Modeling Task Relationships in Multivariate Soft Sensor With Balanced Mixture-of-Experts

Yuxin Huang, Eric Hao Wang, Zhaoran Liu, Licheng Pan, Haozhe Li, and Xinggao Liu

***Abstract*—Accurate estimation of multiple quality variables is critical for building industrial soft sensor models, which have long been confronted with data efficiency and negative transfer issues. Methods sharing backbone parameters among tasks address the data efficiency issue; however, they still fail to mitigate the negative transfer problem. To address this issue, a balanced mixture-of-experts (BMoE) is proposed in this work, which consists of a multigate mixture-of-experts module and a task gradient balancing (TGB) module. The mixture-of-experts module aims to portray task relationships, while the TGB module balances the gradients among tasks dynamically. Both of them cooperate to mitigate the negative transfer problem. Experiments on the typical sulfur recovery unit demonstrate that BMoE models task relationship and balances the training process effectively, and achieves better performance than baseline models significantly.**

***Index Terms*—Deep learning, multitask learning (MTL), soft sensor.**

I. Introduction

PROCESS industry plays an important role in modern industry and is closely related to key industrial manufacturing such as oil, gas, rare metals, iron, and steel, which forms an integral part of modern human life and national economies. Monitoring dynamic changes of components precisely in the process industry has become one of the main concerns in order to meet urgent but harsh demands such as increasing production, reducing materials consumption, protecting environments, and ensuring safety in manufacturing process. However, in process engineering, there are many difficult-to-measure variables that are critical to evaluate process quality. For example, in deep water gas-lift oil well process, downhole pressure carries useful information for the oil refinery process of oil field [1]. The pressure is hard to be consistently measured by hardware sensors like permanent downhole gauges because the measurement will break down under high pressure and salinity environment [2].

Soft sensors, which aim to measure these variables in an indirect manner, can be grouped into model-driven and data-driven methods. With the development of machine learning and database technologies, data-driven approaches dominate the field of soft sensors, which construct statistical estimands of quality variables with measurable process variables [3]. Classical linear statistical models were first applied to soft sensors, such as principal component regression and partial least squares (PLS) [4]. To depict nonlinear relationships between variables, soft sensors based on advanced machine learning were further witnessed, such as support vector regression [5] and extreme learning machine [6]. For example, Zhang et al. [7] proposed a double-level locally weighted extreme learning machine based soft sensor for online prediction of quality variables, which shows better performance compared to conventional statistical methods.

More recently, researchers have turned their attention to deep learning-based soft sensors [8], which map raw data into low-dimensional semantic space to extract spatial or temporal characteristics. Specifically, recurrent neural networks and temporal convolutional networks [9] have been deeply applied to depict temporal dependencies; convolutional neural networks have been widely used to depict spatial dependencies [10], [11]. Autoencoders (AEs) [12] have been widely used to depict dependencies among process variables. For example, Yuan et al. [13] proposed variable-wise weighted stacked AE to acquire output-related representations. Sun et al. [14] proposed gated stacked autoencoder (GSAE), which utilized multiscale representations from multiple layers.

Although accurate soft sensors for the individual quality variable have been achieved, there are always multiple quality variables that need to be measured simultaneously in the industrial process, namely multivariate soft sensor (MVSS). For example, to monitor the separation energy consumption and product purity in reactive distillation process, the product concentration and reactant concentration require to be measured simultaneously. Approaches mentioned above, which model each quality variable independently, ignore the correlation between quality variables and, thus, utilize data in an inefficient way.

Multitask learning has long been used to enhance the data efficiency of neural networks [15]. According to the parameter sharing strategy, it can be categorized into two groups. Methods

Manuscript received 28 April 2022; revised 25 July 2022; accepted 11 August 2022. Date of publication 30 August 2022; date of current version 4 May 2023. This work was supported in part by the National Key R&D Program of China under Grant 2021YFC2101100, in part by the National Natural Science Foundation of China under Grants 62073288, 12075212, 12105246, and 11975207, in part by the Provincial Key R&D Program of Zhejiang under Grants 2020C01038 and 2021C01032 and in part by the and Zhejiang University NGICS Platform and their supports are thereby acknowledged. Paper no. TII-22-1807. *(Corresponding author: Xinggao Liu.)*

The authors are with the State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: horthy123@163.com; 22032130@zju.edu.cn; 22032057@zju.edu.cn; 22132045@zju.edu.cn; lihaozhe@zju.edu.cn; lxg@zju.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2022.3202909.

Digital Object Identifier 10.1109/TII.2022.3202909

1551-3203 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR. Downloaded on January 22,2024 at 04:57:23 UTC from IEEE Xplore. Restrictions apply.

in the first group share parameters in a hard approach, such as UberNet [16] and multilinear relationship networks [17]. This strategy does enhance data efficiency; however, the task relationship is not depicted, suffering from the risk of negative transfer between conflicting tasks. Methods in the second group share parameters in a soft approach, such as cross-stitch networks [18], Sluice networks [19], and multi-task adversarial network (MTAN) [20], which alleviate negative transfer by depicting tasks relationships. Another concern of multitask soft sensor is the seesaw phenomenon proposed in [21]. Specifically, in the joint optimization procedure, the optimizer always focuses on the dominant task at the expense of other tasks, making the training process unbalanced.

As such, both lack of task relationship and the seesaw problem introduce negative transfer, thus, hindering the performance of multitask learning (MTL) models. To alleviate the two problems simultaneously, a balanced mixture-of-experts (BMoE) is developed in this work, which consists of a multigate mixture-of-experts (MMoE) module and a task gradient balancing (TGB) module. Specifically, the MMoE module finds transferable representations by depicting task relationships; the TGB module addresses the seesaw problem by tuning the gradient magnitudes dynamically. Both of them cooperate to mitigate the negative transfer in MVSS.

The contributions of this article are summarized as follows.

1) A MVSS problem is proposed, which aims to estimate multiple quality variables simultaneously. The primary challenge is the negative transfer between quality variables, which is mainly caused by the nontransferable features and seesaw problem.
2) A BMoE model is proposed to address the negative transfer in MVSS, based on a generalized MMoE module and TGB module. Specifically, the MMoE module finds transferable representations by depicting task relationships; and the TGB module addresses the seesaw problem by normalizing the gradient magnitudes among tasks. Both corporate to address the negative transfer.
3) Extensive experiments are conducted in the sulfur recovery unit process, where BMoE finds the transferable representations efficiently, addresses the seesaw problem effectively, and improves the sensor quality significantly.

The rest of this article is organized as follows. Section II presents preliminaries on MoE structure and the GradNorm algorithm. Section III proposes the novel BMoE approach. Section IV shows experiments on famous case studies, the sulfur recovery unit. Finally, Section V concludes the article.

