Approximated Orthogonal Projection Unit: Stabilizing Regression Network Training Using Natural Gradient

Shaoqi Wang[†]   **Chunjie Yang**[†]   **Siwei Lou**[†]

[†] Zhejiang University, China, `{sq_w,cjyang999,swlou}@zju.edu.cn`

Abstract

Neural networks (NN) are extensively studied in cutting-edge soft sensor models due to their feature extraction and function approximation capabilities. Current research into network-based methods primarily focuses on models' offline accuracy. Notably, in industrial soft sensor context, online optimizing stability and interpretability are prioritized, followed by accuracy. This requires a clearer understanding of network's training process. To bridge this gap, we propose a novel NN named the Approximated Orthogonal Projection Unit (AOPU) which has solid mathematical basis and presents superior training stability. AOPU truncates the gradient backpropagation at dual parameters, optimizes the trackable parameters updates, and enhances the robustness of training. We further prove that AOPU attains minimum variance estimation (MVE) in NN, wherein the truncated gradient approximates the natural gradient (NG). Empirical results on two chemical process datasets clearly show that AOPU outperforms other models in achieving stable convergence, marking a significant advancement in soft sensor field.

1 Introduction

Deep learning methods have achieved recent success in many regression areas such as natural language processing, protein structure prediction, and building energy consumption forecasting. However, for these methods to be useful in the industrial soft sensor field, which demands higher immediacy and stability, further research into model structure and the stability of the training process is necessary [1, 2, 3]. The safety and economic impact of factory impose stringent requirements on soft sensor models deployed online [4, 5]. For example, each mini-batch update must not cause significant performance fluctuations to ensure that downstream controllers and monitors do not execute erroneous actions; soft sensor models must be deployed online to avoid fluctuations due to changes in operating conditions and model switching [6]. Common network's training tricks are not suitable for soft sensor contexts. For example, it's not feasible to use checkpoints for early stopping but to always use the latest updated checkpoint; there wouldn't be adaptive learning rate changes but rather a constant learning rate maintained throughout. Experimental results demonstrate that such differences lead to a substantial decline in performance. These constraints necessitate the development of better-suited network architectures for regression task that ensure more stable optimization during training [7, 6].

MVE is the best unbiased estimator under the Mean Squared Error (MSE) criterion, essentially representing the performance ceiling for regression models [8]. Unfortunately, directly applying MVE to NN is challenging due to the difficulty in obtaining the likelihood distribution $p(y|x)$ of inputs x and outputs y [9]. Traditional research on MVE has focused on techniques like Kalman filtering [10, 11], algorithms based on Ordinary Least Squares (OLS) [12, 13], and other system identification research [14, 15, 16] which operate under linear and convex conditions. These methods, while effective within their scope, have limited expressive power [17].
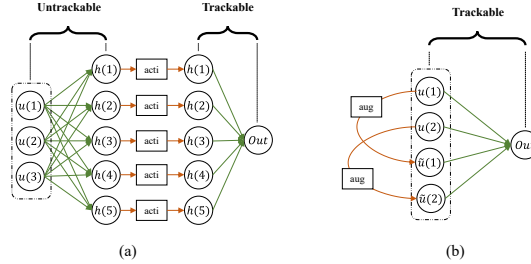
Figure 1: Trackable parameters and Untrackable parameters. Solid green lines represent model parameters, and orange curves represent non-parametric operations. (a) Conventional deep NN framework. (b) Typical broad learning system framework through data enhancement.

Many studies have explored NN optimization from various perspectives such as adaptive learning rates [18], momentum updates [19] and customized loss functions [20]. Compared to the first-order optimization methods, second-order optimization algorithms based on Natural Gradient Descent (NGD) [21, 22] can achieve faster and more stable convergence [23]. This is because NGD considers the manifold distribution of model parameters by computing the Fisher Information Matrix (FIM) during the gradient update process. However, calculating FIM introduces significant computational overhead, making NGD challenging to implement in most machine learning models [24]. Much research is focused on adapting model structures for NGD [25, 26, 27] and reducing its computational costs [28, 29, 30], yet applying NGD to NN optimization remains an unsolved issue [31, 32, 33].

Some studies have approached the regression task from the perspective of targeted modular design [34, 35, 36], emphasizing the construction of local neuron-level rules to assist models in learning conducive features , exemplified by SLSTM [37], SIAE [38], MIF-Autoformer [39], and CBMP [40]. Some endeavors have yielded results with solid theoretical underpinnings [41, 5, 42], such as S4 [43], VAE, VIOG [44]. These examples, integrate with other great work [27, 45, 42, 46], jointly demonstrate excellent integrations of NN with theoretical basis and also have a stronger expressivity compared to the identification algorithms [14, 15, 47]. However, even though these networks possess certain interpretability, it is still unclear whether these prior biases are beneficial for regression task [48, 49]. Furthermore, their mathematical foundations are not rooted in soft sensor tasks, which do not guarantee stable performance in regression [50].

AOPU differs from conventional studies by focusing on training optimization and the overall input-output relationships. Assuming there is a robust feature extraction module (augmentation block), AOPU pays specific attention on better optimization and more stable convergence. AOPU innovatively introduces trackable and dual parameters, enhancing an structure approximation of MVE. The dual parameters are injective representations of trackable parameters, mainly aiding in truncating the gradient backpropagation process. This truncation will be validated as an effective approximation to NGD. The augmentation module boosts AOPU's performance and also acts as an extension interface, making AOPU more versatile. Rank Ratio (RR) is introduced as an interpretability index to provide deep and comprehensive insights into the network dynamics. RR quantifies the ratio of linearly independent samples within a batch, providing a measure of the heterogeneity and diversity of information. By harnessing RR value, we can roughly foresee the performance in advance of the training. A high RR suggests the model output more closely approximates the MVE, and optimization is more in line with NGD, leading to superior performance. Conversely, a low RR implies that the precision of computation is compromised, resulting in inferior performance.

2 AOPU:Methodology

2.1 Trackable vs. Untrackable

Parameters within NN that can be decoupled from input data x and computed through an inner product are defined as **trackable** parameters. Conversely, parameters that cannot be decoupled are classified as **untrackable** parameters.

$$f(x) = W^T x = \langle W, x \rangle \qquad g(x) = M(x)^T x = \langle M(x), x \rangle \tag{1}$$

(a) gradient decline ignores the manifold (radical)   (b) natural gradient recognizes the manifold (conservative)
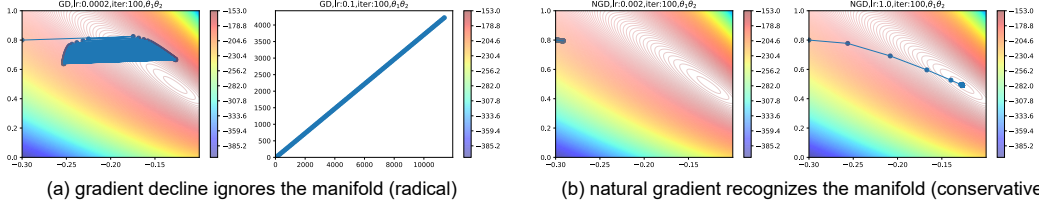
Figure 2: Comparison between NGD and GD. Direction matters more than step size (learning rate) in stable convergence.

where W represents a parameter matrix and $M(\cdot)$ denotes an operator dependent on x. According to the definition, W is identified as a trackable parameter, whereas $M(x)$ is considered untrackable.

A significant proportion of parameters in NN are untrackable as depicted in Fig. 1. This predominance is attributable to the networks' reliance on stacking activation functions to bolster their nonlinear modeling capabilities. Proposition 1 indicates that any parameter influenced by an activation function transitions to being input-dependent, thus rendering it untrackable.

Proposition 1. *There **does not** exist an transition operator T independent of x such that for a given parameter matrix W, and $\forall x_1, x_2$, the following equations hold,*

$$\mathrm{acti}(Wx_1) = T(W)x_1, \quad \mathrm{acti}(Wx_2) \quad = T(W)x_2 \tag{2}$$

Proof is in Appendix C.3.

2.2 Natural Gradient vs. Gradient

Fig. 2 vividly presents the major difference between NGD and GD using a simple GPR in experiment. This GPR had only two parameters, the bias of the mean and the coefficient of the kernel matrix, both constant values. We sampled 100 instances from this GPR and updated these two parameters 100 times using these samples. It was observed that NG require a higher learning rate, while conventional gradients only need a smaller one. The major difference between NG and conventional gradients lies in their directions. Conventional gradients ignore the parameter manifold and treat every parameter equally. NG, by dividing the gradient by its second derivative, treat sensitive parameters cautiously (low gradient) and non-sensitive parameters boldly (high gradient). This adjustment results in different gradient directions and contributes to better convergence.

Nevertheless, the calculation of NG involves considering the inverse of the FIM, thereby introducing computational complexity cubic to the number of parameters, making it infeasible for neural networks. Existing research on NG focuses on conventional machine learning, e.g., considering more complex distributions (such as the product of multiple exponential family) for computing NG. Research in NN domain on NG mostly centers on second-order optimizers (such as AdamW), which are merely first-order approximations of second-order NG.

2.3 Network's structure

We wish to emphasize that within AOPU framework, the truncated gradient of the dual parameters serves as an approximation of the NG of the trackable parameters, while AOPU's structured output approximates the MVE. The fidelity of these approximations is measured by the RR, the closer RR is to 1, the more precise the approximation; conversely the closer RR is to 0, the more precision loss occurs. Furthermore, it can be demonstrated that the output of AOPU fundamentally differs from traditional neural networks: instead of explicitly modeling a mapping from x to y, it implicitly models a mapping from x to $\tilde{x}y$. To ensure that y can be recovered from $\tilde{x}y$, it is imperative that RR equals 1. AOPU also guarantees the convergence of the dual parameters if the input-output relationship can be characterized by specific system. The proof of above is intricate and comprehensive, and one may refer to Appendix A, B and C for more detailed information. This section focuses on the implementation of AOPU.

AOPU utilize data augmentation to replace stacked activation structures to enhance the nonlinear modeling capabilitie. In such designs, the choice of the data augmentation module forms a crucial
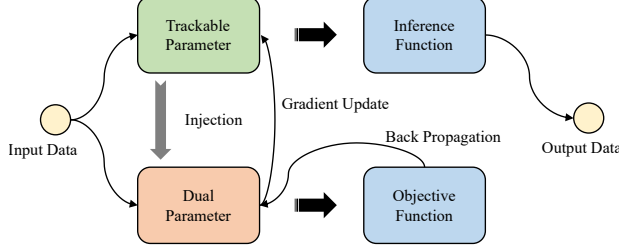
Figure 3: AOPU's data flow schematic. The gradient is backpropagated but truncated at the dual parameter, and this gradient is then used to update the trackable parameter.

model prior. For ease of implementation, AOPU adopts a random weight matrix approach for its data augmentation module [51, 52]. Specifically, suppose the original feature dimension of the input data is d, and the defined model hidden dimension size is h. Let $\hat{G}$ be a fixed Gaussian initialized random weight matrix, $\hat{G} \in \mathbb{R}^{d,h}$, and for input data $x \in \mathbb{R}^{d,b}$ where b is the mini-batch size, the data augmentation process is as follows,

$$\tilde{x} = \text{concat}[\text{acti}(\hat{G}^T x), x] \tag{3}$$

Subsequently, for the augmented $\tilde{x}$, the output is processed using the trackable parameter $\tilde{W} \in \mathbb{R}^{(d+h),o}$, where o represents the output dimension and, for simplicity, we assume $o = 1$ indicating research into univariate output. The output function is then,

$$g(\hat{y}|x) = \tilde{x}^T \tilde{W} \tag{4}$$

The optimization of parameter $\tilde{W}$ in AOPU differs from other networks. We introduce a dual parameter $D(\tilde{W})$ to describe this process,

$$D(\tilde{W}) = \tilde{x}\tilde{x}^T \tilde{W} \tag{5}$$

and the loss is computed using the following objective function,

$$\mathcal{L} = \frac{1}{b} \sum_{i=1}^{b} \left[y_i - [(\tilde{x}^T \tilde{x})^{-1} \tilde{x}^T D(\tilde{W})]_i \right]^2 \tag{6}$$

AOPU innovatively introduces trackable and dual parameters. The dual parameters are injective representations of trackable parameters, mainly aiding in truncating the gradient backpropagation process. During the training process, gradient backpropagation is truncated at $D(\tilde{W})$, and the gradient of the dual parameter updates the original trackable parameter as demonstrated in Fig. 3, thus completing the training of the model.

It is important to note that the training process involves the inversion of a matrix $\tilde{x}^T \tilde{x}$, which is not always invertible, thus introducing numerical computation issue. We employ the reciprocals of the positive singular values to circumvent the solvability issues that arise when an inverse does not exist. However, this approach introduces significant computational precision loss, which in turn prompts a thorough analysis of RR.

We define the metric RR to represent the ratio of the rank of $\tilde{x}$ to its batch size. Clearly, RR is a value between $[0, 1]$, and as RR approaches 1, the process of approximating the inverse of $\tilde{x}^T \tilde{x}$ through eigenvalue decomposition becomes more accurate (owing to the presence of more reciprocals of eigenvalues). When RR equals 1, $\tilde{x}^T \tilde{x}$ is an invertible matrix; conversely, the smaller the RR, the less stable the model's numerical computations and likely poorer performance.

AOPU utilizes parameters' trackability nature to accelearte the NG computation. The key to AOPU's capability to rapidly approximate the NG is its ability to bypass the FIM computation and its inverse. The key to the capability to skip the FIM lies in our utilization of Eq. 10, which separates the model parameters from the variables and data. In this context, we can use the gradient matrix of the expectation parameter with respect to natural parameter to replace the FIM (without actually performing this calculation), thereby substituting the original complicated NG computation with an equivalent conventional gradient computation, i.e., $\nabla_\lambda m = F(\lambda) \rightarrow \nabla_m \lambda = (F(\lambda))^{-1}$, so

that $(F(\lambda))^{-1}\nabla_\lambda \mathcal{L} \to \nabla_m \mathcal{L}$. This allows us to use the automatic differentiation toolbox for rapid calculations. Otherwise we must explicitly compute the inverse of the network's FIM, which is very time-consuming and memory-intensive. For instance, a network with 20kB of 32-bit parameters, which equates to 5120 trainable parameters, requires inverting a 5120-dimensional matrix with each training iteration. This requirement grows with model size and can easily lead to GPU memory shortages. More critically, such large matrix inversions often lead to significant numerical precision loss, severely impairing model performance.

3 Experiments and Analysis

In this section, we detail the experimental results of the AOPU model, analyze the impact of hyperparameters on AOPU, its robustness regarding changes in hyperparameters, its advantages over other comparative algorithms, and some inherent limitations of the model. Comprehensive and detailed experiments and comparisons have been conducted on two publicly available chemical process datasets, Debutanizer and Sulfur Recovery Unit (SRU). For more information of the dataset please refer to Appendix D.

