1067)	(0.0874)	(0.1145)	11.3253	12.8915	(0.1589)	(0.2015)	(0.0748)	(0.0612)	(0.0546)	(0.0307)

The results with the minimum test errors are marked in bold.

TABLE 6 | The accuracy of MTSL-SCRBN, DMTRL, MMoE, GAMTL and AUTOMTL on Mnist data set.

Size	MTSL-SCRBN		DMTRL		MMoE		GAMTL		AUTOMTL	
	training	test	training	test	training	test	training	test	training	test
	92.00%	**67.14%**	91.80.00%	64.67%	94.16%	62.22%	69.47%	57.01%	93.26%	60.78%
50	(0.0065)	(0.0067)	(0.0111)	(0.0099)	(0.0077)	(0.0086)	(0.0303)	(0.0124)	(0.0231)	(0.0212)
	96.15%	**72.25%**	95.90%	70.793%	95.92%	68.15%	84.62%	68.51%	94.56%	70.45%
100	(0.0067)	(0.0066)	(0.0117)	(0.0101)	(0.0084)	(0.0061)	(0.0144)	(0.0158)	(0.0128)	(0.0094)
	97.27%	**82.49%**	97.97%	80.05%	97.04%	74.61%	95.52%	78.89%	96.87%	79.34%
150	(0.0099)	(0.0064)	(0.0082)	(0.0097)	(0.0081)	(0.0063)	(0.0119)	(0.0087)	(0.0097)	(0.0084)

The results with the minimum test errors are marked in bold.

TABLE 7 | Parameter description for the four models.

Models	Parameters	Parameters' Range
MTSL-SCRBN	RBF scale σ	$\sigma \sim U[0, \Omega]$, $\Omega \in (0, 100]$
MTSL-SCSGM	Internal weight parameters $\mathbf{w}$, b,	$(\mathbf{w}, b) \in \sim U^{d+1}[-\Phi, \Phi]$, $\Phi \in \{1, 5, 15, 30, 50, 100, 150, 200\}$
MTSL-SCTANH	Internal weight parameters $\mathbf{w}$, b,	$(\mathbf{w}, b) \in \sim U^{d+1}[-\Phi, \Phi]$, $\Phi \in \{1, 5, 15, 30, 50, 100, 150, 200\}$
MTSL-SCReLU	Internal weight parameters $\mathbf{w}$, b	$(\mathbf{w}, b) \sim U^{d+1}[-\Phi, \Phi]$, $\Phi \in \{1, 5, 15, 30, 50, 100, 150, 200\}$

TABLE 8 | The comparison results of different activation functions based models on two data sets.

Data set	Size	MTSL-SCRBN		MTSL-SCSGM		MTSL-SCTANH		MTSL-SCReLU	
		training	test	training	test	training	test	training	test
Stock	10	0.0796 (0.0039)	**0.1105** (0.0040)	0.0819 (0.0048)	0.1566 (0.0057)	0.0814 (0.0036)	0.1586 (0.0036)	0.0853 (0.0035)	0.1424 (0.0055)
	20	0.0661 (0.0050)	**0.0904** (0.0069)	0.0766 (0.0032)	0.1231 (0.0098)	0.0758 (0.0034)	0.1242 (0.0076)	0.0759 (0.0028)	0.1194 (0.0051)
	30	0.0542 (0.0054)	**0.0793** (0.0078)	0.0671 (0.0020)	0.1128 (0.0061)	0.0665 (0.0021)	0.1037 (0.0076)	0.0678 (0.0012)	0.1009 (0.0046)
Sarcos	700	2.3921 (0.0221)	**3.6667** (0.0364)	3.3670 (0.0263)	5.1386 (0.0868)	3.2588 (0.0308)	4.9160 (0.0746)	3.1868 (0.0140)	5.5491 (0.0824)
	1,400	2.2732 (0.0250)	**3.1740** (0.0179)	3.1829 (0.0161)	4.0958 (0.0713)	3.0747 (0.0178)	4.1529 (0.0598)	2.9880 (0.0112)	4.1854 (0.0767)
	2,100	2.1128 (0.0255)	**2.9291** (0.0392)	2.9860 (0.0136)	3.8698 (0.0764)	2.8853 (0.0127)	3.8420 (0.0494)	2.8539 (0.0705)	3.8211 (0.0506)