II. Preliminaries

A. MTL Models

MTL aims to improve model generalization by leveraging training signals of multiple tasks. The performance would be improved if the associated tasks share complementary information. Deep multitask architectures can be categorized into two groups with respect to parameter sharing strategy. In the hard-sharing group, models typically consist of one hard-shared encoder followed by several task-specific heads [see Fig. 1(a)], which means the hard-shared encoder is forced to be utilized by all different task-specific heads; in the soft-sharing group, the hard-shared encoder is replaced by a feature sharing mechanism that handles the cross-task talk [see Fig. 1(b)]. In that way, parameters of the bottom networks are encouraged to be close by feature sharing mechanisms like loss function, but not forced to be same.

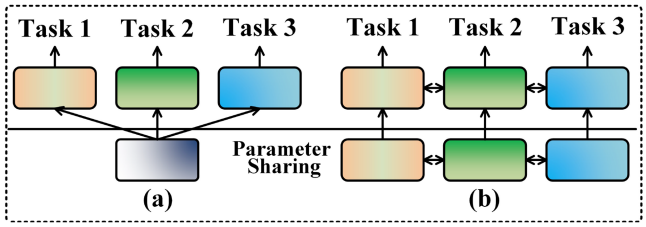

Fig. 1. Structures of MTL models. (a) Hard-share. (b) Soft-share.

B. Positive and Negative Transfer

Given N tasks, the loss function of the MTL model can be given as shown in (1), where $\mathcal{B}$ and $\mathcal{A}$ are the share module and specific module, respectively. The learning process seeks to find an MTL model as minimizing (1) over $\mathcal{B}$ and the $\mathcal{A}_i$s

$$f(\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_N; \mathcal{B}) = \sum_{i=1}^{N} L(g(x_i, \mathcal{B})\mathcal{A}_i, y_i). \quad (1)$$

The phenomenon named positive transfer is defined as the model training that through (1) improves over just training the specific task. On the contrary, the negative transfer suggests the training through (1) cannot improve over just training the specific task. In practice, the quality variables of the chemical process may occur the negative transfer phenomenon in the model training. Take the reaction process as an example. To better monitor the reaction, the main reaction product content and the side reaction product are always bound to be monitored simultaneously. However, the product contents measurement tasks will be tradeoff results from the selectivity and the capacity. This task indicates that the design of the model should consider the negative transfer phenomenon.

C. Mixture of Expert

Fig. 2(a) presents the structure of the mixture-of-experts (MoE), and (2) gives the expression for the MoE

$$o = \sum_{i=1}^{K} g_i(x) \times f_i(x) \quad (2)$$

where the o stands for the output of the model; the $f_i(x)$ indicates the output of the corresponding expert; and the $g_i(x)$ stands for the weight obtained from the gating network. Note that the gating network is designed for calculating the distribution of the corresponding expert, and as such the $g_i(x)$ is ought to satisfy the corresponding constraint

$$\sum_{i=1}^{K} g_i(x) = 1. \quad (3)$$

Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR. Downloaded on January 22,2024 at 04:57:23 UTC from IEEE Xplore. Restrictions apply.

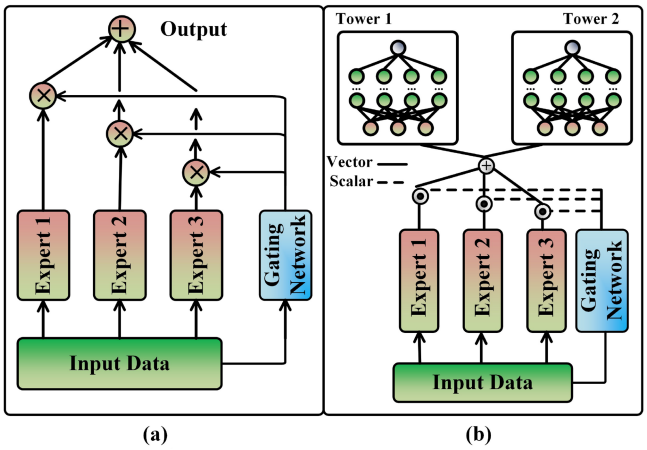

Fig. 2. Structures of (a) mixture-of-experts and (b) multigate mixture-of-experts.

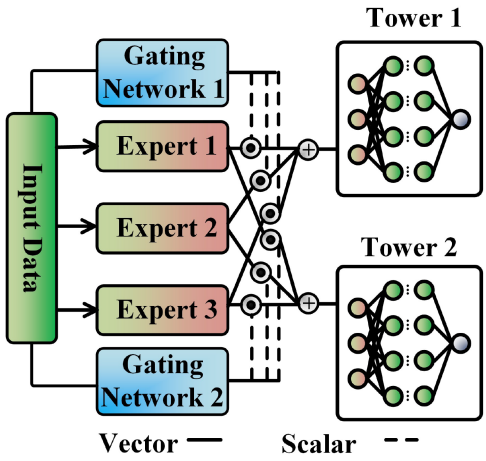

Fig. 3. Structures of multigate mixture-of-experts.

Moreover, as shown in Fig. 2(b), the MoE layer can be introduced into the MTL framework from the perspective of selecting the subset of the expert networks. In one-gated MoE model, the single gating network gives weights of the weighted sum to all expert networks' outputs. On this basis, the weighted sum of features from different experts are then fed to the corresponding networks (known as towers) for specific tasks, which output the prediction results. By adding the MoE layer in the MTL framework, the expert feature can be dynamically weighted for different tasks varying from samples. As such, the negative transfer phenomenon can be alleviated in a way.

III. Proposed Method

A. Problem Statement

Given a size-M set of observed history process variables $[x_1, x_2, \ldots, x_M] \in \mathbb{R}^{M\times F}$, and the corresponding labels corresponding to size-N task $[y_1^n, y_2^n, \ldots, y_M^n] \in \mathbb{R}^{M\times N}$. The task is to predict the labels $[y_{M+1}^n, y_{M+2}^n, \ldots, y_{M+H}^n] \in \mathbb{R}^{H\times N}$ with known process variables $[x_M + 1, x_M + 2, \ldots, x_{M+H}] \in \mathbb{R}^{M\times H}$.

Proposed method is designed based on the following concerns.

1) The backbone architecture needs to be compact and able to extract the abstract semantics beyond the input data.
2) To mitigate the negative transfer problem, the model ought to depict the task relationships and balance the training procedure.