3.1 Baselines

We choose seven different NN models as baselines: Autoformer, Informer, DNN, SDAE, SVAE, LSTM, and RVFLNN, covering four major domains including RNN-based networks, auto-encoder-based networks, attention-based networks, and MLP-based networks. Notably, all baseline models except RVFLNN and AOPU operate solely within the latent space, meaning that there are linear transformations mapping the input data to the latent space and from the latent space to the output space before and after the baseline models. This approach is designed to better control the model size.

3.2 Experiment Implementation

Apart from AOPU, which is trained using the approximated minimum variance estimation loss function as previously described, all other deep learning algorithms are trained using the Mean Squared Error (MSE) loss. AOPU's learning rate for gradient updates is set at 1.0, while for all other deep learning algorithms, it is set at 0.005, with the Adam optimizer used for gradient updates. The learning rates of all models remain static throughout the training process. The experimental setup differs based on the requirements of various models regarding input dimensions. Models such as Autoformer, Informer, and LSTM necessitate an input that includes an additional dimension for 'sequence length'. This dimension is preserved as part of the input structure for these models. Conversely, models like DNN, SDAE, SVAE, AOPU, and RVFLNN do not require this additional dimension. For these models, the sequence length and input dimensions are combined and flattened to serve as the feature dimensions in the input. AOPU's latent space size is set at 2048. Autoformer, Informer, SDAE, and SVAE utilize two layers each for their encoder and decoder layers; LSTM uses two layers of LSTM layers; RVFLNN and AOPU share identical settings. All models except AOPU and RVFLNN have their latent space sizes set at 16 to ensure the trainable parameters size across all models are comparable.

3.3 Main Result

3.3.1 How certain we are about the inverse

According to the previous discussion, the existence of the inverse of $\tilde{x}^T \tilde{x}$ is crucial as it does not only impact the numerical stability of the model but also directly determines whether it is possible to recover y from the approximated mapping relationship $\tilde{x} \to \tilde{x}y$. Clearly, the input feature dimensions $d+h$ and the batch size b significantly affect whether the inverse of $\tilde{x}^T \tilde{x}$ exists. Specifically, the larger the batch size, the more columns $\tilde{x}$ has, and the less likely $\tilde{x}$ is to be column-full-rank; conversely, the longer the sequence length and the larger the input feature dimensions, the more likely $\tilde{x}$ is to be linearly independent and thus column-full-rank. From the following experimental results, it will be clearly observed the impact of batch size and sequence length on RR.
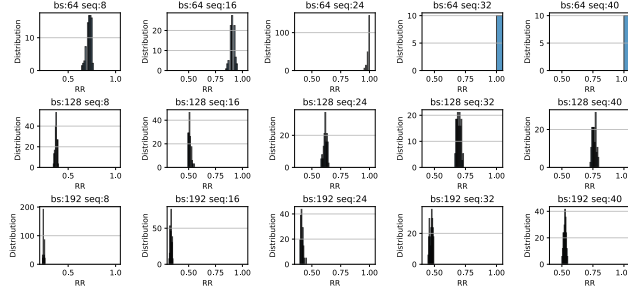
Figure 4: Histogram of the frequency distribution of RR on SRU dataset under varying batch sizes and sequence length settings.
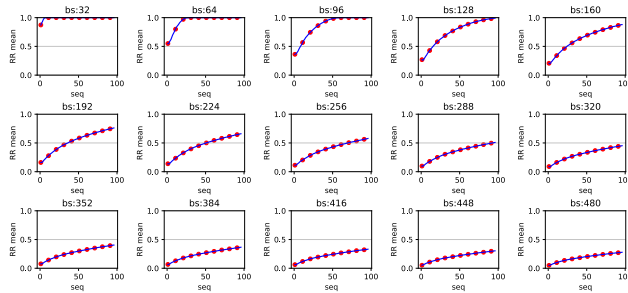

Figure 5: Curve of the mean of RR distribution on SRU dataset under varying batch sizes and sequence length settings.

Fig. 4 shows the distribution of the RR for the AOPU model across various batch sizes and sequence length combinations on the SRU dataset, where **bs** stands for batch size and **seq** for sequence length. The experimental results align with the previous analysis: increasing the batch size with a fixed sequence length significantly decreases the RR distribution, whereas increasing the sequence length with a fixed batch size significantly increases it.

Fig. 5 shows the mean values of RR distribution changing with sequence length under different batch size settings, marked by red circles at every ten data points. Clearly, the experimental results shown in the figure corroborate our analysis that with increasing batch size, the curve's slope becomes flatter, indicating the model's decreasing sensitivity to changes in sequence length. Compared to Fig. 4, Fig. 5 provides additional insights into the mean values of the RR distribution relative to sequence length and batch size, offering a more comprehensive insight for subsequent experimental interpretations. Results of the RR study on the Debutanizer are listed in Appendix F.

3.3.2 Is the training stable

Stability is a crucial characteristic for the online deployment of deep learning models in actual production processes. Specifically, the incremental updates to model parameters following the observation of new mini-batch data should have a smooth impact on model performance. However, experimental results indicate that most networks in the soft sensor field fail to achieve stable convergence. Fig. 6 provides a detailed display of how the MSE metrics for different networks change with training iterations on the SRU validation dataset, with blue solid circles marked every 50 iterations. It is evident that all models, except Autoformer, Informer, LSTM, and AOPU, exhibit significant performance fluctuations as training iterations progress. The density of the blue solid circles can to some extent represent the likelihood of corresponding performance fluctuations.

It can be observed that the SDAE and SVAE networks, despite experiencing significant fluctuations in validation performance (indicated by large fluctuations in the curve), are mostly stable (as shown by the blue circles concentrated below the curve). In contrast, the DNN and RVFLNN networks
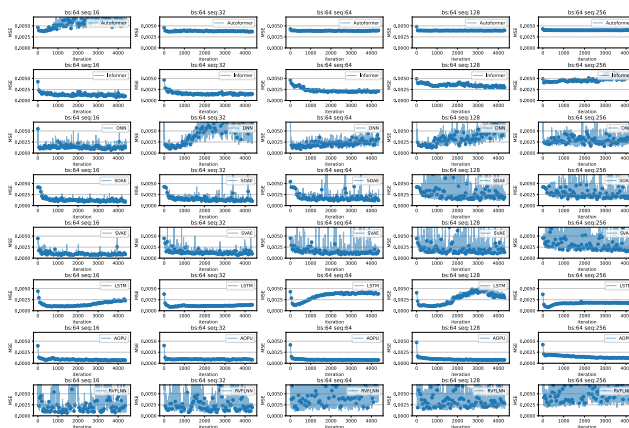
Figure 6: Curves of SRU validation loss changes with training iteration for different models with a fixed batch size of 64 at different sequence length settings. The curves are shown in translucent blue, with a solid blue circle labeled on the curve every 50 iterations.

have relatively unstable convergence (indicated by blue circles evenly distributed above and below the curve). Although Autoformer and Informer have relatively stable convergence dynamics, their performance is relatively poor. Specifically, Autoformer consistently converges to a bad output, whereas Informer can effectively learn under identical settings but is sensitive to changes in seq, which can lead to model performance collapse. The convergence process of LSTM is relatively stable, partly explaining why it is a widely adopted baseline in the field of time series analysis; however, LSTM is significantly prone to overfitting and its performance is not outstanding.

In contrast to all other network, AOPU exhibits exceptionally impressive performance. AOPU demonstrates very stable and rapid convergence, with almost no fluctuations in performance as training iterations progress. Furthermore, AOPU is less sensitive to changes in hyperparameters and does not exhibit significant overfitting, making it a truly reliable and deployable NN model in production processes. We also notice that with bigger batch size setting, the fluctuations will be mitigated. For comparative training dynamics under other batch size settings, refer to Appendix F.

3.3.3 Quantitative analysis

To verify the reliability of the AOPU model's performance and to quantify its comparison with other methods, we implemented two different training strategies. Strategy one involved an early stopping trick and used the best checkpoint to validate the model's performance on the test dataset. Strategy two involved training all models for 40 epochs and using the final checkpoint to test the model performance. All following experiments has batch size set to 64. The outputs of strategy one are presented at Table 1, while the results of strategy two are presented in Table 2. All NN configurations were subjected to 20 independent repeat experiments, with the mean of the experiments represented by uppercase numbers on the left of the table and the standard deviation by lowercase subscript numbers on the right.

From Table 1, we can intuitively compare the optimal performance among all models. Overall, there is not much difference in final performance among the various networks. Notably, almost all MAPE metrics on the Debutanizer dataset exceed 100 due to a sample in the test dataset where the butane content is nearly zero, which significantly distorts the MAPE calculation. While AOPU performs comparably to other network models in terms of the R^2 metric, its stability is significantly superior, as indicated by much lower standard deviations in the R^2 values compared to all other models.

Further, to more closely align with real industrial application scenarios, if we do not record the optimal checkpoint but instead complete training for 40 epochs as shown in Table 2, the performance of the models significantly declines. Despite this, AOPU continues to provide stable and reliable performance. In Table 2, it is noted that using the R^2 metric, AOPU consistently performs best with the sequence length set at 48, and the performance drop from the optimal results calculated using

7

Table 1: The evaluation metrics for different models under different seq settings on Debutanizer and SRU dataset with bs set to 64 and best training epoch recorded

Seq	Name	Debutanizer			SRU		
		MSE	MAPE	R^2	MSE	MAPE	R^2
16	Autoformer	$0.0314_{\pm0.0017}$	$177.2_{\pm24.56}$	$0.1588_{\pm0.0481}$	$0.00312_{\pm0.00037}$	$0.3295_{\pm0.0310}$	$0.2079_{\pm0.0957}$
	Informer	$0.0165_{\pm0.0023}$	$130.9_{\pm19.26}$	$0.5574_{\pm0.0641}$	$0.00090_{\pm0.00011}$	$0.1579_{\pm0.0117}$	$0.7715_{\pm0.0294}$
	DNN	$0.0146_{\pm0.0016}$	$121.9_{\pm30.82}$	$0.6076_{\pm0.0440}$	$0.00081_{\pm0.00017}$	$0.1645_{\pm0.0249}$	$0.7944_{\pm0.0441}$
	SDAE	$0.0134_{\pm0.0021}$	$136.1_{\pm13.79}$	$0.6406_{\pm0.0573}$	$0.00080_{\pm0.00006}$	$0.1453_{\pm0.0102}$	$0.7967_{\pm0.0154}$
	SVAE	$0.0138_{\pm0.0013}$	$128.8_{\pm30.74}$	$0.6294_{\pm0.0352}$	$0.00078_{\pm0.00004}$	$0.1214_{\pm0.0091}$	$0.7992_{\pm0.0102}$
	LSTM	$0.0259_{\pm0.0027}$	$161.2_{\pm50.17}$	$0.3058_{\pm0.0741}$	$0.00088_{\pm0.00011}$	$0.1445_{\pm0.0106}$	$0.7750_{\pm0.0292}$
	AOPU	$0.0149_{\pm0.0009}$	$200.5_{\pm9.277}$	$0.5992_{\pm0.0243}$	$0.00072_{\pm0.00007}$	$0.1694_{\pm0.0114}$	$0.8154_{\pm0.0190}$
	RVFLNN	$0.0114_{\pm0.0019}$	$78.88_{\pm36.11}$	$0.6936_{\pm0.0534}$	$0.00074_{\pm0.00009}$	$0.1584_{\pm0.0148}$	$0.8128_{\pm0.0249}$
24	Autoformer	$0.0295_{\pm0.0020}$	$153.1_{\pm39.66}$	$0.2091_{\pm0.0550}$	$0.00236_{\pm0.00030}$	$0.2685_{\pm0.0262}$	$0.4014_{\pm0.0774}$
	Informer	$0.0154_{\pm0.0042}$	$139.3_{\pm19.59}$	$0.5864_{\pm0.1145}$	$0.00110_{\pm0.00022}$	$0.1670_{\pm0.0232}$	$0.7210_{\pm0.0572}$
	DNN	$0.0122_{\pm0.0027}$	$124.7_{\pm43.75}$	$0.6717_{\pm0.0733}$	$0.00091_{\pm0.00014}$	$0.1671_{\pm0.0192}$	$0.7684_{\pm0.0361}$
	SDAE	$0.0120_{\pm0.0023}$	$144.9_{\pm25.18}$	$0.6786_{\pm0.0631}$	$0.00076_{\pm0.00009}$	$0.1489_{\pm0.0131}$	$0.8070_{\pm0.0234}$
	SVAE	$0.0127_{\pm0.0023}$	$143.0_{\pm33.10}$	$0.6599_{\pm0.0642}$	$0.00072_{\pm0.00004}$	$0.1289_{\pm0.0143}$	$0.8149_{\pm0.0122}$
	LSTM	$0.0262_{\pm0.0029}$	$192.2_{\pm48.06}$	$0.2961_{\pm0.0795}$	$0.00091_{\pm0.00014}$	$0.1420_{\pm0.0122}$	$0.7672_{\pm0.0364}$
	AOPU	$0.0133_{\pm0.0006}$	$178.3_{\pm11.16}$	$0.6421_{\pm0.0181}$	$0.00080_{\pm0.00004}$	$0.1868_{\pm0.0063}$	$0.7965_{\pm0.0110}$
	RVFLNN	$0.0109_{\pm0.0012}$	$67.40_{\pm26.06}$	$0.7070_{\pm0.0338}$	$0.00079_{\pm0.00011}$	$0.1638_{\pm0.0168}$	$0.7986_{\pm0.0297}$
32	Autoformer	$0.0307_{\pm0.0018}$	$199.3_{\pm31.82}$	$0.1770_{\pm0.0495}$	$0.00222_{\pm0.00022}$	$0.2730_{\pm0.0291}$	$0.4366_{\pm0.0573}$
	Informer	$0.0237_{\pm0.0049}$	$119.4_{\pm18.42}$	$0.3648_{\pm0.1325}$	$0.00131_{\pm0.00020}$	$0.1849_{\pm0.0164}$	$0.6679_{\pm0.0250}$
	DNN	$0.0120_{\pm0.0027}$	$93.96_{\pm45.53}$	$0.6779_{\pm0.0735}$	$0.00095_{\pm0.00014}$	$0.1693_{\pm0.0149}$	$0.7589_{\pm0.0374}$
	SDAE	$0.0111_{\pm0.0020}$	$131.9_{\pm27.59}$	$0.686_{\pm0.0539}$	$0.00080_{\pm0.00005}$	$0.1534_{\pm0.0124}$	$0.7959_{\pm0.0145}$
	SVAE	$0.0133_{\pm0.0024}$	$143.9_{\pm33.72}$	$0.6419_{\pm0.0646}$	$0.00078_{\pm0.00007}$	$0.1300_{\pm0.0118}$	$0.8010_{\pm0.0182}$
	LSTM	$0.0257_{\pm0.0029}$	$180.1_{\pm48.29}$	$0.3097_{\pm0.0785}$	$0.00094_{\pm0.00009}$	$0.1488_{\pm0.0084}$	$0.7603_{\pm0.0234}$
	AOPU	$0.0119_{\pm0.0005}$	$155.6_{\pm4.827}$	$0.6791_{\pm0.0155}$	$0.00076_{\pm0.00003}$	$0.1847_{\pm0.0048}$	$0.8063_{\pm0.0078}$
	RVFLNN	$0.0121_{\pm0.0016}$	$57.82_{\pm34.15}$	$0.6751_{\pm0.0445}$	$0.00091_{\pm0.00012}$	$0.1725_{\pm0.0129}$	$0.7679_{\pm0.0308}$
40	Autoformer	$0.0307_{\pm0.0023}$	$199.4_{\pm21.66}$	$0.1771_{\pm0.0642}$	$0.00245_{\pm0.00020}$	$0.2879_{\pm0.0210}$	$0.3790_{\pm0.0509}$
	Informer	$0.0237_{\pm0.0059}$	$119.4_{\pm32.86}$	$0.3648_{\pm0.1597}$	$0.00168_{\pm0.00054}$	$0.2205_{\pm0.0334}$	$0.5723_{\pm0.1369}$
	DNN	$0.0120_{\pm0.0029}$	$93.97_{\pm42.94}$	$0.678_{\pm0.0792}$	$0.00099_{\pm0.00017}$	$0.1784_{\pm0.0191}$	$0.7489_{\pm0.0445}$
	SDAE	$0.0111_{\pm0.0023}$	$134.4_{\pm36.79}$	$0.7009_{\pm0.0618}$	$0.00081_{\pm0.00007}$	$0.1546_{\pm0.0119}$	$0.7938_{\pm0.0180}$
	SVAE	$0.0116_{\pm0.0025}$	$127.2_{\pm31.86}$	$0.6868_{\pm0.0687}$	$0.00082_{\pm0.00012}$	$0.1383_{\pm0.0162}$	$0.7898_{\pm0.0310}$
	LSTM	$0.0261_{\pm0.0039}$	$197.2_{\pm56.01}$	$0.2994_{\pm0.1059}$	$0.00093_{\pm0.00011}$	$0.1479_{\pm0.0093}$	$0.7638_{\pm0.0284}$
	AOPU	$0.0110_{\pm0.0004}$	$143.4_{\pm6.977}$	$0.7030_{\pm0.0121}$	$0.00071_{\pm0.00001}$	$0.1785_{\pm0.0034}$	$0.8193_{\pm0.0048}$
	RVFLNN	$0.0148_{\pm0.0013}$	$53.59_{\pm30.26}$	$0.6013_{\pm0.0373}$	$0.00093_{\pm0.00013}$	$0.1761_{\pm0.0170}$	$0.7644_{\pm0.0337}$
48	Autoformer	$0.0321_{\pm0.0022}$	$189.4_{\pm24.87}$	$0.1390_{\pm0.0614}$	$0.00249_{\pm0.00015}$	$0.2879_{\pm0.0220}$	$0.3676_{\pm0.0380}$
	Informer	$0.0247_{\pm0.0069}$	$122.6_{\pm36.11}$	$0.3385_{\pm0.1865}$	$0.00190_{\pm0.00043}$	$0.2270_{\pm0.0386}$	$0.5186_{\pm0.1092}$
	DNN	$0.0114_{\pm0.0027}$	$110.0_{\pm42.75}$	$0.6941_{\pm0.0732}$	$0.00105_{\pm0.00013}$	$0.1796_{\pm0.0168}$	$0.7332_{\pm0.0344}$
	SDAE	$0.0105_{\pm0.0014}$	$128.5_{\pm16.98}$	$0.7182_{\pm0.0400}$	$0.00083_{\pm0.00005}$	$0.1564_{\pm0.0106}$	$0.7890_{\pm0.0133}$
	SVAE	$0.0105_{\pm0.0017}$	$116.5_{\pm34.32}$	$0.7180_{\pm0.0473}$	$0.00087_{\pm0.00010}$	$0.1389_{\pm0.0098}$	$0.7789_{\pm0.0256}$
	LSTM	$0.0268_{\pm0.0036}$	$185.6_{\pm63.68}$	$0.2813_{\pm0.0982}$	$0.00091_{\pm0.00010}$	$0.1471_{\pm0.0099}$	$0.7691_{\pm0.0312}$
	AOPU	$0.0107_{\pm0.0003}$	$144.9_{\pm7.492}$	$0.7118_{\pm0.0087}$	$0.00070_{\pm0.00001}$	$0.1796_{\pm0.0019}$	$0.8209_{\pm0.0037}$
	RVFLNN	$0.0157_{\pm0.0026}$	$60.11_{\pm29.00}$	$0.5792_{\pm0.0717}$	$0.00100_{\pm0.00016}$	$0.1833_{\pm0.0205}$	$0.7454_{\pm0.0412}$

