Appendix for SRNet+

Yuanzhen Luo[1], Qiang Lu[1,*], Haoyang Li[1], Wei Wei[1], and Jake Luo[2]

[1] Beijing Key Laboratory of Petroleum Data Mining, China University of Petroleum, Beijing, China
[2] Department of Health Informatics and Administration, University of Wisconsin Milwaukee, Milwaukee, WI, USA
luqiang@cup.edu.cn;

1 Appendix A: Implement details

1.1 Network Structure

As shown in Figure 4 in our paper, we trained a MLP with 7 linear layers and LeNet with three convolutional modules. In Figure 4, “Linear-x” represents a linear layer with an output dimension of x. “ReLU-Drop” indicates passing through a ReLU activation layer followed by a Dropout layer. The Dropout layer randomly zeroes the outputs of neurons with a probability of 0.1 during training. d represents the output dimension of the dataset. “Conv2d-C6-K5” denotes a 2D convolutional layer with 6 output channels and a kernel size of 5; “BatchNorm2D” stands for batch normalization layer, “MaxPool-K2-S2” represents a max-pooling layer with a kernel size of 2 and a convolutional stride of 2. The LeNet designed for this experiment can be divided into 3 convolutional modules and 1 fully connected module. It’s worth noting that the deep fully connected model has 7 linear mapping layers, which for the SRNet model would require 7 symbolic layers to find interpretative formulas corresponding to each hidden layer. However, for SRNet+, after module partitioning, only 3 module symbolic layers are needed to represent the entire network model.

1.2 Training on MLP

For deep fully connected neural networks, to ensure fair comparison, SRNet+ maintains a consistent symbolic layer encoding structure with SRNet (CGP combined with linear mapping). Additionally, the training algorithm of SRNet+ is essentially similar to SRNet, except for some simplifications made to accelerate training as follows: SRNet+ employs the Adam optimizer to directly optimize the parameters of linear mapping and constants instead of quasi-Newton methods. Moreover, it adds an L1 regularization term to penalize the size of the linear mapping parameters in the module symbolic layer. The optimization training loss objective is as follows:

$$\begin{aligned} \mathcal{L}_{fc} &= fitness + \gamma \mathcal{R}(\Theta) \\ fitness &= \frac{1}{L-1}\sum_{i}^{L-1} \mathcal{L}_{mse}(\hat{\mathbf{h}}^i, \mathbf{h}^i) + \mathcal{L}_{mse}(\hat{\mathbf{h}}^L, \mathbf{h}^L) \end{aligned} \tag{1}$$

where $\mathcal{R}(\Theta)$ denotes the L1 regularization penalty term, γ represents the penalty coefficient, $\mathcal{L}_{mse}$ is the Mean Squared Error (MSE), $\hat{\mathbf{h}}^i$ and $\mathbf{h}^i$ is the output of the ith symbolic layer and ith network module respectively.

1.3 Hyperparameter settings

In the experiment for interpreting the deep fully connected network, to ensure a fair comparison with SRNet and demonstrate the performance of SRNet+ in module partitioning and training optimization algorithms, most of the hyperparameters for the fully connected module symbolic layer are kept consistent with SRNet, as shown in Table 2 from [1]. However, to expedite training, the settings of the following hyperparameters are adjusted: (1) The gradient descent part adopts the Adam optimizer instead of quasi-Newton methods, with a learning rate of 1e-3. (2) The number of training epochs is uniformly adjusted to 2000 epochs (with 300 epochs for fine-tuning on SRNet+ and 2000 epochs for overall fine-tuning). Additionally, the regularization penalty coefficient γ is set to 1e-5.

In the experiment for interpreting convolutional neural networks, SRNet+ adopts end-to-end gradient descent optimization. The neural network $NN_{\mathbf{W}}$ for symbolic layers consists of 5 linear mapping layers of dimension 100 each, followed by ReLU activation layers. To simplify the PDE formula, EQL utilizes a 2-layer fully connected structure. The input to EQL is pre-set partial differential terms, specifically $[u_x, u_y, u_{xx}, u_{yy}, u_{xy}]$. Each layer of EQL has candidate symbols $[+, \times, \sin, \cos, \mathrm{square}, \log]$. The regularization penalty coefficient γ for EQL is set to 1e-4, and α and β are set to 1.0 and 0.5, respectively. To prevent gradient explosion during training, a linear warm-up schedule strategy is employed for training EQL. That is, the learning rate for the tth epoch is set to $min(3e-3\times t/w, 3e-3)$, where w is the warm-up epochs, set to 10% of the total training epochs. Additionally, before updating EQL parameters, their gradients are clipped to a maximum normalized value of 1.0. Considering the relatively large size of the MNIST dataset, a batch size of 512 is set for this experiment.

1.4 Definition of Module Attribution

We first divide neural network into two main functional areas as follows:

Convolutional Functional Area: which only contains a series of convolutional layers, normalization layers, and continuous subnetworks of single-value functions. The convolutional layers include: N-dimensional convolutional layer (ConvNd), N-dimensional pooling layer (PoolNd), where N $\in$ [1, 3]; normalization layers include commonly used batch normalization (BatchNorm), layer normalization (LayerNorm). Single-value functions include all defined activation functions and the Dropout layer. For example, in the CNN in Figure **??**, {Conv2d, Activation, Pool2d, Conv2d, Activation, Pool2d} constitutes a convolutional functional area.