B. MMoE Module

The one-gated MoE network does not model the difference between tasks explicitly, and the negative transfer issue therefore remains unsolved. To this end, inspired by [22], the MMoE model is designed in the proposed framework. As shown in Fig. 3, raw data are firstly fed into an embedding layer as

$$z = f(x; \theta) \tag{4}$$

where θ is the embedding look-up table hardly shared across tasks. Afterward, K expert networks $\phi_{\mathrm{k}}, k = 1, 2, \ldots, K$ are utilized to extract representations that are softly shared across tasks. Then, to estimate the ith quality variable, the experts' outputs are weighted and summed using the gating networks

$$\psi_{\mathrm{i}} = \sum_{\mathrm{k}=1}^{K} g_{\mathrm{i}}^{\mathrm{k}}(z)_{\mathrm{k}} \phi_{\mathrm{k}}(z), \quad i = 1, 2, \ldots, N \tag{5}$$

where $g_{\mathrm{i}}^{\mathrm{k}}(z)$ denotes the gating network of the kth expert to estimate the ith quality variable. Finally, the estimate of the ith quality variable is given as

$$\widehat{y}_{\mathrm{i}} = \Phi_{\mathrm{i}}(\psi_{\mathrm{i}}), \quad i = 1, 2, \ldots, N \tag{6}$$

where Φ_{i} denotes the task-specific tower function. Leveraging the task-specific tower functions and softly shared experts, task-specific features are extracted and the negative transfer is remitted.

Even though the structure of the gating network in the MMoE is similar to the MoE model, there still remain differences. Particularly, MoE model equips one gate merely for different tasks, which indicates that the gate needs to serve all the relevant towers, while the MMoE model equips each task with a gating network. On this basis, the gating network would activate the best expert for the specific variable, and generate distributions varying the tasks. To this extent, the negative transfer phenomenon can be mitigated in a way.

C. Gradient Balance Module

Another challenge to MVSSs is the risk of imbalanced training. Specifically, training shared parameters in multivariate prediction structure is not always balanced for different tasks, with the optimizer overly focusing on the dominant task at the expense of other tasks. To this end, the gradient balance module [23] is designed to balance the gradient magnitudes by adjusting the weights of tasks dynamically.

Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR. Downloaded on January 22,2024 at 04:57:23 UTC from IEEE Xplore. Restrictions apply.

Generally, the loss function for MVSS is

$$L = \sum_{\mathrm{i}=1}^{N} w_{\mathrm{i}} L_{\mathrm{i}}(t) \tag{7}$$

where $L_{\mathrm{i}}(t)$ is the prediction loss of the ith quality variable at the time t, and w_{i} is the corresponding weight. However, the fixed weights are hard to decide, requiring dedicated parameter tuning, especially when the number of quality variables increases. Meanwhile, the unchangeable weight tends to make the training process suffer from training imbalance. Therefore, the dynamic weights that fluctuate over time are introduced

$$L(t) = \sum_{\mathrm{i}=1}^{N} w_{\mathrm{i}}(t) L_{\mathrm{i}}(t). \tag{8}$$

Let W be the parameters that are hardly shared across tasks, i.e., the embedding layer in (4). The gradient's norm of $w_{\mathrm{i}}(t)L_{\mathrm{i}}(t)$ with respect to W is

$$G_W^{(\mathrm{i})}(t) = || \bigtriangledown_W \, w_{\mathrm{i}}(t) L_{\mathrm{i}}(t)||_2 \tag{9}$$

which measures the gradient magnitude of the ith task.

Forcing gradient magnitudes equal ignores the difference across tasks. Generally, tasks with higher loss magnitudes ought to dominate the weight updating process. As such, the relative inverse training rate (RITR) is introduced. Specifically, defining a baseline magnitude of loss function at time t as

$$\widetilde{L}_{\mathrm{i}}(t) = L_{\mathrm{i}}(t)/L_{\mathrm{i}}(0) \tag{10}$$

and the RITR w.r.t. the ith task is

$$r_{\mathrm{i}}(t) = \frac{\widetilde{L}_{\mathrm{i}}(t)}{\mathbb{E}[\widetilde{L}_{\mathrm{i}}(t)]}, \tag{11}$$

$$\mathbb{E}[\widetilde{L}_{\mathrm{i}}(t)] = \sum_{\mathrm{i}=1}^{N} \widetilde{L}_{\mathrm{i}}(t)/N. \tag{12}$$

Afterward, magnitudes of the ith task $G_W^{(\mathrm{i})}(t)$ is adjusted to $\bar{G}_W(t) \times [r_{\mathrm{i}}(t)]^{\alpha}$, where α is a hyperparameter that needs to be tuned. Specifically, α sets the force that pulls different tasks back to a common training balance. When the α is higher, it encourages the update of networks' weights to depend more on the tasks that produce relatively high losses. Then, the discrepancy between $G_W^{(\mathrm{i})}(t)$ and $\bar{G}_W(t) \times [r_{\mathrm{i}}(t)]^{\alpha}$ for all tasks is

$$L_{\mathrm{grad}}(t; w_{\mathrm{i}}(t)) = \sum_{\mathrm{i}} \left\| G_W^{(\mathrm{i})}(t) - \bar{G}_W(t) \times [r_{\mathrm{i}}(t)]^{\alpha} \right\|_1. \tag{13}$$

The target gradient norm $\bar{G}_W(t) \times [r_{\mathrm{i}}(t)]^{\alpha}$ is treated as constant when differentiating this loss function to avoid the loss weights $w_{\mathrm{i}}(t)$ dropping to zero. Finally, the weight $w_{\mathrm{i}}(t), i = 1, 2, \ldots, N$ is updated by

$$w(t+1) = w(t) + \lambda \bigtriangledown_{w_{\mathrm{i}}} L_{\mathrm{grad}}. \tag{14}$$

D. Architecture of Balanced MoE

Fig. 4 shows the architecture of balanced MoE, which is composed of MoE module and TGB module. For the MoE module, the embedding layer embeds inputs into a semantic representation, which is fed into gating networks and expert networks. The output of them are aggregated as per (5), followed by task-specific towers which predict the corresponding quality variables. For the TGB module, every task is endowed with an initial weight value $\omega_{\mathrm{i}}(0) = 1$. After that, the hyperparameter α needs to be decided. The embedding layer is chosen to be the layer that actually applied GradNorm, from which we extract the weights W. Thereafter, in the training process, the forward propagation is executed, and loss $L_{\mathrm{i}}(t)$ for batch x_{i} is computed. Consequently, the $G_W^{\mathrm{i}}(t)$, $r_{\mathrm{i}}(t)$, L_{grad}, $\bigtriangledown_{w_{\mathrm{i}}} L_{\mathrm{grad}}$ are obtained. On this basis, the $w_{\mathrm{i}}(t)$ is updated according to the (14), while $\Theta(t)$ is updated using the standard backward pass. $w_{\mathrm{i}}(t)$ also needs to be renormalized to make sure that $\sum_{\mathrm{i}=1}^{N} w_{\mathrm{i}}(t+1) = N$ for N tasks.