strategy one is minimal. For instance, on the Debutanizer dataset, AOPU's final optimal performance is 0.6054, which is a 14.9% decrease from the original 0.7118; on the SRU dataset, its best final performance is 0.8069, only a 1.7% decrease from 0.8209.

Compared to the results on the Debutanizer and SRU datasets, other models show more significant declines: Autoformer dropped from 0.2091 and 0.4366 to -0.2396 and 0.1536 respectively, a decline of 123.9% and 64.8%; Informer decreased from 0.5864 and 0.7715 to 0.4047 and 0.6897 respectively, declines of 30.9% and 10.6%; DNN dropped from 0.6941 and 0.7944 to 0.5109 and 0.5639 respectively, declines of 26.3% and 29.0%; SDAE decreased from 0.7182 and 0.8070 to 0.4470 and 0.7209 respectively, declines of 37.7% and 10.7%; SVAE dropped from 0.7180 and 0.8149 to 0.4315 and 0.7397 respectively, declines of 39.9% and 9.2%; LSTM went from 0.3097 and 0.7750 to -0.3903 and 0.6280 respectively, declines of 226.0% and 19.0%; RVFLNN decreased from 0.7070 and 0.8128 to 0.2844 and 0.5652 respectively, declines of 59.8% and 30.5%.

In terms of robustness, AOPU shows a significant improvement. It is observed that AOPU's model performance steadily improves as sequence length increases, in contrast to the other comparison networks, which exhibit large fluctuations. For example, RVFLNN's R^2 on the SRU dataset drastically drops from 0.5652 at seq 16 to -0.8785 at seq 32, indicating extreme robustness.

In terms of stability, AOPU not only demonstrates an outstanding advantage in terms of average R^2 values but also in standard deviation. From Table 2, it can be seen that the standard deviation for AOPU between the two training strategies changes only slightly, for instance, from 0.0087 to

Table 2: The evaluation metrics for different models under different seq settings on Debutanizer and SRU dataset with bs set to 64 and no best training epoch recorded

| | Model | Dataset & Metric | | | | | |
| Seq | Name | Debutanizer | | | SRU | | |
		MSE	MAPE	R^2	MSE	MAPE	R^2
16	Autoformer	$0.0942_{\pm 0.0370}$	$217.0_{\pm 78.49}$	$-1.5240_{\pm 0.9912}$	$0.00836_{\pm 0.00426}$	$0.5362_{\pm 0.1755}$	$-1.118_{\pm 1.0810}$
	Informer	$0.0283_{\pm 0.0070}$	$140.8_{\pm 30.66}$	$0.2414_{\pm 0.1885}$	$0.00122_{\pm 0.00036}$	$0.1750_{\pm 0.0226}$	$0.6897_{\pm 0.0918}$
	DNN	$0.0215_{\pm 0.0028}$	$161.8_{\pm 46.14}$	$0.4233_{\pm 0.0760}$	$0.00172_{\pm 0.00073}$	$0.2238_{\pm 0.0410}$	$0.5639_{\pm 0.1850}$
	SDAE	$0.0217_{\pm 0.0036}$	$144.6_{\pm 43.52}$	$0.4184_{\pm 0.0977}$	$0.00113_{\pm 0.00017}$	$0.1559_{\pm 0.0211}$	$0.7114_{\pm 0.0453}$
	SVAE	$0.0217_{\pm 0.0061}$	$161.6_{\pm 53.46}$	$0.4166_{\pm 0.1644}$	$0.00113_{\pm 0.00044}$	$0.1469_{\pm 0.0329}$	$0.7121_{\pm 0.1131}$
	LSTM	$0.0521_{\pm 0.0152}$	$215.5_{\pm 102.7}$	$-0.3956_{\pm 0.4094}$	$0.00251_{\pm 0.00112}$	$0.2055_{\pm 0.0325}$	$0.3634_{\pm 0.2844}$
	AOPU	$0.0215_{\pm 0.0007}$	$206.6_{\pm 9.059}$	$0.4239_{\pm 0.0211}$	$0.00098_{\pm 0.00013}$	$0.1963_{\pm 0.0132}$	$0.7518_{\pm 0.0336}$
	RVFLNN	$0.0329_{\pm 0.0391}$	$107.1_{\pm 40.42}$	$0.1171_{\pm 1.0470}$	$0.00171_{\pm 0.00112}$	$0.2540_{\pm 0.0867}$	$0.5652_{\pm 0.2853}$
24	Autoformer	$0.0969_{\pm 0.0293}$	$315.7_{\pm 113.7}$	$-1.5980_{\pm 0.7864}$	$0.00457_{\pm 0.00145}$	$0.3901_{\pm 0.0764}$	$-0.1590_{\pm 0.3682}$
	Informer	$0.0222_{\pm 0.0053}$	$134.9_{\pm 29.93}$	$0.4047_{\pm 0.1432}$	$0.00146_{\pm 0.00047}$	$0.1878_{\pm 0.0280}$	$0.6280_{\pm 0.1201}$
	DNN	$0.0204_{\pm 0.0036}$	$159.7_{\pm 27.21}$	$0.4518_{\pm 0.0972}$	$0.00281_{\pm 0.00126}$	$0.2760_{\pm 0.0699}$	$0.2879_{\pm 0.3206}$
	SDAE	$0.0206_{\pm 0.0069}$	$152.2_{\pm 44.43}$	$0.4470_{\pm 0.1873}$	$0.00118_{\pm 0.00023}$	$0.1688_{\pm 0.0219}$	$0.6998_{\pm 0.0590}$
	SVAE	$0.0217_{\pm 0.0048}$	$152.6_{\pm 33.46}$	$0.4165_{\pm 0.1306}$	$0.00106_{\pm 0.00034}$	$0.1582_{\pm 0.0248}$	$0.7286_{\pm 0.0873}$
	LSTM	$0.0544_{\pm 0.0146}$	$226.9_{\pm 129.5}$	$-0.4574_{\pm 0.3932}$	$0.00313_{\pm 0.00141}$	$0.1878_{\pm 0.0337}$	$0.6280_{\pm 0.3582}$
	AOPU	$0.0185_{\pm 0.0005}$	$193.5_{\pm 5.519}$	$0.5040_{\pm 0.0136}$	$0.00093_{\pm 0.00009}$	$0.2022_{\pm 0.0099}$	$0.7628_{\pm 0.0232}$
	RVFLNN	$0.0338_{\pm 0.0470}$	$135.0_{\pm 86.13}$	$0.0945_{\pm 1.2610}$	$0.00507_{\pm 0.00725}$	$0.3854_{\pm 0.2346}$	$-0.2849_{\pm 1.8360}$
32	Autoformer	$0.0583_{\pm 0.0194}$	$217.3_{\pm 48.84}$	$-0.5629_{\pm 0.5216}$	$0.00334_{\pm 0.00101}$	$0.3279_{\pm 0.0407}$	$0.1536_{\pm 0.2573}$
	Informer	$0.0359_{\pm 0.0134}$	$135.4_{\pm 43.65}$	$0.0371_{\pm 0.3612}$	$0.00193_{\pm 0.00064}$	$0.2163_{\pm 0.0331}$	$0.5091_{\pm 0.1637}$
	DNN	$0.0206_{\pm 0.0049}$	$173.2_{\pm 44.64}$	$0.4459_{\pm 0.1316}$	$0.00278_{\pm 0.00146}$	$0.2780_{\pm 0.0858}$	$0.2949_{\pm 0.3695}$
	SDAE	$0.0211_{\pm 0.0045}$	$179.3_{\pm 61.21}$	$0.4342_{\pm 0.1228}$	$0.00110_{\pm 0.00032}$	$0.1812_{\pm 0.0253}$	$0.7209_{\pm 0.0824}$
	SVAE	$0.0239_{\pm 0.0083}$	$151.3_{\pm 46.68}$	$0.3598_{\pm 0.2234}$	$0.00102_{\pm 0.00049}$	$0.1669_{\pm 0.0361}$	$0.7397_{\pm 0.1256}$
	LSTM	$0.0519_{\pm 0.0204}$	$243.7_{\pm 75.24}$	$-0.3903_{\pm 0.5471}$	$0.00284_{\pm 0.00132}$	$0.2186_{\pm 0.0513}$	$0.2803_{\pm 0.3348}$
	AOPU	$0.0171_{\pm 0.0003}$	$191.0_{\pm 4.483}$	$0.5396_{\pm 0.0105}$	$0.00085_{\pm 0.00004}$	$0.1945_{\pm 0.0070}$	$0.7833_{\pm 0.0118}$
	RVFLNN	$0.0305_{\pm 0.0340}$	$86.84_{\pm 46.24}$	$0.1809_{\pm 0.9113}$	$0.00741_{\pm 0.01854}$	$0.4012_{\pm 0.4214}$	$-0.8785_{\pm 4.6942}$
40	Autoformer	$0.0462_{\pm 0.0136}$	$213.2_{\pm 62.30}$	$-0.2396_{\pm 0.3648}$	$0.00341_{\pm 0.00077}$	$0.3528_{\pm 0.0453}$	$0.1343_{\pm 0.1970}$
	Informer	$0.0194_{\pm 0.0059}$	$119.5_{\pm 23.91}$	$-0.1124_{\pm 0.5220}$	$0.00201_{\pm 0.00049}$	$0.2283_{\pm 0.0285}$	$0.4911_{\pm 0.1259}$
	DNN	$0.0182_{\pm 0.0033}$	$166.4_{\pm 36.94}$	$0.5109_{\pm 0.0892}$	$0.00422_{\pm 0.00360}$	$0.3376_{\pm 0.1262}$	$-0.0703_{\pm 0.9134}$
	SDAE	$0.0223_{\pm 0.0053}$	$158.9_{\pm 37.77}$	$0.4028_{\pm 0.1430}$	$0.00140_{\pm 0.00096}$	$0.1987_{\pm 0.0462}$	$0.6455_{\pm 0.2437}$
	SVAE	$0.0214_{\pm 0.0063}$	$147.5_{\pm 43.71}$	$0.4259_{\pm 0.1705}$	$0.00118_{\pm 0.00070}$	$0.1699_{\pm 0.0296}$	$0.7001_{\pm 0.1790}$
	LSTM	$0.0534_{\pm 0.0211}$	$219.6_{\pm 80.80}$	$-0.4307_{\pm 0.5669}$	$0.00391_{\pm 0.00244}$	$0.2460_{\pm 0.0584}$	$0.0090_{\pm 0.6181}$
	AOPU	$0.0156_{\pm 0.0003}$	$179.0_{\pm 4.713}$	$0.5804_{\pm 0.0105}$	$0.00076_{\pm 0.00004}$	$0.1843_{\pm 0.0052}$	$0.8069_{\pm 0.0101}$
	RVFLNN	$0.0267_{\pm 0.0130}$	$88.96_{\pm 30.42}$	$0.2844_{\pm 0.3489}$	$0.00338_{\pm 0.00277}$	$0.3133_{\pm 0.1144}$	$0.1442_{\pm 0.7023}$
48	Autoformer	$0.0492_{\pm 0.0122}$	$226.2_{\pm 46.25}$	$-0.3187_{\pm 0.3287}$	$0.00338_{\pm 0.00044}$	$0.3655_{\pm 0.0331}$	$0.1440_{\pm 0.1124}$
	Informer	$0.0440_{\pm 0.0175}$	$123.7_{\pm 40.20}$	$-0.1798_{\pm 0.4709}$	$0.00248_{\pm 0.00072}$	$0.2389_{\pm 0.0309}$	$0.3712_{\pm 0.1826}$
	DNN	$0.0194_{\pm 0.0043}$	$167.9_{\pm 37.16}$	$0.4799_{\pm 0.1166}$	$0.00393_{\pm 0.00219}$	$0.3284_{\pm 0.1002}$	$0.0046_{\pm 0.5544}$
	SDAE	$0.0212_{\pm 0.0066}$	$153.0_{\pm 44.14}$	$0.4316_{\pm 0.1772}$	$0.00120_{\pm 0.00028}$	$0.1831_{\pm 0.0272}$	$0.6953_{\pm 0.0723}$
	SVAE	$0.0212_{\pm 0.0059}$	$161.0_{\pm 39.33}$	$0.4315_{\pm 0.1595}$	$0.00109_{\pm 0.00038}$	$0.1811_{\pm 0.0378}$	$0.7223_{\pm 0.0984}$
	LSTM	$0.0682_{\pm 0.0319}$	$271.5_{\pm 131.8}$	$-0.8286_{\pm 0.8559}$	$0.00539_{\pm 0.00357}$	$0.2784_{\pm 0.0844}$	$-0.3652_{\pm 0.9060}$
	AOPU	$0.0147_{\pm 0.0003}$	$177.4_{\pm 4.092}$	$0.6054_{\pm 0.0094}$	$0.00076_{\pm 0.00003}$	$0.1862_{\pm 0.0063}$	$0.8069_{\pm 0.0092}$
	RVFLNN	$0.0479_{\pm 0.0555}$	$106.6_{\pm 74.56}$	$-0.2843_{\pm 1.489}$	$0.00898_{\pm 0.00208}$	$0.4238_{\pm 0.3727}$	$-1.2760_{\pm 5.2820}$