The results with the minimum test errors are marked in bold.

dimensional small sample problem, we firstly reduce the dimensionality of the data set, and then use MTSL-SCRBN for training and prediction. There are many dimensionality reduction methods, such as Principal Component Analysis(PCA) (Pearson, 1901), Latent Dirichlet Allocation(LDA) (Blei et al., 2003), Sequential Markov Blanket Criterion (SMBC) (Pratama et al., 2017), Auto Encoder (Hinton and Salakhutdinov, 2006) and so on. Here, we use the performance of the dimensionality-reduced data in the MTSL-SCRBN as the selection criterion, and choose Auto Encoder to reduce the dimension of original data set into 30 dimensions. It can be seen from **Table 6** that the performance of MTSL-SCRBN which uses dimensionality reduction data set is a little bit better than that of other Multi-task deep learning algorithms, but as the sample size increases, the performance of the two algorithms gradually approaches.

4.3 Comparison Experiment Results for Different Activation Functions

The previous results show that the proposed MTSL-SCRBN algorithm is effective for multi-task learning in the case of small samples. This subsection mainly discusses the impact of selecting different activation functions on algorithm performance. Here we select other three usually used activation functions. They are sigmoid function, Tanh function and ReLU function. After replacing the radial basis functions in MTSL-SCRBN with these three functions respectively, the model names are respectively called MTSL-SCSGM, MTSL-SCTANH and MTSL-SCReLU. We choose to conduct comparative experiments on the Stock and SARCOS data sets.

For different data sets, the parameters contained in each algorithm need to be randomly set or cross verified within a certain range. The specific selection range of each parameter is given in **Table 7**.

Table 8 depicts the RMSE results of stochastic configuration multi-task learning models based on four different activation functions. Under the training samples with different sample sizes in the two data sets, the MTSL-SCRBN, which based on radial basis functions, has certain advantages over other three models in terms of performance.

5 CONCLUSION

In this paper, we propose a multi-task supervised learning framework based on stochastic configuration radial basis network. It can be effectively used in classification and regression problems when a single task has a small number of samples. The series experiment results on the four data sets show the proposed MTSL-SCRBN achieves a good performance compared with some existing methods.

Interesting areas for further directions include using the proposed algorithm in hyperspectral remote sensing image classification and other related research areas, considering the impact of using different activation functions in the network, and trying to explore the range of the sample size of the data set to use the multi-task learning method.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/Supplementary Material, further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

X-MD: Conceptualization, Methodology, Formal analysis, Supervision, Project administration. XK: Writing-original draft, Software, Investigation. XZ: Validation, Writing—review and editing.

FUNDING

This work was supported in part by the Characteristic and Preponderant Discipline of Key Construction Universities in Zhejiang Province (Zhejiang Gongshang University- Statistics) and the special fund for basic business expenses of colleges and universities in Zhejiang Province.

REFERENCES

Argyriou, A., Evgeniou, T., and Pontil, M. (2008). Convex Multi-Task Feature Learning. *Mach. Learn* 73, 243–272. doi:10.1007/s10994-007-5040-8

Argyriou, A., Evgeniou, T., and Pontil, M. (2007). “Multi-task Feature Learning,” in *Advances in Neural Information Processing Systems*, 41–48. doi:10.7551/mitpress/7503.003.0010

Baxter, J. (2000). A Model of Inductive Bias Learning. *Jair* 12, 149–198. doi:10.1613/jair.731

Ben-David, S., and Schuller, R. (2003). “Exploiting Task Relatedness for Multiple Task Learning,” in *Learning Theory and Kernel Machines* (Springer), 567–580. doi:10.1007/978-3-540-45167-9_41

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *J. Mach. Learn. Res.* 3, 993–1022. doi:10.1162/jmlr.2003.3.4-5.993

Broomhead, D., and Lowe, D. (1988). *Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Networks*. Malvern, United Kingdom: Royal Signals and Radar Establishment Malvern United Kingdom. Tech. rep.