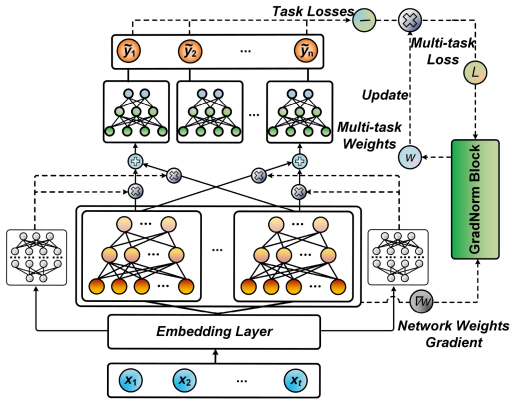

Fig. 4. Structures of proposed Balanced mixture-of-experts.

Comparing to the widely used Relu in (15), the Mish in (16) is designed as the activation function [24]. As such, the model can keep good performance as the network layer increases, thanks to the smoothness brought by the Mish, which makes the model receive and spread the information better. Besides, to remit the overfitting phenomenon, the dropout operation that randomly disconnects some parameters by a certain percentage during the training process is adopted. In addition, the dynamic learning rate is also adopted for the framework, which means the learning rate will decay to a much smaller one after certain training epochs

$$Relu(x) = \max(0, x) \tag{15}$$

$$Mish(x) = x \cdot \tanh(ln(1 + e^x)). \tag{16}$$

E. Algorithm Procedure

Total algorithm procedure is formulated as Algorithm 1, and the performance of the model is evaluated using root mean square error (RMSE), mean absolute error (MAE), and R^2 listed as below

$$\mathrm{RMSE} = \sqrt{\frac{1}{M} \sum\nolimits_{\mathrm{i}=1}^{M} (y_{\mathrm{i}} - \widehat{y}_{\mathrm{i}})^2} \tag{17}$$

$$\mathrm{MAE} = \frac{1}{M} \sum_{\mathrm{i}=1}^{M} |(y_{\mathrm{i}} - \widehat{y}_{\mathrm{i}})| \tag{18}$$

$$R^2 = 1 - \frac{\sum_{\mathrm{i}=1}^{M} (y_{\mathrm{i}} - \widehat{y}_{\mathrm{i}})^2}{\sum_{M} (y_{\mathrm{i}} - \bar{y})^2} \tag{19}$$

Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR. Downloaded on January 22,2024 at 04:57:23 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1: BMoE Training Algorithm.

Forward Propagation

Input: Sample data X after normalization.

Parameter: T: max iteration number; K: number of expert networks; f: function of shared layer; ϕ_{k} function of kth expert network; g_{i} function of gating network with respect to ith variable; t_{i} function of the tower network with respect to ith variable.

Output: Predictions for variables $\{\widehat{y}_{\mathrm{i}}, i = 1, 2, \ldots, N\}$

for $t \in [0, \mathrm{T}]$ **do**
  for Batch input data $x \in X$ **do**
    $\widehat{x} \leftarrow f(x)$ (through shared layer to reduce dimension of input data).
    Compute $\phi_{\mathrm{k}}(\widehat{x})$ (through experts layer).
    Compute $g_{\mathrm{i}}(\widehat{x})$ (through gating networks)
    $\psi_{\mathrm{i}}(\widehat{x}) \leftarrow \sum_{\mathrm{k}=1}^{K} g_{\mathrm{i}}(\widehat{x})_{\mathrm{k}} \phi_{\mathrm{k}}(\widehat{x})$; $\widehat{y}_{\mathrm{i}} \leftarrow \Phi_{\mathrm{i}}(\psi_{\mathrm{i}}(\widehat{x}))$
  end for
end for

Backward Propagation

Input: α: relative inverse training rate coefficient; W: network weights of shared layer; $\omega_{\mathrm{i}}(0)$: initial task weight for task i; y_{i}: batch real data for task i; $\widehat{y}_{\mathrm{i}}$: batch prediction data for task i; $\mathcal{W}$: total network weights

Parameter: T: max iteration number; f_{i}: loss function for task i; L_{i}: prediction loss of task i; $G_W^{(\mathrm{i})}$: the L_2 norm of the gradient of the weighted single-task loss $\omega_{\mathrm{i}}(t)L_{\mathrm{i}}(t)$ with respect to the shared weights W; $\bar{G}_W$: average gradient norm within all tasks; $\tilde{L}_{\mathrm{i}}$: loss ratio for task i; $r_{\mathrm{i}}(t)$: relative inverse training rate for task i; $Lstrok_{;\mathrm{grad}}$: L_1 loss function between actual and target gradient norms; N: number of tasks;

Output: Trained multi-task network.

for $t \in [0, \mathrm{T}]$ **do**
  $L_{\mathrm{i}}(t) \leftarrow f_{\mathrm{i}}(y_{\mathrm{i}}, \widehat{y}_{\mathrm{i}})$; $L(t) \leftarrow \sum_{\mathrm{i}} \omega_{\mathrm{i}}(t) L_{\mathrm{i}}(t)$
  $G_W^{(\mathrm{i})}(t) \leftarrow \|\nabla_W \omega_{\mathrm{i}}(t) L_{\mathrm{i}}(t)\|_2$;
  $\bar{G}_W(t) \leftarrow E_{\mathrm{task}}[G_W^{(\mathrm{i})}(t)]$
  $\tilde{L}_{\mathrm{i}}(t) \leftarrow \frac{L_{\mathrm{i}}(t)}{L_{\mathrm{i}}(0)}$; $r_{\mathrm{i}}(t) \leftarrow \frac{\tilde{L}_{\mathrm{i}}(t)}{E_{\mathrm{task}}[\tilde{L}_{\mathrm{i}}(t)]}$
  $L_{\mathrm{grad}} \leftarrow \sum_{\mathrm{i}} |G_W^{(\mathrm{i})}(t) - \bar{G}_W(t) \times [r_{\mathrm{i}}(t)]^{\alpha}|_1$
  Compute $\nabla_{\omega_{\mathrm{i}}} L_{\mathrm{grad}}$, with Keeping targets $\bar{G}_W(t) \times [r_{\mathrm{i}}(t)]^{\alpha}$ constant
  update $\omega_{\mathrm{i}}(t+1)$ using $\nabla_{\omega_{\mathrm{i}}} L_{\mathrm{grad}}$; update $\mathcal{W}(t+1)$ using $\nabla_{\mathcal{W}} L(t)$
  Normalize $\omega_{\mathrm{i}}(t+1)$ to make that $\sum_{\mathrm{i}} \omega_{\mathrm{i}}(t+1) = N$
end for

where M is the number of samples; $y_{\mathrm{i}}, \widehat{y}, \bar{y}$ denote the real value, predicted value, mean value of y, respectively. Generally, lower value of RMSE and MAE indicates lower prediction error of the model, while higher value of R^2 score shows a better ability for the model to make predictions.

IV. Experiments

In this section, experiments are conducted to evaluate performance of proposed method and answer the following research topics.

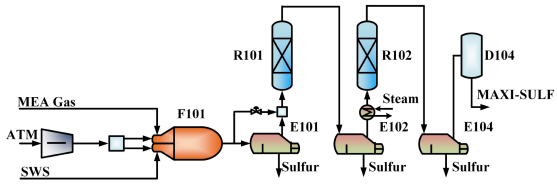

Fig. 5. Flowsheet of the SRU unit.