0.0094 on the Debutanizer dataset, an increase of about 8.0%. In contrast, the standard deviation for other networks often increases several-fold, such as Autoformer on the Debutanizer dataset, which increases from an optimal 0.0481 to 0.3287, an increase of about 583.4%; similar trends are observed with other models.

3.4 Ablation Study

In this section, we further investigate the effects of structural designs for augmentation through some ablation studies, examining the impacts of the ReLU piecewise activation function, the Tanh smooth activation function, and normalization on AOPU's performance. It is important to note that if AOPU is trained using direct gradient descent without dual parameter updates, it actually degenerates to an RVFLNN model, and this part of the ablation study has been detailed in section 3.3.3.

From Table 3 we can draw two conclusions: The first is normalization significantly impairs AOPU's model performance. The second it ReLU piecewise non-linear activation function suits worse for AOPU than the Tanh activation function. As previously analyzed in A where both the input data $\tilde{x}$ and y should to be zero mean, hence reducing the covariance operator R to an inner product operator. However, piecewise linear functions like ReLU and LeakyReLU are not zero-mean, which violates such assumptions.

9

Table 3: AOPU evaluation metrics on the Debutanizer and SRU datasets under various combinations of activation functions and layer normalization settings.

Structure			Dataset & Metric					
Acti		Norm		Debutanizer			SRU	
Relu	Tanh	LaNorm	MSE	MAPE	R^2	MSE	MAPE	R^2
☑	☐	☐	$0.0162_{\pm 0.00030}$	$183.2_{\pm 4.8862}$	$0.5654_{\pm 0.0082}$	$0.00074_{\pm 0.00002}$	$0.1838_{\pm 0.0031}$	$0.8115_{\pm 0.0060}$
☐	☑	☐	$0.0103_{\pm 0.00021}$	$148.6_{\pm 6.3927}$	$0.7216_{\pm 0.0059}$	$0.00077_{\pm 0.00003}$	$0.1902_{\pm 0.0049}$	$0.8026_{\pm 0.0084}$
☑	☐	☑	$0.0633_{\pm 0.02113}$	$117.2_{\pm 57.219}$	$-0.6961_{\pm 0.5650}$	$0.58831_{\pm 0.17940}$	$4.5829_{\pm 0.6003}$	$-147.98_{\pm 45.422}$
☐	☑	☑	$0.0189_{\pm 0.00662}$	$157.6_{\pm 62.892}$	$0.4930_{\pm 0.1773}$	$0.00205_{\pm 0.00082}$	$0.3149_{\pm 0.0619}$	$0.4803_{\pm 0.2055}$

Table 4: Comprehensive experiments on activation function influence to AOPU

Structure		Dataset & Metric					
Class	Acti		Debutanizer			SRU	
		MSE	MAPE	R^2	MSE	MAPE	R^2
zero-mean	Hard Shrink	$0.0103_{\pm 0.0003}$	$151.9_{\pm 6.359}$	$0.7236_{\pm 0.0081}$	$0.00076_{\pm 0.00002}$	$0.1882_{\pm 0.0037}$	$0.8069_{\pm 0.0059}$
	Tanh	$0.0104_{\pm 0.0004}$	$152.0_{\pm 6.425}$	$0.7202_{\pm 0.0114}$	$0.00076_{\pm 0.00002}$	$0.1894_{\pm 0.0052}$	$0.8056_{\pm 0.0106}$
	Tanh Shrink	$0.0103_{\pm 0.0002}$	$152.7_{\pm 5.805}$	$0.7221_{\pm 0.0068}$	$0.00077_{\pm 0.00003}$	$0.1891_{\pm 0.0056}$	$0.8043_{\pm 0.0093}$
	Soft Sign	$0.0103_{\pm 0.0003}$	$152.8_{\pm 6.309}$	$0.7225_{\pm 0.0091}$	$0.00077_{\pm 0.00002}$	$0.1905_{\pm 0.0033}$	$0.8040_{\pm 0.0068}$
	Soft Shrink	$0.0104_{\pm 0.0002}$	$152.6_{\pm 4.548}$	$0.7201_{\pm 0.0060}$	$0.00076_{\pm 0.00003}$	$0.1872_{\pm 0.0050}$	$0.8074_{\pm 0.0081}$
non-zero-mean	Sigmoid	$0.0157_{\pm 0.0004}$	$192.3_{\pm 6.336}$	$0.5777_{\pm 0.0124}$	$0.00080_{\pm 0.00003}$	$0.1925_{\pm 0.0048}$	$0.7955_{\pm 0.0089}$
	Relu6	$0.0171_{\pm 0.0004}$	$187.9_{\pm 5.502}$	$0.5416_{\pm 0.0109}$	$0.00074_{\pm 0.00003}$	$0.1836_{\pm 0.0051}$	$0.8111_{\pm 0.0076}$
	RRelu	$0.0165_{\pm 0.0004}$	$186.3_{\pm 5.224}$	$0.5582_{\pm 0.0074}$	$0.00076_{\pm 0.0004}$	$0.1858_{\pm 0.0047}$	$0.8072_{\pm 0.0105}$
	Hard Swish	$0.0103_{\pm 0.0002}$	$149.2_{\pm 5.500}$	$0.7232_{\pm 0.0078}$	$0.00077_{\pm 0.00003}$	$0.1900_{\pm 0.0036}$	$0.8026_{\pm 0.0076}$
	Mish	$0.0103_{\pm 0.0003}$	$151.4_{\pm 6.260}$	$0.7229_{\pm 0.0098}$	$0.00079_{\pm 0.00003}$	$0.1927_{\pm 0.0054}$	$0.7999_{\pm 0.0089}$

To validate the analysis regarding the effects of zero-mean and non-zero-mean activation functions on AOPU's performance, an additional comparative experiment was conducted. This experiment included 20 independent repetitions for activation functions classified into zero-mean, Hard Shrink, Tanh, Tanh Shrink, Soft Sign, and Soft Shrink, and non-zero-mean, Sigmoid, Relu6, RRelu, Hardswish, and Mish. The results are listed in Table 4

The experimental results largely confirmed the hypotheses outlined previously. In the zero-mean group, whether on the Debutanizer or SRU dataset, fluctuations in MSE, MAPE, and R^2 metrics were consistently controlled within 1% (with the maximum R^2 fluctuation being 0.48%, from 0.7236 to 0.7201). Conversely, in the non-zero-mean group, Sigmoid, Relu6, and RRelu all demonstrated notable performance declines on the Debutanizer dataset. Notably, although Hard Swish and Mish are classified as non-zero-mean activation functions, they did not negatively impact AOPU's performance. This could likely be attributed to the fact that Hard Swish and Mish are approximately zero-mean near the zero index, unlike Sigmoid, Relu6, and RRelu, which are non-zero-mean across any arbitrary small neighborhoods.

4   Conclusion and Limitation

This paper introduces a novel NN regression model, AOPU, which is grounded in solid mathematics basis and validated through extensive experiments. The results demonstrate its superior performance, robustness, and training stability. The development of AOPU lays the foundation for the practical implementation of deep learning soft sensor techniques in industrial processes and provides guidance for subsequent control, monitoring, and optimization management of these processes. The introduction of RR also illuminates a promising and valuable direction for exploring the design of augmentation models. Such prospective topics of value encompass how to reduce the sensitivity of AOPU to batch size and sequence length, how to derive the NG optimization of the augmentation model, and how to bolster the nonlinear modeling capability of the augmentation model.

We note that AOPU is not a "plug-and-play" model; it requires adjustments based on actual data conditions. AOPU necessitates a clear understanding of the RR distribution of data intended for application to guide the selection of batch size and sequence length hyperparameters. This requirement stems from the inherent matrix inversion operations in AOPU. When the RR value is too low, noise during the AOPU training process can greatly exceed the effective information, potentially leading to model divergence as Appendix F discusses.

References

[1] Xinyu, Z., G. Zhiqiang. Automatic deep extraction of robust dynamic features for industrial big data modeling and soft sensor application. *IEEE Transactions on Industrial Informatics*, 16(7):4456–4467, 2020.

[2] Xiaoxia, C., S. Xuhua, T. Chudong. Multi-time-scale tfe prediction for iron ore sintering process with complex time delay. *Control Engineering Practice*, 89:84–93, 2019.

[3] Xiaofeng, Y., Z. Jiao, H. Biao, et al. Hierarchical quality-relevant feature representation for soft sensor modeling: A novel deep learning strategy. *IEEE Transactions on Industrial Informatics*, 16(6):3721–3730, 2020.

[4] Xiaofeng, Y., H. Biao, W. Yalin, et al. Deep learning-based feature representation and its application for soft sensor modeling with variable-wise weighted sae. *IEEE Transactions on Industrial Informatics*, 14(7):3235–3243, 2018.

[5] Zhiqiang, G. Process data analytics via probabilistic latent variable models: A tutorial review. *Industrial and Engineering Chemistry Research*, 57(38):12646–12661, 2018. Doi: 10.1021/acs.iecr.8b02913.

[6] Zhang, Y., Q. Wen, x. wang, et al. Onenet: Enhancing time series forecasting models under concept drift by online ensembling. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, S. Levine, eds., *Advances in Neural Information Processing Systems*, vol. 36, pages 69949–69980. Curran Associates, Inc., 2023.

[7] Siwei, L., Y. Chunjie, Z. Xujie, et al. Blast furnace ironmaking process monitoring with time-constrained global and local nonlinear analytic stationary subspace analysis. *IEEE Transactions on Industrial Informatics*, pages 1–14, 2023.

[8] Daniel, S., U. Elisabeth. On multilevel best linear unbiased estimators. *SIAM/ASA Journal on Uncertainty Quantification*, 8(2):601–635, 2020.

[9] Peter, G., J. Ramesh, R. Mohammad. Adaptive experimental design with temporal interference: A maximum likelihood approach. In *Advances in Neural Information Processing Systems*, vol. 33, pages 15054–15064. Curran Associates, Inc., 2020.

[10] Pinheiro, T. C. F., A. d. S. Silveira. Stochastic model predictive control using laguerre function with minimum variance kalman filter estimation. *International Journal of Dynamics and Control*, 11(3):1330–1350, 2023.

[11] Yuming, C., S.-A. Daniel, W. Rebecca. Autodifferentiable ensemble kalman filters. *SIAM Journal on Mathematics of Data Science*, 4(2):801–833, 2022.

[12] Chong, Y., Y. Chunjie, L. Junfang, et al. Forecasting of iron ore sintering quality index: A latent variable method with deep inner structure. *Computers in Industry*, 141:103713, 2022.

[13] Yong, L., W. Haixu, W. Jianmin, et al. Non-stationary transformers: Exploring the stationarity in time series forecasting. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, A. Oh, eds., *Advances in Neural Information Processing Systems*, vol. 35, pages 9881–9893. Curran Associates, Inc., 2022.

[14] Elad, H., L. Holden, S. Karan, et al. Spectral filtering for general linear dynamical systems. In *Advances in Neural Information Processing Systems*, vol. 31. Curran Associates, Inc., 2018.

[15] Elad, H., S. Karan, Z. Cyril. Learning linear dynamical systems via spectral filtering. In *Advances in Neural Information Processing Systems*, vol. 30. 2017.

[16] Peter, V. O., D. M. Bart. N4sid: Subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*, 30(1):75–93, 1994. Special issue on statistical signal processing and control.

[17] Gianluigi, P., A. Aleksandr, G. Daniel, et al. Deep networks for system identification: A survey. *arXiv preprint arXiv:2301.12832*, 2023.

[18] Ilya, L., H. Frank. Decoupled weight decay regularization. In *International Conference on Learning Representations*. 2019.

[19] Kingma, D. P., J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2017.

[20] James, H., R. Magnus, L. Neil. Fast variational inference in the conjugate exponential family. In *Advances in Neural Information Processing Systems*, vol. 25. Curran Associates, Inc., 2012.

[21] Amari, S. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998. Yv501 Times Cited:1681 Cited References Count:41.

[22] James, M. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21, 2020. Np3tx Times Cited:69 Cited References Count:73.

[23] Matthew, D. H., M. B. David, W. Chong, et al. Stochastic variational inference. *Journal of Machine Learning Research*, 14:1303–1347, 2013. 168md Times Cited:1175 Cited References Count:103.

[24] Hensman, J., N. Fusi, N. D. Lawrence. Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*, 2013.

[25] Matthias, B., v. d. W. Mark, R. Carl Edward. Understanding probabilistic sparse gaussian process approximations. In *Advances in Neural Information Processing Systems*, vol. 29. Curran Associates, Inc., 2016.