Caruana, R. (1997). Multitask Learning. *Mach. Learn.* 28, 41–75. doi:10.1023/a:1007379606734

Chen, X., He, J., Lawrence, R., and Carbonell, J. (2012). “Adaptive Multi-Task Sparse Learning with an Application to Fmri Study,” in *Proceedings of the 2012 SIAM International Conference on Data Mining*, 212–223. doi:10.1137/1.9781611972825.19

Cortes, C., and Vapnik, V. (1995). Support-vector Networks. *Mach. Learn* 20, 273–297. doi:10.1007/BF00994018

Duong, L., Cohn, T., Bird, S., and 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*, 2, 845–850. Short Papers. doi:10.3115/v1/p15-2139

Hinton, G. E., and Salakhutdinov, R. R. (2006). Reducing the Dimensionality of Data with Neural Networks. *Science* 313, 504–507. doi:10.1126/science.1127647

Jacob, L., Vert, J., and Bach, F. (2009). “Clustered Multi-Task Learning: A Convex Formulation,” in *Advances in Neural Information Processing Systems*, 745–752.

Lancaster, P., and Tismenetsky, M. (1985). *The Theory of Matrices: With Applications*. Elsevier.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based Learning Applied to Document Recognition. *Proc. IEEE* 86, 2278–2324. doi:10.1109/5.726791

Liu, T., Tao, D., Song, M., and Maybank, S. J. (2017). Algorithm-dependent Generalization Bounds for Multi-Task Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* 39, 227–241. doi:10.1109/TPAMI.2016.2544314

Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., and Chi, E. (2018). “Modeling Task Relationships in Multi-Task Learning with Multi-Gate Mixture-Of-Experts,” in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom: Association for Computing Machinery), 1930–1939. doi:10.1145/3219819.3220007

Oliveira, S. H. G., Gonçalves, A. R., and Von Zuben, F. J. (2019). “Group Lasso with Asymmetric Structure Estimation for Multi-Task Learning,” in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19* (Macao, China: International Joint Conferences on Artificial Intelligence Organization), 3202–3208. doi:10.24963/ijcai.2019/444

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *Lond. Edinb. Dublin Philosophical Mag. J. Sci.* 2, 559–572. doi:10.1080/14786440109462720

Powell, M. (1987). *Radial Basis Functions for Multivariable Interpolation: A Review*. Oxford, United Kingdom: Algorithms for approximation, 143–167.

Pratama, M., Lu, J., Lughofer, E., Zhang, G., and Er, M. J. (2017). An incremental learning of concept drifts using evolving type-2 recurrent fuzzy neural networks. *IEEE Trans. Fuzzy Syst.* 25, 1175–1192. doi:10.1109/tfuzz.2016.2599855

Tin-Yan Kwok, T., and Dit-Yan Yeung, D. (1997). Objective functions for training new hidden units in constructive neural networks. *IEEE Trans. Neural Netw.* 8, 1131–1148. doi:10.1109/72.623214

Wang, D., and Li, M. (2017). Robust stochastic configuration networks with kernel density estimation for uncertain data regression. *Inf. Sci.* 412-413, 210–222. doi:10.1016/j.ins.2017.05.047

Wang, D., and Li, M. (2017). Stochastic configuration networks: Fundamentals and algorithms. *IEEE Trans. Cybern.* 47, 3466–3479. doi:10.1109/tcyb.2017.2734043

Yang, Y., and Hospedales, T. (2017). “Deep multi-task representation learning: A tensor factorisation approach,” In International Conference on Learning Representations.

Zhang, L., Liu, X., and Guan, H. (2021). *Automtl: A Programming Framework for Automated Multi-Task Learning*, 13076. ArXiv abs/2110.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.