TABLE I
Relevant Variables

Variables	Notations
x_1	Gas flow MEA_GAS
x_2	Air flow AIR_MEA
x_3	Secondary air flow AIR_MEA_2
x_4	Gas flow in Sour Water Stripping (SWS) zone
x_5	Air flow in SWS zone AIR_SWS_TOT
y_1	Concentration of H_2S in the tail gas
y_2	Concentration of SO_2 in the tail gas

1) Does BMoE outperform other approaches significantly?
2) How does gradient balance module balance training procedure?
3) Is BMoE robust under different settings of hyperparameters?

A. Experimental Setup

1) Dataset: The sulfur recovery unit (SRU) is a vital part in refinery process, which removes pollutants from acid gas streams produced in chemical process to recover sulfur from process [25]. The following reactions will take place in SRU:

$$\begin{cases} 2\mathrm{H_2S} + 3\mathrm{O_2} = 2\mathrm{SO_2} + 2\mathrm{H_2O} \\ \frac{2}{3}x\mathrm{H_2S} + \frac{1}{3}x\mathrm{SO_2} = \mathrm{S}_x + \frac{2}{3}x\mathrm{H_2O}. \end{cases} \tag{20}$$

The tail gas of SRU still contains H_2S and SO_2 residuals, which should be monitored before being released into the atmosphere in case such contaminants cause danger to the public. However, the SRU system has features including but not limited to nonlinearity and multivariate. It is difficult to directly measure H_2S and SO_2 concentration due to the harsh working environment, which is necessary to construct soft sensors of such components. From the perspective of chemical reactions, decreasing from H_2S concentration may result in increasing of SO_2 concentration from the first reaction. However, from the second reaction, the decrease of H_2S concentration will decrease the SO_2 concentration. How to establish task relationships between these two components remains a challenge.

To construct the soft sensor model, five process variables marked in the flowsheet of SRU in Fig. 5 are selected as the covariates. The detailed descriptions are listed in Table I.

2) Experimental Details: To better illustrate the superiority of the proposed methods, some single-task learning methods that are widely used in soft sensors are selected to evaluate our proposed model: Partial least squares regression, AE, and its variant, stacked autoencoder (SAE), gated stacked autoencoder (GASE) [14], long short-term memory network

Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR. Downloaded on January 22,2024 at 04:57:23 UTC from IEEE Xplore. Restrictions apply.

TABLE II
PREDICTIVE PERFORMANCE (MEAN ± STD) FOR H_2S AND SO_2

		H_2S			SO_2		
	Models	RMSE	MAE	R^2	RMSE	MAE	R^2
ST	PLS	0.0347±0.0002	0.0207±0.0001	0.6236±0.0033	0.0268±0.0000	0.0206±0.00002	0.7867±0.0007
	AE	0.0333±0.0008	0.0236±0.0020	0.6547±0.0167	0.0266±0.0011	0.0212±0.0009	0.7895±0.0170
	SAE	0.0334±0.0007	0.0242±0.0009	0.6517±0.0137	0.0265±0.0010	0.0210±0.0008	0.7917±0.0163
	GSAE	0.0286±0.0008	0.0190±0.0013	0.7442±0.0148	0.0255±0.0006	0.0197±0.0006	0.8063±0.0086
	LSTM	0.0290±0.0004	0.0178±0.0006	0.7368±0.0081	0.0261±0.0004	0.0197±0.0003	0.7980±0.0067
	VALSTM	0.0278±0.0010	0.0178±0.0009	0.7568±0.0165	0.0268±0.0007	0.0207±0.0007	0.7847±0.0111
	SLSTM	0.0231±0.0003	0.0162±0.0007	0.8321±0.0040	0.0216±0.0006	0.0152±0.0007	0.8604±0.0075
MT	GSAE	0.0419±0.0038	0.0310±0.0029	0.4476±0.1035	0.0274±0.0007	0.0213±0.0009	0.7774±0.0118
	VALSTM	0.0290±0.0016	0.0180±0.0011	0.7357±0.0289	0.0300±0.0012	0.0232±0.0008	0.7296±0.0206
	SLSTM	0.0253±0.0004	0.0199±0.0005	0.7983±0.0071	0.0303±0.0003	0.0268±0.0002	0.7251±0.0051
	MMoE	0.0234±0.0006	0.0165±0.0009	0.8286±0.0093	0.0229±0.0010	0.0181±0.0005	0.8440±0.0142
Ours	**BMoE**	**0.0207±0.0013***	**0.0149±0.0007***	**0.8665±0.0163***	**0.0222±0.0006**	**0.0176±0.0006**	**0.8534±0.0079**

(LSTM) [26], variable attention-based long short-term memory (VALSTM) [27], and supervised long short-term memory (SLSTM) [28].

Since these models were originally designed for univariate prediction, which is incompatible with the motivation of this work. For the concern of fairness, the multivariate variants are also proposed in the result comparison. In the single-task learning models, each quality variable is modeled via one model, while in the MTL models multiple quality variables are modeled by one model.

The data is divided with a ratio of 6:2:2 for training, validation, and test. Afterward, normalization is applied for all the subsets separately. All models are trained with a learning rate of 0.01, and a weight decay rate of 0.001. All experiments are proposed on a desktop with Intel i9 Gen, Nvidia RTX 2060 under Python 3.8. The PLS model is constructed on the Sklearn package of Python, and the deep learning models are constructed on PyTorch. All experiments are run at least ten times under ten random seeds.

B. Performance Comparison

The comparison between the performance of the proposed BMoE and that of its baseline methods is mainly considered in this section, which is conducted on the SRU benchmark, with mean and standard deviation reported over ten runs with different random seeds. The numerical indices are listed in Table II. Underlined results indicate the best baselines over each metric. "*" marks the methods that improve the best baselines significantly at p-value ≤ 0.05 over paired samples t-test, and "**" marks the methods that improve the best baselines at p-value ≤ 0.01. (Single task for each model (ST), multitask (MT) trained at the same time for each model). Moreover, the detailed prediction results of our BMoE model are presented in Fig. 8.

In Table II, the proposed BMoE achieves significant improvement compared with different baselines. Nevertheless, the BMoE does not show the best performance for the SO_2 prediction task. It achieves the best balanced performance between these two gas content simultaneous prediction tasks among those baselines by keeping a highly precise prediction result for the SO_2 contents while achieving relatively better performance on the H_2S prediction task at the same time. More intuitively, it is hard for these single-task baseline methods to learn these multiple quality variables simultaneously.