[26] Trong Nghia, H., H. Quang Minh, L. Bryan Kian Hsiang. A unifying framework of anytime sparse gaussian process regression models with stochastic variational inference for big data. In F. Bach, D. Blei, eds., *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37 of *Proceedings of Machine Learning Research*, pages 569–578. PMLR, Lille, France, 2015.

[27] Dutordoir, V., J. Hensman, M. van der Wilk, et al. Deep neural networks as point estimates for deep gaussian processes. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, J. W. Vaughan, eds., *Advances in Neural Information Processing Systems*, vol. 34, pages 9443–9455. Curran Associates, Inc., 2021.

[28] Wu, L., K. Mohammad Emtiyaz, S. Mark. Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. In K. Chaudhuri, R. Salakhutdinov, eds., *Proceedings of the 36th International Conference on Machine Learning*, vol. 97 of *Proceedings of Machine Learning Research*, pages 3992–4002. PMLR, 2019.

[29] Wu, L., N. Frank, E. Khan Mohammad, et al. Tractable structured natural-gradient descent using local parameterizations. In M. Meila, T. Zhang, eds., *Proceedings of the 38th International Conference on Machine Learning*, vol. 139 of *Proceedings of Machine Learning Research*, pages 6680–6691. PMLR, 2021.

[30] Mohammad, K., L. Wu. Conjugate-computation variational inference : Converting variational inference in non-conjugate models to inferences in conjugate models. In A. Singh, J. Zhu, eds., *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, vol. 54 of *Proceedings of Machine Learning Research*, pages 878–887. PMLR, 2017.

[31] James, M., G. Roger. Optimizing neural networks with kronecker-factored approximate curvature. In F. Bach, D. Blei, eds., *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37 of *Proceedings of Machine Learning Research*, pages 2408–2417. PMLR, Lille, France, 2015.

[32] Wu, L., D. Felix, E. Runa, et al. Can we remove the square-root in adaptive gradient methods? a second-order perspective. *arXiv preprint arXiv:2402.03496*, 2024.

[33] —. Structured inverse-free natural gradient: Memory-efficient and numerically-stable kfac for large neural nets. *arXiv preprint arXiv:2312.05705*, 2023.

[34] Qingqiang, S., G. Zhiqiang. A survey on deep learning for data-driven soft sensors. *IEEE Transactions on Industrial Informatics*, 17(9):5853–5866, 2021.

[35] Luigi, F., G. Salvatore, R. Alessandro, et al. *Soft Sensors for Monitoring and Control of Industrial Processes*, vol. 22. Springer, 2007. Springer.

[36] Zhiqiang, G., H. Biao, S. Zhihuan. Mixture semisupervised principal component regression model and soft sensor application. *AIChE Journal*, 60(2):533–545, 2014.

[37] Yuan, X., L. Li, Y. Wang. Nonlinear dynamic soft sensor modeling with supervised long short-term memory network. *IEEE Transactions on Industrial Informatics*, 16(5):3168–3176, 2020.

[38] Yuan, X., Y. Wang, C. Yang, et al. Stacked isomorphic autoencoder based soft analyzer and its application to sulfur recovery unit. *Information Sciences*, 534:72–84, 2020.

[39] Chong, Y., Y. Chunjie, Z. Xinmin, et al. Multisource information fusion for autoformer: Soft sensor modeling of feo content in iron ore sintering process. *IEEE Transactions on Industrial Informatics*, 19(12):11584–11595, 2023.

[40] Feng, Y., Y. Chunjie, Z. Xinmin, et al. A 3-d convolution-based burn-through point multistep prediction model for sintering process. *IEEE Transactions on Industrial Electronics*, 71(4):4219–4229, 2024.

[41] Zhichao, C., G. Zhiqiang. Knowledge automation through graph mining, convolution, and explanation framework: A soft sensor practice. *IEEE Transactions on Industrial Informatics*, 18(9):6068–6078, 2022.

[42] Haoyi, Z., Z. Shanghang, P. Jieqi, et al. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):11106–11115, 2021.

[43] Albert, G., G. Karan, R. Christopher. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

[44] Zhichao, C., S. Zhihuan, G. Zhiqiang. Variational inference over graph: Knowledge representation for deep process data analytics. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–16, 2023.

[45] Yi, T., B. Dara, M. Donald, et al. Synthesizer: Rethinking self-attention for transformer models. In M. Meila, T. Zhang, eds., *Proceedings of the 38th International Conference on Machine Learning*, vol. 139 of *Proceedings of Machine Learning Research*, pages 10183–10192. PMLR, 2021.

[46] James, W., B. Viacheslav, T. Alexander, et al. Efficiently sampling functions from gaussian process posteriors. In H. D. III, A. Singh, eds., *Proceedings of the 37th International Conference on Machine Learning*, vol. 119 of *Proceedings of Machine Learning Research*, pages 10292–10302. PMLR, 2020.

[47] Zhiqiang, G., S. Zhihuan. Nonlinear soft sensor development based on relevance vector machine. *Industrial and Engineering Chemistry Research*, 49(18):8685–8693, 2010. Doi: 10.1021/ie101146d.

[48] Lim, S. H. Understanding recurrent neural networks using nonequilibrium response theory. *Journal of Machine Learning Research*, 22(47):1–48, 2021.

[49] Zachary C., L., B. John, E. Charles. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[50] Ailing, Z., C. Muxi, Z. Lei, et al. Are transformers effective for time series forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(9):11121–11128, 2023.

[51] Chen, C. L. P., Z. Liu. Broad learning system: An effective and efficient incremental learning system without the need for deep architecture. *IEEE Transactions on Neural Networks and Learning Systems*, 29(1):10–24, 2018.

[52] Malik, A., R. Gao, M. Ganaie, et al. Random vector functional link network: Recent developments, applications, and future directions. *Applied Soft Computing*, 143:110377, 2023.

[53] Zhengyuan, Z., M. Panayotis, B. Nicholas, et al. Stochastic mirror descent in variationally coherent optimization problems. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett, eds., *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[54] —. On the convergence of mirror descent beyond stochastic convex programming. *SIAM Journal on Optimization*, 30(1):687–716, 2020.

[55] Chaobing, S., Z. Zhengyuan, Z. Yichao, et al. Optimistic dual extrapolation for coherent non-monotone variational inequalities. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin, eds., *Advances in Neural Information Processing Systems*, vol. 33, pages 14303–14314. Curran Associates, Inc., 2020.

[56] Bo, D., H. Niao, D. Hanjun, et al. Provable bayesian inference via particle mirror descent. In A. Gretton, C. C. Robert, eds., *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, vol. 51 of *Proceedings of Machine Learning Research*, pages 985–994. PMLR, Cadiz, Spain, 2016.

[57] Luigi, F., G. Salvatore, R. Alessandro, et al. *Soft sensors for monitoring and control of industrial processes*. Springer, London, UK, 2007.

A Network's mechanism

In this section, we first demonstrate that AOPU implements a MVE through NN, explain the relationship between data augmentation, minimum variance, and orthogonal projection. We then discuss the physical significance and necessity of the dual parameter from NGD perspective.

A.1 From MVE perspective

We start by giving the definition of the MVE,

Definition 1. *Given the independent variable x and the dependent variable y, $f^*(y|x)$ is said to be the MVE for y if the following hold,*

$$\mathrm{E}[(y - f^*(y|x))^2] \leq \mathrm{E}[(y - f(y|x))^2] \tag{7}$$

where $\mathrm{E}[\cdot]$ denotes the expectation operator, and $f(y|x)$ represents any unbiased arbitrary estimation function for y given x.

According to definition 1, it is straightforward that the MVE is the optimal unbiased estimator under the Mean Squared Error loss metric, providing a performance boundary for all regression networks. However, the solution to the MVE, detailed in Appendix C.1, represented as $\int y p(y|x) dy$ where $p(\cdot)$ denotes the probability operator, is challenging to determine. Since the prior knowledge of the likelihood distribution $p(y|x)$ is not accessible, this integral is difficult to solve. Instead of delving into modeling the likelihood, AOPU turns to referencing solvable linear MVE operators for network design. Specifically, when the function form of $f(y|x)$ is constrained to be linear with respect to x, it can be set as $f(y|x) = W_{mve}x + b$, with the solution $W_{mve} = \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1}$ and $b = \mathrm{E}[y] - W_{mve}\mathrm{E}[x]$, where R_{ab} represents the covariance matrix $\mathrm{E}\left[(a - \mathrm{E}[a])(b - \mathrm{E}[b])^T\right]$. For clarity, the proof is listed in Appendix C.2.

Given that variables x and y have been normalized to have zero mean, i.e., $\mathrm{E}[x] = 0$ and $\mathrm{E}[y] = 0$, it follows that $b = 0$, and the covariance operator R degenerates into an inner product operation. Revisiting the loss function of AOPU, it is evident from Eq. 6 that AOPU essentially estimates the parameters W_{mve} of the linear MVE. Here the $(\tilde{x}^T\tilde{x})^{-1}$ is aligned with the $\mathrm{R}_{\tilde{x}\tilde{x}}^{-1}$, and $\tilde{x}^T D(\tilde{W})$ ought correspond to the estimation of the input-output covariance matrix. It is noteworthy that the parameter W_{mve} is not the estimator's output. Therefore, by approximating W_{mve} in the loss function, AOPU implies an important assumption: unlike other regression algorithms that explicitly model the mapping relationship from $\tilde{x}$ to y, i.e., $\tilde{x} \to y$, AOPU implicitly models the relationship $\tilde{x} \to \tilde{x}y$. Given that $\tilde{x} \in \mathbb{R}^{d+h,b}$ and $y \in \mathbb{R}^{b,1}$, the key to deriving y from known $\tilde{x}$ and $\tilde{x}y$ lies in the requirement that $\tilde{x}$ must be column-full-rank. This requirement aligns with the numerical stability needs during the computation process of AOPU, thereby establishing a self-consistent mathematical framework for the unit.

The geometric interpretation of the linear minimum variance estimator as orthogonal projection underpins the naming of the AOPU. AOPU differs from the orthogonal projection in threefold: (1) Orthogonal projection is a non-parametric batch algorithm, whereas AOPU operates as a parametric, gradient-based mini-batch optimization algorithm. (2) Orthogonal projection requires the covariance matrix's inverse to exist definitively, whereas AOPU can employ approximate inverses for its computations. (3) Orthogonal projection strictly adheres to linear minimum variance estimation, but AOPU introduces non-linearity through data augmentation, allowing it to serve as a versatile minimum variance estimator. The data augmentation techniques illustrated in Fig. 1 are critical for enhancing the expressive capabilities of AOPU. We initially improve model expressiveness through a fixed, randomly initialized Gaussian matrix $\hat{G}$. However, this approach remains confined within the linear transformation. Consequently, in subsequent experiments, the inputs are further augmented using LeakyRelu($\hat{G}^T x$), pushing the model beyond linear transformations.

A.2 From NGD perspective

In this section, we introduce NGD, including the computation of FIM, how to reduce the computational complexity of NGD through EM, and ultimately demonstrate that the truncated gradient of AOPU is an approximated NG.

We begin with an introduction to the most basic optimization algorithm used in neural network training, Gradient Descent (GD). Assuming the network parameters are represented as λ (λ represents $\tilde{W}$ in AOPU), and ∇ denotes the gradient, GD can be defined by the following equation:

$$\text{GD:} \quad \lambda_{t+1} = \lambda_t - \alpha \nabla_{\lambda_t} \mathcal{L}(\lambda_t) \tag{8}$$

where $\alpha > 0$ represents the network's learning rate. Many optimization algorithms assist the network in escaping local optima and accelerating convergence by incorporating momentum gradients and adaptive learning rates. However, fundamentally, these are first-order optimization methods (considering only first-order derivatives) and typically exhibit suboptimal performance in practice. NG, by calculating the FIM, exploits the information geometry of the model's output distribution to accelerate convergence more effectively. NGD can be defined by the following equation:

$$\text{NGD:} \quad \lambda_{t+1} = \lambda_t - \beta F(\lambda_t)^{-1} \nabla_{\lambda_t} \mathcal{L}(\lambda_t) \tag{9}$$

where β, often set to 1, is a scale factor. $F(\lambda_t)$ is the negative expectation of the second-order derivative regarding the network parameter λ_t, expressed as $F(\lambda_t) = -\mathrm{E}_{p(\hat{y})}[\nabla^2_{\lambda_t} \log p(\hat{y}|\lambda_t)]$. The essence of the FIM is to act as a preconditioning matrix that properly scales the gradient on the manifold of the parameter space. The primary difference between NGD and GD is that FIM accounts for dependencies among parameters, unlike the assumption of independent gradients for each parameter as in Eq. 8 with α. NGD generally exhibits better performance compared to GD when FIM is well-defined.

The computational cost of calculating the FIM and its inverse is substantial, i.e., cubic in terms of time complexity and quadratic in terms of space complexity. However, for specific model structures, the computation of NGD can be as straightforward as that of GD. We provide the definition of the exponential family as follows,

$$p(z|\lambda) := h(z) \exp\left[\langle \phi(z), \lambda \rangle - A(\lambda)\right] \tag{10}$$

where $p(z|\lambda)$ is defined as an exponential family function with the structure given on the right-hand-side, $h(z)$ is the base measure, $\phi(z)$ is the sufficient statistics, λ is the natural parameter, and $A(\lambda)$ is the log-partition function. The expectation-parameter is defined as the expected value of the sufficient statistics, expressed as $m(\lambda) = \mathrm{E}_{p(z|\lambda)}[\phi(z)]$. For exponential family distributions, we propose the following proposition,

Proposition 2. *The FIM of $p(z|\lambda)$ with respect to λ is equivalent to the gradient of the expectation parameter $m(\lambda)$ with respect to λ.*

Proof. *In Appendix C.4.*

Therefore, the NGD can be simply expressed as follows:

$$\text{NGD:} \quad \lambda_{t+1} = \lambda_t - \beta \nabla_{m(\lambda_t)} \mathcal{L}(\lambda_t) \tag{11}$$

By utilizing the MSE loss function, we can establish a connection between AOPU and NGD. From Eq. 6, it is known that the output of AOPU is the inner product of the model parameter $\tilde{W}$ and the augmented variable $\tilde{x}$. Supposed the output is viewd as an Gaussian distribution characterized by mean $\tilde{x}^T \tilde{W}$ and covariance I, where I is an appropriately dimensioned identity matrix. The following optimization objective can be proven to be equivalent to MSE:

$$\begin{aligned} \hat{\mathcal{L}}(\tilde{W}) &= \mathrm{E}_{\mathcal{N}(\hat{y}|\tilde{x}^T\tilde{W}, I)}[(y - \hat{y})^T(y - \hat{y})] \\ &= (y - \hat{y})^T(y - \hat{y}) + \mathrm{Tr}(I) \\ &\approx \mathrm{MSE}(y, \hat{y}) \end{aligned} \tag{12}$$

where $\approx$ denotes the equivalence up to constants.