According to our results, these single-task baseline methods will suffer performance loss while they are forced to accomplish multitasks at the same time. For instance, the content of H_2S prediction task is strongly affected by the jointly SO_2 prediction task, while the R2 drops from 0.7442 when the model is trained for the single task separately to 0.4476, and shows great variance. The LSTM-based baseline methods perform better, but they still generally show poorer performance facing multitasks, which means that the highly related quality variables prediction tasks affect the other prediction tasks in a bad way, as called negative transfer. Furthermore, back to the AE-based models, SAE may not perform as well as a single AE, while the GSAE works much better in the meantime. LSTM and its variants like VALSTM and SLSTM are considered to be great improvements as they perform best among these single-task learning baselines. These phenomena indicate that the aforementioned reactions will influence the data-driven model performance, and the simultaneous prediction of two gases should consider the reactions to avoid the negative transfer, which makes the model deteriorate.

C. Effectiveness of the GradNorm Block.

In this section, the effectiveness of the GradNorm block is discussed in detail. The contribution of the GradNorm block is studied via extracting the training process of the model with constant and equal weights. Meanwhile, the adaptive weights are adjusted by the GradNorm at the same time. Note that, the learning rate of the training process is dropped as 1/10 of the initial learning rate after 100 epochs.

Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR. Downloaded on January 22,2024 at 04:57:23 UTC from IEEE Xplore. Restrictions apply.

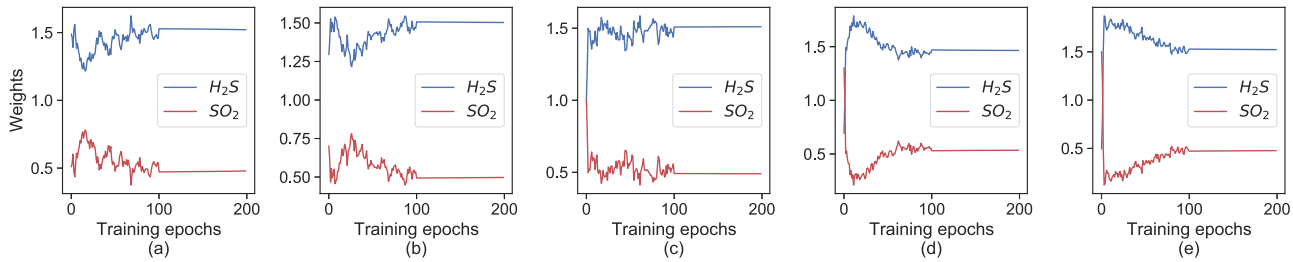

Fig. 6. Dynamic change of task weights adjusted by the GradNorm block with different initial task weights. (a) (1.5, 0.5), (b) (1.3, 0.7), (c) (1.0, 1.0), (d) (0.7, 1.3), and (e) (0.5, 1.5) for (H_2S, SO_2).

In Fig. 9, the performance of the quality prediction tasks on H_2S and SO_2 indicated by the RMSE, R2, and MAE, respectively, is presented. According to Fig. 9, as the H_2S prediction task is conducted similarly, with GradNorm block, the SO_2 prediction task performs much better, the R2 score of which raises from 0.8286 to 0.8665, which means that the seesaw phenomenon between multitasks has been overcome by the GradNorm block. To better learn how the GradNorm block works in the training process, dynamic change of task weights started by different initial weight ratios is studied as shown in Fig. 6. In such training processes, only the initial task weight ratio is changed from 0.5:1.5 to 1.5:0.5 (weight of H_2S prediction task loss: weight of SO_2 prediction task loss). It is obvious that according to the result, no matter what the initial task weight ratio is, the weights are always consistent at nearly 1.5 for SO_2 and 0.5 for H_2S after the training process. This consistency proves the task balance ability of the GradNorm block.

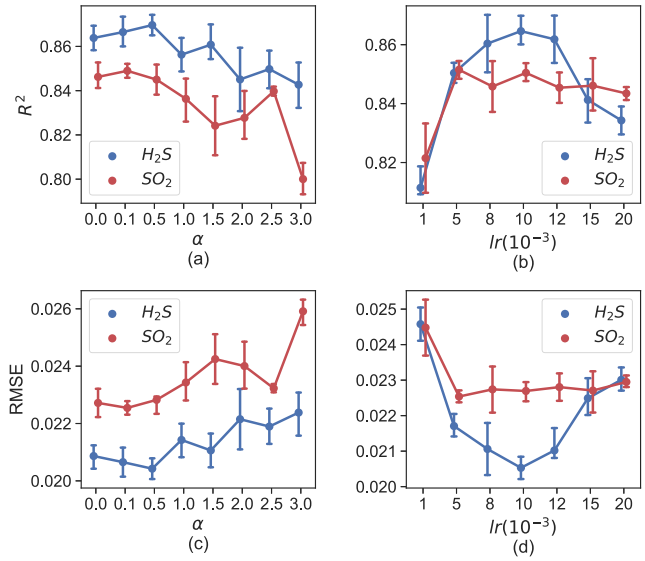

Fig. 7. Parameter study of the relative inverse training rate coefficient α and the learning rate lr. (a), (c) for α measured by R2, and RMSE respectively; (b), (d) for lr measured by R2, and RMSE respectively.

D. Parameter Sensitivity Study

Two critical hyperparameters in BMoE are discussed in this section, 1) learning rate lr; and 2) relative inverse training rate coefficient α, which strongly influence the performance of BMoE.

In Fig. 7 , the influence of α is studied by varying it in the range [0,3]. Generally, BMoE performs well when α is at a relatively low value, while it drops obviously with α increasing. Note that $\alpha = 0$ means that the GradNorm block always wants to adjust the norms of backpropagated gradients of each task to become equal at W, the subnet weights. Especially, R2 of H_2S prediction and SO_2 prediction achieve 0.8665 and 0.8490, respectively, while the R2 of H_2S prediction reduced from 0.8397 to 0.8000 when the α changes from 2.5 to 3.0. Moreover, the performances of the H_2S and SO_2 prediction tasks roughly keep the same when α increases. It illustrates that α does not cause drastically preference for any single task, but influences the total performance of the BMoE model. As such, α is generally suggested within the range [0.1, 0.5].

Furthermore, the initial learning rate of BMoE varies from 0.001 to 0.02. Remarkably, as the learning rate increases, BMoE performs better before $lr = 0.01$, and then turns to performs worse after that point. Especially, the R2 of H_2S prediction and SO_2 prediction achieve 0.8653 and 0.8556, respectively, at $lr = 0.01$, while they are 0.8114, 0.8215 at $lr = 0.001$ and 0.8409, 0.8387 at $lr = 0.02$. Another observation is that the performance of the BMoE becomes unstable when lr is at a relatively high value. In that way, lr is generally suggested within range [0.008, 0.012].

E. Running Time

The running time of our proposed model is studied in Fig. 10. The results show that the complexity of our proposed model is concentrated on the back propagation process, specifically, the GradNorm algorithm. According to our results, our model can generate prediction result in a relatively high speed, comparing to traditional sensors.