Proposition 3. *The NG of $\hat{\mathcal{L}}(\tilde{W})$ with respect to $\tilde{W}$ is approximately equivalent to the gradient of $\hat{\mathcal{L}}(D(\tilde{W}))$ with respect to dual parameter $D(\tilde{W})$.*

Proof. *In Appendix C.5*

Proposition 3 valids the effectiveness of using truncated gradients for parameter updates in AOPU. Fact that $\hat{\mathcal{L}}(D(\tilde{W}))$ is actuallt the aforementioned loss function $\mathcal{L}$, and it can only be an approximation of $\hat{\mathcal{L}}(\tilde{W})$ because the inverse of $\tilde{x}^T \tilde{x}$ may not be well-defined as discussed in sections 2.3 and A.1. Therefore, RR not only serves as a measure of AOPU's numerical stability and its approximation degree to MVE but is also employed to quantify how closely truncated gradient approximates the NG.

B Convergence Analysis

In this section, we are going to analyze the convergence of AOPU referencing the conclusions from [53, 54]. We eventually demonstrate that under the condition $\tilde{x}$ is column-full-rank, AOPU converges to the optimal solution almost surely. Firstly, thanks to the trackability of parameters, AOPU is capable of being proven a coherent optimization problem under a strict assumption, which agrees on previous analysis, is made about the distribution of the observed samples y during the proof; AOPU's truncated gradient is then proven to structurally ensure consistency with the Stochastic Mirror Descent (SMD), specifically that the dual parameters correspond directly to the mirror map; the assumptions in [53] about regularity (assumption 3), differentiability (assumption 1), and bounded second moments with Lipschitz continuity (assumption 2) are also proven to be met under the condition $\tilde{x}$ is column-full-rank. Finally, by referencing theorem 3.4 from [53], we prove that AOPU can enter arbitrarily small neighborhood of the optimal parameter solution $\tilde{\mathcal{W}}^*$.

Definition 2. *The optimization problem* $\min \tilde{\mathcal{L}}(W)$ *is said to be coherent if*

$$\left\langle \nabla \mathrm{E}[\tilde{\mathcal{L}}(W)], W - W^* \right\rangle \geq 0 \quad \text{for all } W \in \mathcal{W}, W^* \in \mathcal{W}^* \tag{13}$$

with equality holds if and only if $W \in \mathcal{W}^*$, *where* $\mathcal{W}$ *is the feasible parameter space and* $\mathcal{W}^*$ *is additionally constrainted by* $\mathcal{W}^* = \arg\min \tilde{\mathcal{L}}$.

Proposition 4. *If the observed sample* y *is characterized by an underlying* $D(\tilde{W})^* \in \mathcal{D}(\tilde{\mathcal{W}})^*$, *D for short, AOPU's training objective is coherent with respect to* D.

Proof. *In Appendix C.6*

Referring the work of [53, 55], Definition 2 introduces the concept **coherent** which involves the analysis of the first-order derivatives of the loss function. For conventional NN, due to the untrackability of parameters, analyzing parameter gradients is exceedingly challenging. Although some studies [56, 32, 31] have explored the parameters' local characteristics, analyzing their global properties remains difficult. Proposition 4 summarizes the properties of coherence of AOPU briefly.

Proposition 5. *If the regularizer in* Q *is characterized by square Mahalanobis distance with covariance matrix* $\Sigma = \tilde{x}\tilde{x}^T$ *instead of Euclidean distance* $\Sigma = I$, *AOPU's training strategy is identical to SMD.*

Proof. *In Appendix C.7*

In SMD, each iteration involves calculating the stochastic gradient from the model's current state, updating within the dual space, and then mapping back to the parameter space. This process is outlined in Algorithm 1, where X and Y represent the parameters and dual parameters, respectively, and subscripts denote the iteration number. Q represents the mirror map, which is defined as follows,

$$Q(Y) = \arg\max_X \langle X, Y \rangle - h(X) \tag{14}$$

where h acts as a regularizer. Proposition 5 summarizes the connection between AOPU's training and SMD by carefully selecting regularizer.

Algorithm 1 Stochastic mirror descent

Require: Initial Y_0
 1: $n \leftarrow 0$
 2: **repeat**
 3:     $X_n = Q(Y_n)$
 4:     $Y_{n+1} = Y_n - \alpha_{n+1}\nabla\tilde{\mathcal{L}}(X_n)$
 5:     $n \leftarrow n + 1$
 6: **until** end
 7: **return** solution candidate X_n

The assumption about regularity fundamentally guarantees the continuity of the Fenchel coupling at the point where Y equals the subgradient of h with respect to X, and can typically be considered trivially satisfied. Regarding the properties of differentiability and bounded second moments and

Lipschitz continuity, the objective function of AOPU is typically quadratic, thus both its first and second derivatives exist and are linearly related to $(\tilde{x}^T\tilde{x})^{-1}$. These assumptions are only satisfied when $\tilde{x}$ is column-full-rank, i.e., when the inverse is well-defined.

Combining the conclusions from the proofs discussed above with theorem 3.4 from [53], it can be determined that AOPU's dual parameters $D(\tilde{W})$ always converge to the optimal values under the constraints. Essentially, the primary difference between using $\tilde{W}$ for inference and $D(\tilde{W})$ for training is term $(\tilde{x}^T\tilde{x})^{-1}$. Therefore, when $\tilde{x}$ is column-full-rank, the convergence of the dual parameters $D(\tilde{W})$ is equivalent to the convergence of the parameters $\tilde{W}$. This conclusion is consistent with the results from the earlier MVE analysis and highlights the important role of RR.

The proof of convergence complements the final piece of the puzzle for the deployment of AOPU in advanced industrial applications. It provides solid theoretical support in the aspects of derivation processes, optimization procedures, state monitoring, and convergence assurance. It can be confidently stated that AOPU is ready to be applied in industrial soft sensing, having established robust foundations for operational reliability and efficacy.

C   Mathematic Proof

C.1   Solution to General Minimum Variance Estimator

In this subsection we are about to prove that given x, the solution to the general minimum variance estimator of y is $\int y p(y|x) dy$, i.e., $\mathrm{E}_{y|x}[y]$. Since it is the expectation of likelihood, this result is intuitive to prove. Rewrite the covariance calculation in the following,

$$
\begin{aligned}
&\mathrm{E}\left[(y - f(y|x))(y - f(y|x))^T\right] \\
=&\mathrm{E}\left[(y + \mathrm{E}_{y|x}[y] - \mathrm{E}_{y|x}[y] - f(y|x))(y + \mathrm{E}_{y|x}[y] - \mathrm{E}_{y|x}[y] - f(y|x))^T\right] \\
=&\mathrm{E}\left[(y - \mathrm{E}_{y|x}[y])(y - \mathrm{E}_{y|x}[y])^T\right] + \mathrm{E}\left[(\mathrm{E}_{y|x}[y] - f(y|x))(\mathrm{E}_{y|x}[y] - f(y|x))^T\right] + \\
&\mathrm{E}\left[(y - \mathrm{E}_{y|x}[y])(\mathrm{E}_{y|x}[y] - f(y|x))^T\right] + \mathrm{E}\left[(\mathrm{E}_{y|x}[y] - f(y|x))(y - \mathrm{E}_{y|x}[y])^T\right]
\end{aligned}
\tag{15}
$$

Noting that $f(y|x)$ is not a conditional probabilistic distribution representation, it denotes a function that takes x as input, and the output of such function is regarded as an estimator of y. In conclusion, $f(y|x)$ is fundamentally independent of y, therefore, for the term $\mathrm{E}\left[(\mathrm{E}_{y|x}[y] - f(y|x))(y - \mathrm{E}_{y|x}[y])^T\right]$ we can rewrite it into,

$$
\begin{aligned}
&E\left[(\mathrm{E}_{y|x}[y] - f(y|x))(y - \mathrm{E}_{y|x}[y])^T\right] \\
=&\int\int (\mathrm{E}_{y|x}[y] - f(y|x))(y - \mathrm{E}_{y|x}[y])^T p(x,y) dx dy \\
=&\int (\mathrm{E}_{y|x}[y] - f(y|x))\left(\int (y - \mathrm{E}_{y|x}[y])^T p(y|x) dy\right) p(x) dx \\
=&\int (\mathrm{E}_{y|x}[y] - f(y|x))(\mathrm{E}_{y|x}[y] - \mathrm{E}_{y|x}[y])^T p(x) dx \\
=&0
\end{aligned}
\tag{16}
$$

The conclusion also applies to the term $\mathrm{E}\left[(y - \mathrm{E}_{y|x}[y])(\mathrm{E}_{y|x}[y] - f(y|x))^T\right]$. Consequently, the last two terms in Eq. 16 consistently equal zero. Given that $\mathrm{E}\left[(\mathrm{E}_{y|x}[y] - f(y|x))(\mathrm{E}_{y|x}[y] - f(y|x))^T\right]$ is semi-positive definite, it follows that $\mathrm{E}\left[(y - \mathrm{E}_{y|x}[y])(y - \mathrm{E}_{y|x}[y])^T\right]$ establishes a lower bound for $\mathrm{E}\left[(y - f(y|x))(y - f(y|x))^T\right]$. Equality holds if and only if $f(y|x) = \mathrm{E}_{y|x}[y]$ which completes the proof.

C.2   Solution to Linear Minimum Variance Estimator

In this subsection, we are about to prove that the solution to the linear minimum variance estimator is $W_{mve} = \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1}$ and $b = E[y] - W_{mve}E[x]$. Initially, it is straightforward to see that the value of b renders the estimator unbiased. By simply taking the expectation, we can complete the proof. Again

we rewrite the covariance calculation in the following,

$$\begin{aligned}
&\mathrm{E}\left[(y - f(y|x))(y - f(y|x))^T\right] \\
&=\mathrm{E}\left[(y - \mathrm{E}[y] + W_{mve}\mathrm{E}[x] - W_{mve}x)(y - \mathrm{E}[y] + W_{mve}\mathrm{E}[x] - W_{mve}x)^T\right]
\end{aligned} \tag{17}$$

Incorporating the covariance matrices $\mathrm{R}_{yy} = \mathrm{E}\left[(y - \mathrm{E}[y])(y - \mathrm{E}[y])^T\right]$, $\mathrm{R}_{xx} = \mathrm{E}\left[(x - \mathrm{E}[x])(x - \mathrm{E}[x])^T\right]$, and $\mathrm{R}_{xy} = \mathrm{E}\left[(x - \mathrm{E}[x])(y - \mathrm{E}[y])^T\right]$. We can reformulate Eq. 17 as $\mathrm{R}_{yy} + W_{mve}\mathrm{R}_{xx}W_{mve}^T - \mathrm{R}_{yx}W_{mve}^T - W_{mve}\mathrm{R}_{xy}$. Upon simplification, this equation transforms to,

$$\begin{aligned}
&\mathrm{R}_{yy} + W_{mve}\mathrm{R}_{xx}W_{mve}^T - \mathrm{R}_{yx}W_{mve}^T - W_{mve}\mathrm{R}_{xy} \\
&=W_{mve}\mathrm{R}_{xx}W_{mve}^T - W_{mve}\mathrm{R}_{xx}\mathrm{R}_{xx}^{-1}\mathrm{R}_{xy} - \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1}\mathrm{R}_{xx}W_{mve}^T + \mathrm{R}_{yy} + \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1}\mathrm{R}_{xy} - \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1}\mathrm{R}_{xy} \\
&=(W_{mve} - \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1})\mathrm{R}_{xx}(W_{mve} - \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1})^T + \mathrm{R}_{yy} - \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1}\mathrm{R}_{xy}
\end{aligned} \tag{18}$$

Noting that $(W_{mve} - \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1})\mathrm{R}_{xx}(W_{mve} - \mathrm{R}_{yx}\mathrm{R}_{xx}^{-1})^T$ is again semi-positive definite, indicating that the optimal W_{mve} is identical to $\mathrm{R}_{yx}\mathrm{R}_{xx}^{-1}$, which completes the proof.

C.3   Proof to Proposition 1

Suppose there exists an operator T independent of x such that for a given W and any inputs x_1 and x_2, the Eq. 2 holds. From the linearity property of operators, it follows that,

$$\begin{aligned}
\mathrm{acti}(Wx_1) + \mathrm{acti}(Wx_2) &= T(W)(x_1 + x_2) \\
\mathrm{acti}(Wx_1) + \mathrm{acti}(Wx_2) &= \mathrm{acti}(W(x_1 + x_2))
\end{aligned} \tag{19}$$

Due to the nonlinearity of the activation function, it is clear that Eq. 19 doesn't hold, consequently completing the proof.

C.4   Proof to Proposition 2

We are about to give concise and precise proof in this section, starting by proving the connection between the expectation-parameter and the log-partition function.

Proposition 6. *The expectation-parameter equals to the gradient of log-partition function with respect to natural parameter.*

Proof. *Since EM represents a probability distribution, the log-partition function acts as a normalizing factor, thus the following identity holds true,*

$$A(\lambda) = \log \int h(z) \exp\left(\langle \phi(z), \lambda \rangle\right)dz. \tag{20}$$

Therefore, expectation-parameter could be derived from differentiating $A(\lambda)$ with respect to λ.

$$\begin{aligned}
\nabla_\lambda A(\lambda) &= \nabla_\lambda \log \int h(z) \exp\left(\langle \phi(z), \lambda \rangle\right)dz \\
&= \frac{\nabla_\lambda \int h(z) \exp\left(\langle \phi(z), \lambda \rangle\right)dz}{\int h(z) \exp\left(\langle \phi(z), \lambda \rangle\right)dz} \\
&= \frac{\nabla_\lambda \langle \phi(z), \lambda \rangle \left(\int h(z) \exp\left(\langle \phi(z), \lambda \rangle\right)dz\right)}{\int h(z) \exp\left(\langle \phi(z), \lambda \rangle\right)dz} \\
&= \mathrm{E}_{p(z|\lambda)}\phi(z)
\end{aligned} \tag{21}$$

Clearly, $A(\lambda)$ is the only term that is second-order derivable in the score function $\log p(z|\lambda)$. The FIM can then be intuitively derived from its definition.

$$\begin{aligned}
F(\lambda) &= -\mathrm{E}_{p(z|\lambda)}[\nabla_\lambda^2 \log p(z|\lambda)] \\
&= -\mathrm{E}_{p(z|\lambda)}[-\nabla_\lambda^2 A(\lambda)] \\
&= \nabla m(\lambda)
\end{aligned} \tag{22}$$

C.5   Proof to Proposition 3

We first reiterate that treating the output $g(\hat{y}|x)$ as a Gaussian distribution is merely a prior assumption and does not alter the structure or computation of AOPU. Representing this Gaussian distribution as the minimal EM can be expressed as follows,

$$
\begin{aligned}
\mathcal{N}(\hat{y}|\tilde{x}^T\tilde{W}, I) &= (2\pi)^{-\frac{d+h}{2}} |I|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}\left\langle \hat{y} - \tilde{x}^T\tilde{W}, \hat{y} - \tilde{x}^T\tilde{W} \right\rangle\right] \\
&= (2\pi)^{-\frac{d+h}{2}} |I|^{-\frac{1}{2}} \exp\left[\left\langle \tilde{x}\hat{y}, \tilde{W} \right\rangle - \frac{1}{2}\left(\hat{y}^T\hat{y} + \tilde{W}^T\tilde{x}\tilde{x}^T\tilde{W}\right)\right]
\end{aligned}
\tag{23}
$$

Under this representation, the sufficient statistics and the natural parameter are respectively $\tilde{x}\hat{y}$ and $\tilde{W}$. From this, by the definition of the expectation parameter, we can calculate $m(\tilde{W}) = \mathrm{E}_{\mathcal{N}(\hat{y}|\tilde{x}^T\tilde{W}, I)}[\tilde{x}\hat{y}]$ equals to $\tilde{x}\tilde{x}^T\tilde{W}$ which is exactly identical to $D(\tilde{W})$. According to proposition 2, the FIM with respect to $\tilde{W}$ is equivalent to the gradient of $D(\tilde{W})$ with respect to $\tilde{W}$. Thus, by introducing $D(\tilde{W})$, we can accelerate the NGD computation with respect to $\tilde{W}$ as shown below,

Note that $\hat{\mathcal{L}}(\tilde{W})$ is equivalent to MSE. To compute the gradient of $\hat{\mathcal{L}}$ at $D(\tilde{W})$ using automatic differentiation tools and avoid complex algebraic operations, we design $\hat{\mathcal{L}}(D(\tilde{W}))$ as,

$$
\hat{\mathcal{L}}(D(\tilde{W})) = \mathrm{E}_{\mathcal{N}(\hat{y}|(\tilde{x}^T\tilde{x})^{-1}\tilde{x}^T D(\tilde{W}), I)}[(y-\hat{y})^T(y-\hat{y})]
\tag{24}
$$

Clearly, $\hat{\mathcal{L}}(D(\tilde{W}))$ is identical to the $\mathcal{L}$ introduced in section 2.3.