V. Conclusion

A novel MTL model combined with MMoE structure and GradNorm algorithm as a dynamic modeling approach in the soft sensor field called BMoE was developed in this article, which can predict multiple variables that are hard to be directly measured in the actual industrial process with good accuracy. As the task imbalance scenario remains one of the main challenges for MTL, the GradNorm algorithm was applied to balance the gradients between different tasks. The experiments on the

Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR. Downloaded on January 22,2024 at 04:57:23 UTC from IEEE Xplore. Restrictions apply.

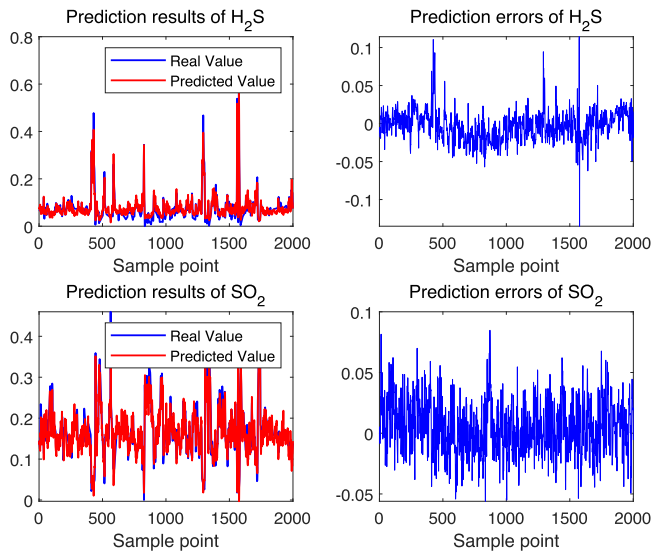

Fig. 8. Prediction results on SRU benchmark.

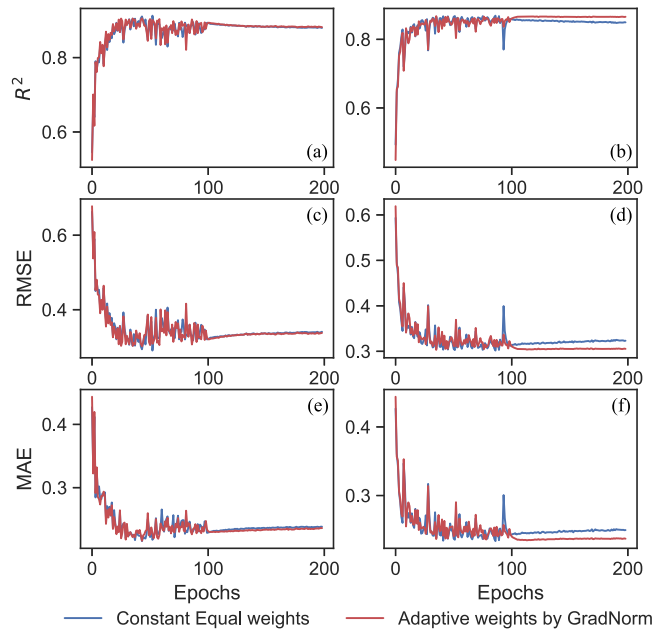

Fig. 9. Performance of the GradNorm block during the training process. (a), (c), (e) for H_2S prediction in R2, RMSE, and MAE respectively; (b), (d), (f) for SO_2 prediction in R2, RMSE, and MAE, respectively.

SRU benchmark showed that the proposed BMoE performs better than the widely used AE-based and LSTM-based baseline methods like GSAE, VALSTM, and SLSTM.

As for future work, more MTL methods like subnetwork routing [29], PLS [21], and their reasonable application to the soft sensor field is still worth exploring, such models can further improve the efficiency of joint representation learning while helping solve negative transfer problem. Secondly, as samples in the chemical process are generally time series data, a further block that can deal better with time series characteristics needs to be concerned.

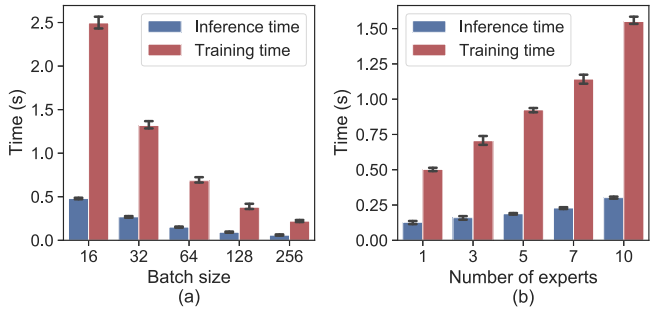

Fig. 10. Inference time and training time ($\mu \pm 3\sigma$) for a single epoch of all the training data (about 6000 samples) with various (a) batch size and (b) number of experts.

ACKNOWLEDGMENT

The first author Y. Huang would like to thank Mr. Zhichao Chen at Zhejiang University for his selfless help and constructive suggestions on the writing of this article.

REFERENCES

[1] E. Camponogara, A. Plucenio, A. F. Teixeira, and S. R. Campos, "An automation system for gas-lifted oil wells: Model identification, control, and optimization," *J. Petroleum Sci. Eng.*, vol. 70, no. 3, pp. 157–167, 2010.

[2] H. J. Galicia, Q. P. He, and J. Wang, "A reduced order soft sensor approach and its application to a continuous digester," *J. Process Control*, vol. 21, no. 4, pp. 489–500, 2011.

[3] P. Kadlec, B. Gabrys, and S. Strandt, "Data-driven soft sensors in the process industry," *Comput. Chem. Eng.*, vol. 33, no. 4, pp. 795–814, 2009.

[4] T. Komulainen, M. Sourander, and S.-L. Jämsä-Jounela, "An online application of dynamic PLS to a dearomatization process," *Comput. Chem. Eng.*, vol. 28, no. 12, pp. 2611–2619, 2004.

[5] D. Eon Lee, S.-O. Song, and E. Sup Yoon, "A nonlinear soft sensor based on modified SVR for quality estimation in polymerization," *IFAC Proc. Volumes*, vol. 36, no. 5, pp. 879–883, 2003.

[6] A. R. de Miranda, T. M. G. d. A. Barbosa, R. Araújo, and S. G. S. Alcalá, "An on-line extreme learning machine with adaptive architecture for soft sensor design," in *Proc. IEEE SENSORS*, 2016, pp. 1–3.

[7] X. Zhang, X. Deng, and P. Wang, "Double-level locally weighted extreme learning machine for soft sensor modeling of complex nonlinear industrial processes," *IEEE Sensors J.*, vol. 21, no. 2, pp. 1897–1905, Jan. 2021.

[8] B. Shen, L. Yao, and Z. Ge, "Nonlinear probabilistic latent variable regression models for soft sensor application: From shallow to deep structure," *Control Eng. Pract.*, vol. 94, 2020, Art. no. 104198.