C.6   Proof to Proposition 4

We now assume the observed sample y is fully characterized by dual parameter D addition with a zero-mean random variable ϵ. Such constraint implies that there exists an optimal parameter set $\mathcal{D}^*$ which fully captures the mean trend of y, i.e., $y = (\tilde{x}^T\tilde{x})^{-1}\tilde{x}^T\mathcal{D}^* + \epsilon$. The coherence definition could be rewritten as follows,

$$
\begin{aligned}
&\left\langle \nabla\mathrm{E}[\hat{\mathcal{L}}(D)], D - D^* \right\rangle \\
=&\mathrm{E}\left[\left\langle \nabla\hat{\mathcal{L}}(D), D - D^* \right\rangle\right] \\
=&\mathrm{E}\left[\left\langle (\tilde{x}^T\tilde{x})^{-1}\tilde{x}^T D - y, (\tilde{x}^T\tilde{x})^{-1}\tilde{x}^T(D - D^*) \right\rangle\right] \\
=&\mathrm{E}\left[\left\langle (\tilde{x}^T\tilde{x})^{-1}\tilde{x}^T D - y, (\tilde{x}^T\tilde{x})^{-1}\tilde{x}^T(D - D^*) - y + y \right\rangle\right] \\
=&\mathrm{E}\left[\hat{\mathcal{L}}(D)\right] + \mathrm{E}\left[\left\langle (\tilde{x}^T\tilde{x})^{-1}\tilde{x}^T D - y, y - (\tilde{x}^T\tilde{x})^{-1}\tilde{x}^T D^* \right\rangle\right] \\
=&\mathrm{E}\left[\hat{\mathcal{L}}(D) - \hat{\mathcal{L}}(D^*)\right] + \mathrm{E}\left[\left\langle (\tilde{x}^T\tilde{x})^{-1}\tilde{x}^T(D - D^*), \epsilon \right\rangle\right]
\end{aligned}
\tag{25}
$$

In Eq. 25, the first equation arises due to the linear invariance of the gradient with respect to expectation. The second equation is derived by expanding the objective function and calculating its gradient, followed by reorganization. The third equation results from adding and subtracting the same variable y on the right-hand side of the second equation. The fourth equation reconstructs the objective function and cross-terms from the third equation. The fifth equation reconstructs the objective function under optimal parameter settings from the fourth equation.

Note that the objective function under globally optimal parameter settings is necessarily less than or equal to the objective function under any other parameter settings, thus term $\mathrm{E}[\hat{\mathcal{L}}(D) - \hat{\mathcal{L}}(D^*)] \geq 0$. Given the previous assumption that each instance within the optimal parameter set perfectly captures the trend in y, the second term on the right-hand side of the fifth equation is equivalent to the previously defined zero-mean random variable ϵ, and hence the second expected value is identically zero. Thus, it is proven that AOPU's optimization is coherent.

C.7   Proof to Proposition 5

Using $dist_M$ and $dist_E$ represent Mahalanobis distance and Euclidean distance respectively we have $dist_E(x, y; \Sigma) = \sqrt{x^T\Sigma^{-1}y}$ and $dist_E(x, y) = \sqrt{x^Ty}$. The major difference between them is that
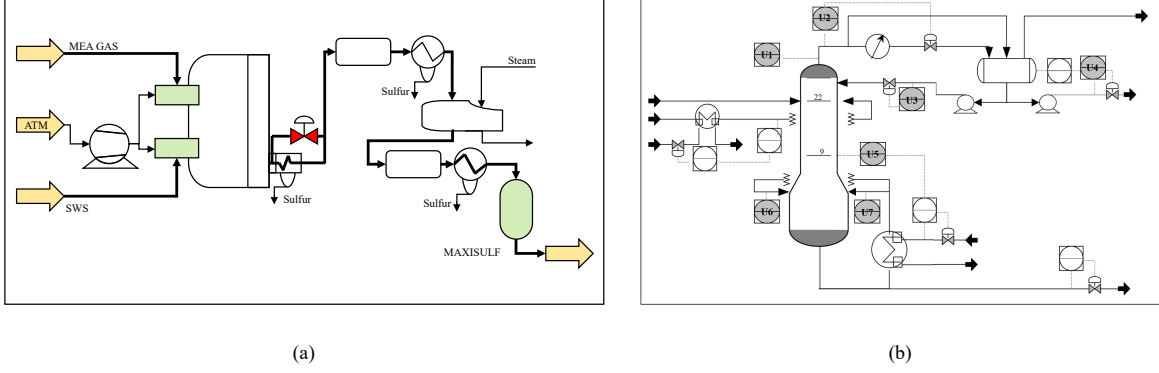
Figure 7: Schematic diagram of two industrial process. (a) SRU. (b) Debutanizer.

the former adjusts for the distribution of data across different dimensions. Euclidean distance is essentially the Mahalanobis distance when the covariance matrix is I (i.e., when dimensions are independent and identically distributed). Both Mahalanobis and Euclidean distances are strictly convex functions with respect to the input, making them suitable for use as regularizer terms in mirror maps. Referring to Algorithm 1, we have revised the training strategy for AOPU, presented in Algorithm 2. It is evident that both share a consistent optimization structure, thus structurally ensuring that AOPU's optimization process aligns with SMD. The key to the proof lies in establishing the relationship between the mirror map in SMD and the dual parameter in AOPU.

$$
\begin{aligned}
&\nabla \left[\left\langle \tilde{W}_n, D_n \right\rangle - h(D_n) \right] \\
=&\nabla \left[\left\langle \tilde{W}_n, D_n \right\rangle - \frac{1}{2} D_n^T (\tilde{x}\tilde{x}^T)^{-1} D_n \right] \\
=&\tilde{W}_n - (\tilde{x}\tilde{x}^T)^{-1} D_n
\end{aligned}
\tag{26}
$$

Algorithm 2 SMD in AOPU

Require: Initial $D(\tilde{W})_0$
1: $n \leftarrow 0$
2: **repeat**
3:     $D(\tilde{W})_n = Q(\tilde{W}_n)$
4:     $\tilde{W}_{n+1} = \tilde{W}_n - \eta \nabla \hat{\mathcal{L}}(D(\tilde{W})_n)$
5:     $n \leftarrow n + 1$
6: **until** end
7: **return** solution candidate $\tilde{W}_n$

Clearly, the mirror map is the solution where the gradient with respect to D_n is zero in $\left\langle \tilde{W}_n, D_n \right\rangle - h(D_n)$. Eq. 26 details this gradient computation process, where the first equation is obtained by incorporating the square Mahalanobis distance into the regularizer h, and the second equation is derived by differentiating with respect to D_n. The solution $D_n = \tilde{x}\tilde{x}^T\tilde{W}$ precisely matches the definition of the dual parameter in AOPU, confirming the coherence of AOPU's optimization strategy with the principles of SMD.

D Dataset Description

D.1 Debutanizer

The Debutanizer column is part of a desulfuring and naphtha splitter plant. It is required to maximize the C5 (stabilized gasoline) content in the Debutanizer overheads(LP gas splitter feed), and minimize

Table 5: Variable Description

Debutanizer			SRU		
Process Variables	Unit	Description	Process Variables	Unit	Description
U_1	$^\circ$C	Top temperature	U_1	$m^3 \cdot h^{-1}$	Gas flow MEA_GAS
U_2	kg·cm^{-2}	Top pressure	U_2	$m^3 \cdot h^{-1}$	Air flow AIR_MEA
U_3	$m^3 \cdot h^{-1}$	Reflux flow	U_3	$m^3 \cdot h^{-1}$	Secondary air flow AIR_MEA_2
U_4	$m^3 \cdot h^{-1}$	Flow to next process	U_4	$m^3 \cdot h^{-1}$	Gas flow in SWS zone
U_5	$^\circ$C	6th temperature	U_5	$m^3 \cdot h^{-1}$	Air flow in SWS zone
U_6	$^\circ$C	Bottom temperature A			
U_7	$^\circ$C	Bottom temperature B			

the C4 (butane) content in the Debutanizer bottoms (Naphtha splitter feed) [57]. However, the butane content is not directly measured on the bottom flow, but on the overheads of the downstream deisopentanizer column by the gas chromatograph resulting in a large measuring delay, which is the reason soft sensor steps in.

The dataset comprises 2,394 records, each featuring 7 relevant sensor measurements. The flowchart of the Debutanizer column, detailing the locations of these sensors and their respective descriptions, is presented in Fig. 7 (b). The corresponding details can also be found in Table 5.

D.2 Sulfur Recovery Unit

The sulfur recovery unit (SRU) removes environmental pollutants from acid gas streams before they are released into the atmosphere. The main chamber is fed with MEA gas, and combustion is regulated, in air deficiency, by supplying an adequate airflow (AIR_MEA). The secondary combustion chamber is mainly fed with SWS gas and a suitable air flow is provided (AIR_SWS). The combustion of SWS gas occurs in a separate chamber with excess air, in order to prevent the formation of ammonium salts in the equipment, thereby giving rise to the generation of nitrogen and nitrogen oxides. Air flows are controlled by plant operators to guarantee a correct stoichiometric ratio in the tail gas. Control is improved by a closed-loop algorithm which regulates a further airflow (AIR_MEA_2) on the basis of analysis of the tail gas composition. On-line analyzers are used to measure the concentration of both hydrogen sulfide and sulfur dioxide in the tail gas of each sulfur line. Hydrogen sulfide and sulfur dioxide frequently cause damage to sensors, which often have to be removed for maintenance. The design of soft sensors able to predict H2S and SO2 concentrations is therefore required.

The dataset contains 10,080 records with 5 relevant sensor measurements. The flowchart for the SRU is illustrated in Fig. 7 (a), with the input descriptions provided in Table 5

E Hyperparameter Scanning

The hyperparameter selection is guided by two principles: first, to ensure the model size of various comparative methods remains comparable; second, to choose hyperparameters that optimize model performance. Fig. 8 shows the hyperparameter scanning result, where we can see for SRU the smaller setup is recommended, while for Debutanizer bigger model possibly leads to better but still limited performance. However, the model size of the compared methods increases dramatically with layers and hidden dims, which means that the efficiency of parameters drops. Therefore, we chose hyperparameter settings that keep the model size comparable to that of AOPU, maintaining a balance between performance and efficiency. We also study the influence of the learning rate on the stable convergence. Fig. 9 shows that NGD outperforms the conventional gradient method on stable convergence under various learning rate setups, further validating AOPU's contributions.

F Supplementary Figure of Training Dynamics

In this section, we complete the comprehensive RR distribution experimental results and the stability of the training process of AOPU on the SRU and Debutanizer datasets. We note that, with a fixed sequence length and increasing batch size, the variance during training generally decreases for other models, as illustrated in Fig. 10 and Fig. 11. However, for AOPU, increasing bs actually diminishes
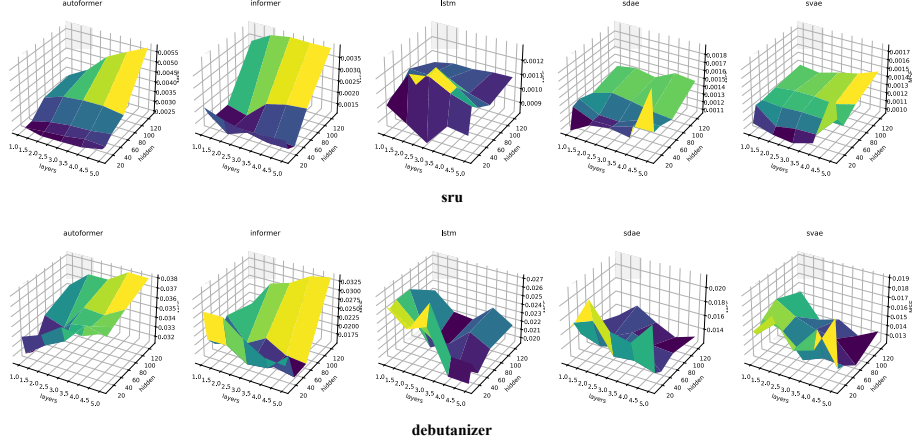
22

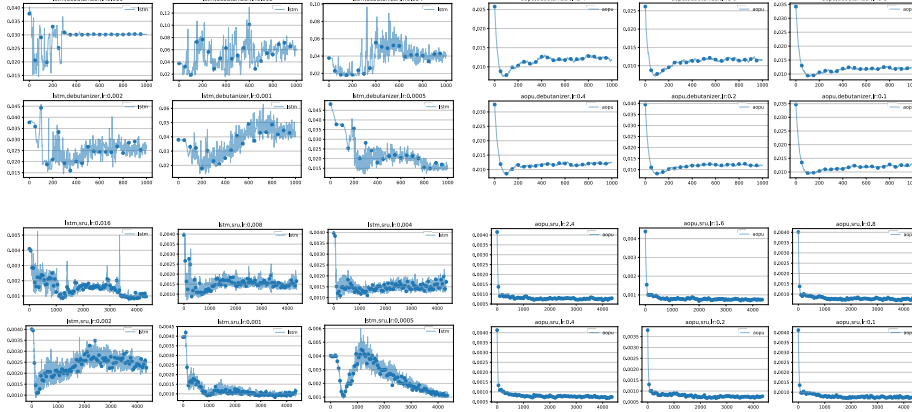Figure 8: Hyper parameter scanning result on SRU and Debutanizer.