[9] K. S. Rana, L.-W. Chen, L.-H. Tang, and W.-T. Hong, "A study on speech enhancement using deep temporal convolutional neural network," in *Proc. IEEE Int. Conf. Consum. Electron.-Taiwan*, 2021, pp. 1–2.

[10] M. Luo, X. Chang, L. Nie, Y. Yang, A. G. Hauptmann, and Q. Zheng, "An adaptive semisupervised feature analysis for video semantic recognition," *IEEE Trans. Cybern.*, vol. 48, no. 2, pp. 648–660, Feb. 2018.

[11] K. Chen, L. Yao, D. Zhang, X. Wang, X. Chang, and F. Nie, "A semisupervised recurrent convolutional attention model for human activity recognition," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 5, pp. 1747–1756, May 2020.

[12] Y. Wu, D. Liu, X. Yuan, and Y. Wang, "A just-in-time fine-tuning framework for deep learning of SAE in adaptive data-driven modeling of time-varying industrial processes," *IEEE Sensors J.*, vol. 21, no. 3, pp. 3497–3505, Feb. 2021.

[13] X. Yuan, B. Huang, Y. Wang, C. Yang, and W. Gui, "Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted SAE," *IEEE Trans. Ind. Inform.*, vol. 14, no. 7, pp. 3235–3243, Jul. 2018.

[14] Q. Sun and Z. Ge, "Deep learning for industrial KPI prediction: When ensemble learning meets semi-supervised data," *IEEE Trans. Ind. Inform.*, vol. 17, no. 1, pp. 260–269, Jan. 2021.

Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR. Downloaded on January 22,2024 at 04:57:23 UTC from IEEE Xplore. Restrictions apply.

[15] X. Chang and Y. Yang, "Semisupervised feature analysis by mining correlations among multiple tasks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2294–2305, Oct. 2017.
[16] I. Kokkinos, "UberNet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5454–5463.
[17] M. Long, C. Zhangjie, J. Wang, and Y. S. Philip, "Learning multiple tasks with deep relationship networks," in *Proc. Adv. Neural Inf. Process. Syst. 30: Annu. Conf. Neural Inf. Process. Syst.*, G. Isabelle et al., Eds., Long Beach, CA, USA, Dec. 2017, pp. 1594–1603. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/03e0704b5690a2dee1861dc3ad3316c9-Abstract.html
[18] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3994–4003.
[19] S. Ruder, J. Bingel, I. Augenstein, and A. Søgaard, "Latent multi-task architecture learning," in *Proc. 33rd AAAI Conf. Artif. Intell.*, Honolulu, Hawaii, USA: AAAI Press, Jan.-Feb. 2019, pp. 4822–4829. [Online]. Available: https://dblp.org/rec/conf/aaai/RuderBAS19.bib
[20] Y. Liu, Z. Wang, H. Jin, and I. Wassell, "Multi-task adversarial network for disentangled feature learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3743–3751.
[21] H. Tang, J. Liu, M. Zhao, and X. Gong, "Progressive layered extraction (PLE): A novel multi-task learning (MTL) model for personalized recommendations," in *Proc. 14th ACM Conf. Recommender Syst.*, New York, NY, USA, 2020, pp. 269–278.
[22] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, New York, NY, USA, 2018, pp. 1930–1939.
[23] Z. Chen, V. Badrinarayanan, C. Lee, and A. Rabinovich, "Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *Proc. 35th Int. Conf. Mach. Learn.*, D. G. Jennifer and K. Andreas, Eds., Stockholm, Sweden: PMLR, vol. 80, Jul. 2018, pp. 793–802. [Online]. Available: http://proceedings.mlr.press/v80/chen18a.html
[24] D. Misra, "Mish: A self regularized non-monotonic neural activation function," in *Proc. 31st British Mach. Vis. Conf.*, UK: BMVA Press, Sep. 2020, pp. 1594–1603. [Online]. Available: https://www.bmvc2020-conference.com/assets/papers/0928.pdf
[25] L. Fortuna, A. Rizzo, M. Sinatra, and M. Xibilia, "Soft analyzers for a sulfur recovery unit," *Control Eng. Pract.*, vol. 11, no. 12, pp. 1491–1500, 2003.
[26] W. Ke, D. Huang, F. Yang, and Y. Jiang, "Soft sensor development and applications based on LSTM in deep neural networks," in *Proc. IEEE Symp. Ser. Comput. Intell.*, 2017, pp. 1–6.
[27] X. Yuan, L. Li, Y. Wang, C. Yang, and W. Gui, "Deep learning for quality prediction of nonlinear dynamic processes with variable attention-based long short-term memory network," *Can. J. Chem. Eng.*, vol. 98, no. 6, pp. 1377–1389.
[28] X. Yuan, L. Li, and Y. Wang, "Nonlinear dynamic soft sensor modeling with supervised long short-term memory network," *IEEE Trans. Ind. Inform.*, vol. 16, no. 5, pp. 3168–3176, May 2020.
[29] J. Ma, Z. Zhao, J. Chen, A. Li, L. Hong, and E. H. Chi, "SNR: Sub-network routing for flexible parameter sharing in multi-task learning," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, pp. 216–223.

Yuxin Huang received the B.Eng. in automation in 2021 from the College of Control Science and Engineering, Zhejiang University, Hangzhou, China, where he is currently working toward the master's degree in cyberspace security.

His research interests include multitask learning, soft sensor modeling, and time series analysis.

Eric Hao Wang received the B.Eng. degree in aerospace engineering from Central South University, Changsha, China, in 2020. He is currently working toward the master's degree in electronic engineering with Zhejiang University, Hangzhou, China.

He is currently an Intern Researcher with Microsoft Research Asia, Beijing, China. His research interests include nuclear process monitoring and time series analysis.

Zhaoran Liu received the B.E. degree in automation from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2020. He is currently working toward the master's degree in system engineering with the College of Control Science and Engineering, Zhejiang University, Hangzhou, China.

His research interests include deep learning, signal processing, and natural language processing.

Licheng Pan received his bachelor's degree in automation from the College of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2002. He is currently pursuing his Master's degree in systems engineering at Zhejiang University.

His research interests include neural differential equations and time series generation.

Haozhe Li received the B.E. degree in engineering from Jiangnan University, Wuxi, China, in 2020. He is currently working toward the master's degree in system engineering with the College of Control Science and Engineering, Zhejiang University, Hangzhou, China.

His research interests include signal processing, graph neural network, and deep learning.

Xinggao Liu received the Ph.D. degree in control science and technology from Zhejiang University, Hangzhou, China, in 2000.

He is currently a Professor in control science and engineering with Zhejiang University. His research interests include image processing, machine learning, multivariate statistical modeling, optimization, and dynamic optimization.

Authorized licensed use limited to: INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR. Downloaded on January 22,2024 at 04:57:23 UTC from IEEE Xplore. Restrictions apply.