Figure 9: Training dynamics under various learning rate setups.

performance. As seen in Fig. 10, when bs is 128 and seq is 16, AOPU's performance has declined compared to bs of 64 and seq of 16 in Fig. 6, characterized by increased fluctuations and an upward shift in the loss curve. Even more critical, Fig. 11 shows that when bs is 288 and seq is 16, AOPU encounters convergence issues.

This performance degradation is due to the gradual shift of the RR distribution toward zero as bs increases. In the AOPU structure design, RR serves not only as an indicator of numerical stability during the matrix's inversion in forward computation but also as a theoretical foundation for AOPU's recovery of y from $\tilde{x}y$. Therefore, lower RR correlates with poorer AOPU performance. From Fig. 5, we observe that with bs at 128 and seq at 16, the mean RR is around 0.5, while at BS of 288 and sequence length of 16, the mean RR distribution is near 0.25. Given that the RR distribution is highly concentrated, the mean value represents the statistical properties of the entire distribution, indicating that AOPU cannot converge when RR falls below 0.25.

Another interpretation of AOPU's lack of convergence is that when RR is too low (e.g., assuming RR is zero), AOPU's pseudo-inverse $(\tilde{x}^T \tilde{x})^{-1}$ has been calculated as $(\tilde{x}^T \tilde{x})$, effectively converting the original normalization process $(\tilde{x}^T \tilde{x})^{-1} \tilde{x}^T \tilde{x}$ into a squared process of $(\tilde{x}^T \tilde{x}) \tilde{x}^T \tilde{x}$, significantly deviating from the initial computational assumptions and causing model collapse. Similar observations can be found in the Debutanizer experiment results in Fig. 12, Fig. 13, Fig. 14, Fig. 15, and Fig. 16.

Figure 10: Curves of SRU validation loss changes with training iteration for different models with fixed batch size of 128 at different sequence length settings.
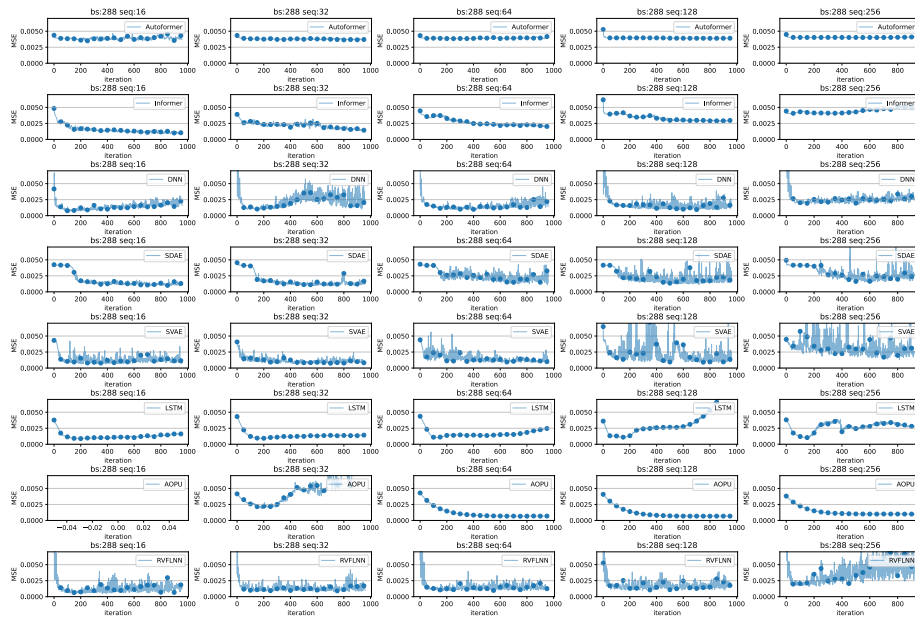

Figure 11: Curves of SRU validation loss changes with training iteration for different models with fixed batch size of 288 at different sequence length settings.
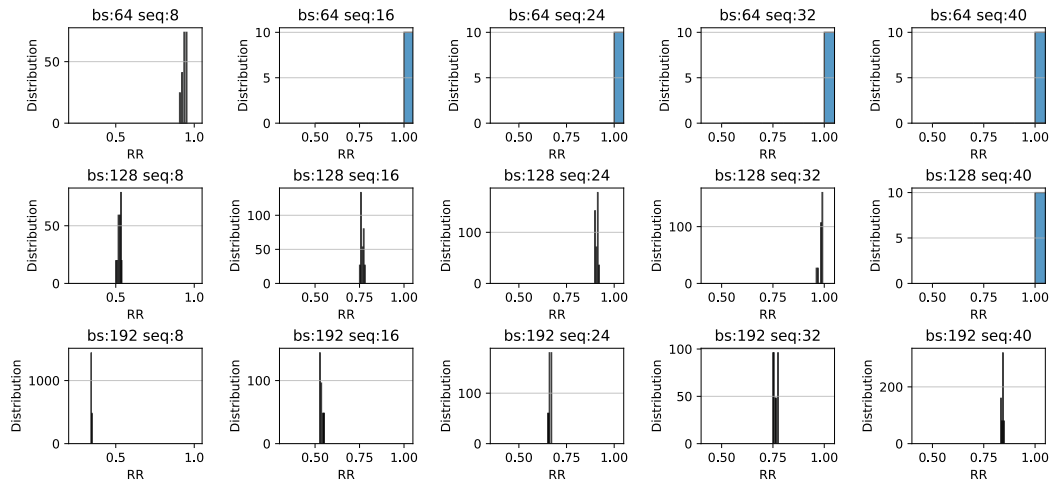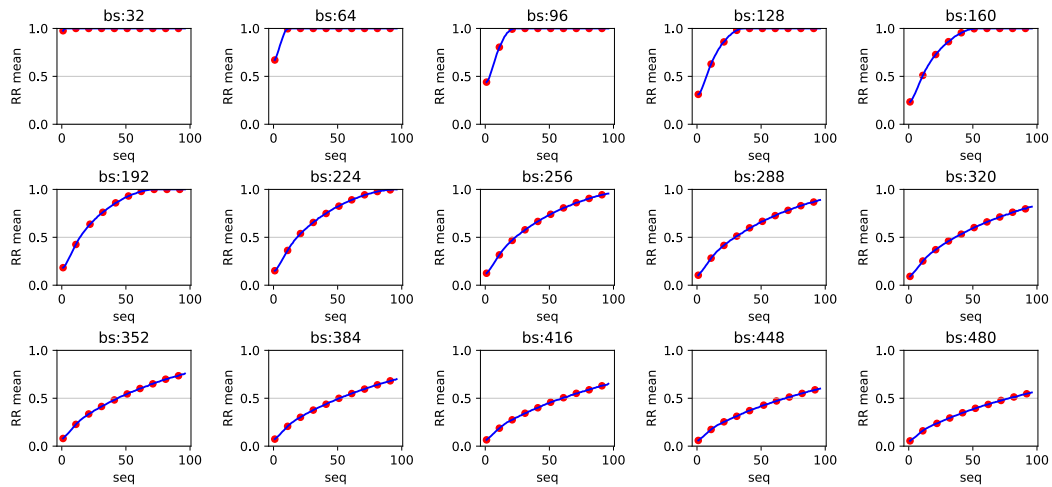
Figure 12: Histogram of the frequency distribution of RR on Debutanizer dataset under varying batch sizes and sequence length settings. In each subplot, the horizontal axis represents RR, while the vertical axis indicates frequency, with the distribution normalized. Subplots within the same column have the same sequence length, while subplots within the same row have the same batch size.

Figure 13: Curve of the mean of RR distribution on Debutanizer dataset under varying batch sizes and sequence length settings. In each subplot, the horizontal axis represents sequence length, while the vertical axis indicates the mean of RR distribution. The batch size increases from left to right and from top to bottom.
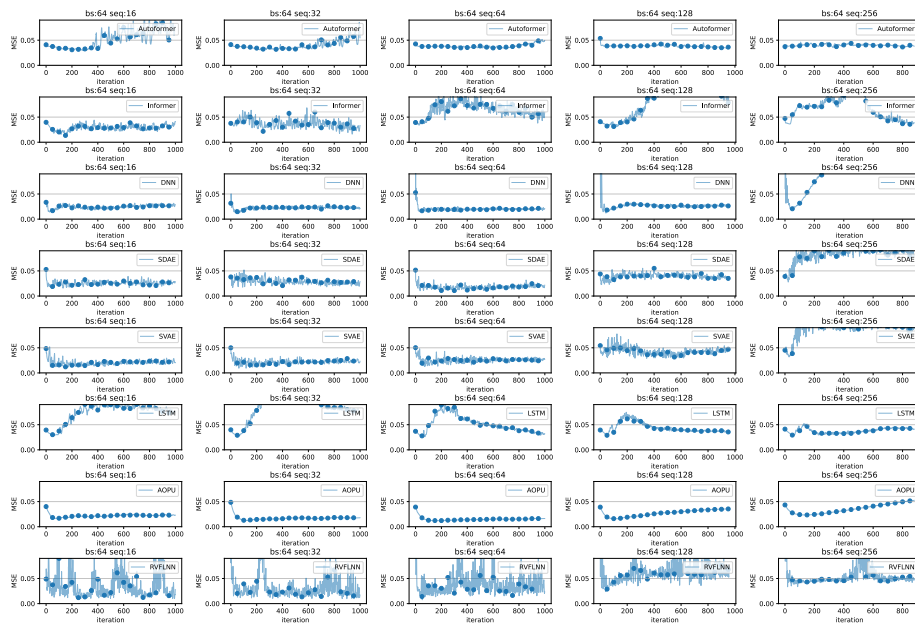
Figure 14: Curves of Debutanizer validation loss changes with training iteration for different models with a fixed batch size of 64 at different sequence length settings.
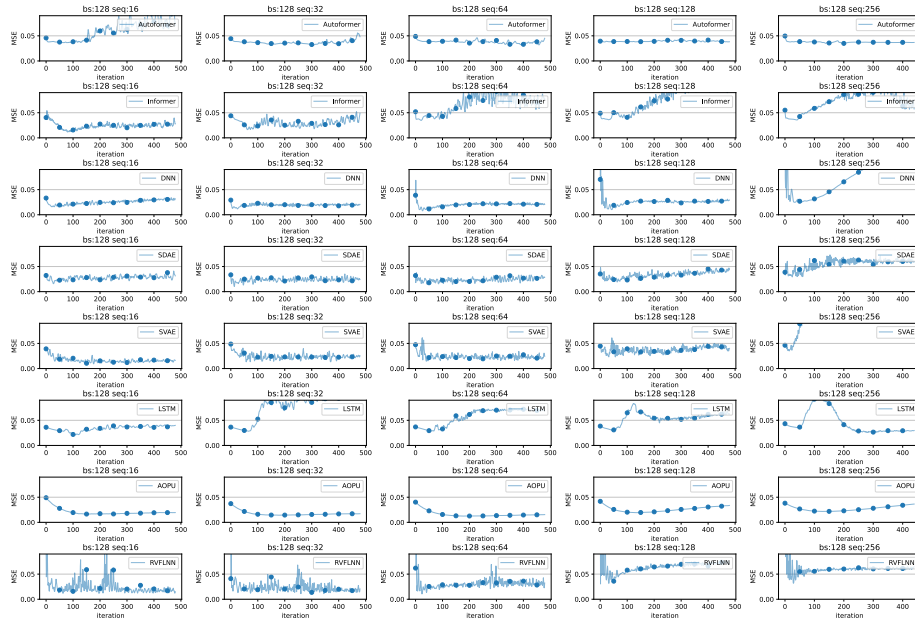
Figure 15: Curves of Debutanizer validation loss changes with training iteration for different models with a fixed batch size of 128 at different sequence length settings.
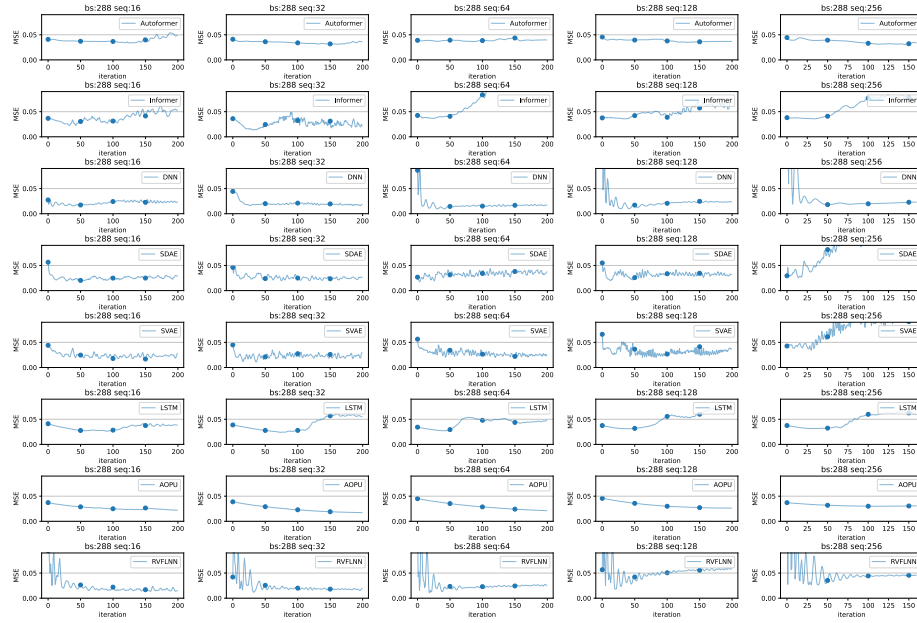
Figure 16: Curves of Debutanizer validation loss changes with training iteration for different models with a fixed batch size of 288 at different sequence length settings.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main idea of AOPU is an industrially usable stabilized neural network, whose model structure, mathematical support and experimental results, ablation experiments have been included.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We integrate limitation and conclusion to explore the limitations of AOPU compared to conventional neural networks in terms of batch size and sequence length selection constraint.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The proof have been given following each statement.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: All experiment details have been disclosed in section 3.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [NA]

   Justification: The authors have developed their work within an iron-making facility where both the code and data are treated as confidential. Despite these restrictions, we anticipate that the AOPU can be readily replicated by reviewers and future researchers. This is due to the model's straightforward and uncomplicated design, **NO** complex modules or intricate information exchanges between them. This paper focus on establishing a comprehensive framework to support the AOPU's effectiveness. Notably, replicating the model simply involves applying gradient truncation at the appropriate stages (dual parameter stage), underscoring its simplicity and efficiency.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All experiment details have been disclosed in section 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All experimental results were obtained after 20 repetitions. The standard deviation is also analyzed in section 3 to ensure that the results are significant

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Computer information are disclosed in section 3

Guidelines:
- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: This paper conforms the Code of Ethics.

Guidelines:
- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: AOPU is an industrial-directed usable neural network, which we believe has no potential societal impact.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: AOPU is an industrial-directed usable neural network, which we believe has no such risk.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: This paper have used the existing code repository: Autoformer, Informer; and dataset: Debutanizer, SRU, and we have cite the original paper.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: This paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.