Fully Connected Functional Area: which only contains a series of linear layers, normalization layers, and continuous subnetworks of single-value functions. The linear layers are linear mapping layers (Linear). For example, in the

CNN in Figure 2, Linear, Activation, Linear, Activation, Linear, Softmax constitutes a fully connected functional area.

To find modules with the same structure and parameters, AutoPUM first defines corresponding attributes for each network module separately, and extracts corresponding attribute values from the original network structure. Finally, network modules with the same attribute values are assigned the same module number and labeled as the same module (Add “(Cloned)” marker. For example, in the convolutional module SeqConv-1, the attribute channels represents the input and output channel numbers of each convolutional kernel, “kernel_sizes” represents the kernel sizes of each convolutional kernel; the fully connected module SeqLinear-1 has attributes “hidden_sizes” (dimensions of each linear layer mapping) and “n_layer” (number of linear layers in the module). We define the attributes of network module, as shown in Table 1

Module	Attribution	Statement
Convolution module	channels	number of input and output channels
	kernel_sizes	kernels' sizes in the module
	in_shape	input shape of the module
	out_shape	output shape of the module
	type	pre-defined module's type
Fully connected module	hidden_sizes	hidden sizes of all linear layer in the module
	n_layers	number of layers in the module
	type	pre-defined module's type

Table 1. The attributes definition of network module.

After assigning corresponding attribute values to each module, the module reuse stage checks the attributes of each module, and the modules with consistent attributes are assigned the same module number and labeled as the same module. For example, as shown in Figure 2 in our paper, the original SeqConv-1, SeqConv-3, and SeqConv-4 will all be reset and labeled as SeqConv-1, SeqConv-1 (Cloned), and SeqConv-1 (Cloned).Then we separately divide network modules in each functional area. Each network module is assigned with corresponding attributes.

2 Appendix B: Experiment results

2.1 Expression Analysis

To contrast the complexity of interpretable formulas obtained by SRNet+ and SRNet, Table 2 and 3 list the interpretable formulas of the first 4 and last 3 network layers of a deep fully connected network described by SRNet (detailed formulas for other datasets are provided in the appendix). Table 4 lists the interpretable formulas of various module symbolic layers describing the same fully connected network by SRNet+. It is noteworthy that SRNet comprises 7 symbolic layers, thus resulting in 7 interpretable formulas, whereas SRNet+ has only 3 symbolic layers, ultimately yielding 3 interpretable formulas.

Dataset	$f_\theta^1(\mathbf{x})$	$f_\theta^2(\hat{\mathbf{h}}^1)$
K0	$sin\left(0.84x + 0.15\right)$	$\left(\hat{h}^1\right)_5 - \left(\hat{h}^1\right)_6^2$
K1	$-sin\left(0.76x_0\right)$	$\left(\hat{h}^1\right)_4 / cos\left(\left(\hat{h}^1\right)_1\right)$
F0	$cos\left(x_0^{0.5}\right)$	$0.52 - \left(\hat{h}^1\right)_4$
F1	$x_3 - log\left(x_0 + x_1\right) - 1.28$	$tan\left(tan\left(\left(\hat{h}^1\right)_3\right)\right)$
Datasets	$f_\theta^3(\hat{\mathbf{h}}^2)$	$f_\theta^4(\hat{\mathbf{h}}^3)$
K0	$tan\left(\left(\hat{h}^2\right)_6\right) / cos\left(\left(\hat{h}^2\right)_7\right)$	$\left(\hat{h}^3\right)_6 - tan\left(\left(\hat{h}^3\right)_7\right)$
K1	$\left(\hat{h}^2\right)_1 + 2.0 * \left(\hat{h}^2\right)_7$	$-\left(\hat{h}^3\right)_1 + \left(\hat{h}^3\right)_7$
F0	$-\left(\hat{h}^2\right)_1^2 + \left(\hat{h}^2\right)_4$	$\left(\hat{h}^3\right)_1 - cos\left(\left(\hat{h}^3\right)_2 / \left(\hat{h}^3\right)_5\right)$
F1	$\left(\hat{h}^2\right)_0 - \left(\hat{h}^2\right)_6$	$0.08 * \left(\hat{h}^3\right)_4^2$

Table 2. Explanatory expressions for the **first 4** symbolic layers in SRNet

From Table 2 and 3, it can be observed that although the symbolic layer formulas obtained by SRNet have lower complexity, SRNet associates each symbolic layer with the interpretation of one network layer. Therefore, SRNet requires a total of 7 generic formulas for symbolic layers to interpret the entire deep neural network. When these 7 generic formulas for symbolic layers are nested, the overall formula describing the entire MLP becomes extremely complex, making it difficult to understand and read. Hence, we cannot show the complex overall formulas of SRNet for the current task.

In contrast to SRNet, Table 4 illustrates that SRNet+ requires only 3 module symbolic layers to generate 3 interpretable generic formulas to describe the entire MLP. Even when the 3 interpretable generic formulas of SRNet+ are nested, a relatively concise overall formula can still be obtained, as shown in Table 5. It can be observed that although the overall formula shown in Table 5 is still somewhat complex, it can intuitively describe which features and operators the MLP is based on for inference and prediction.

Through comparing and discussing the symbolic layer formulas and overall formulas of SRNet and SRNet+, it can be observed that under the influence of the AutoPUM, in the interpretation experiment of the same deep neural network, SRNet+ can use fewer symbolic layers (3) compared to the 7 symbolic layers of the SRNet model to obtain a more concise overall formula. This greatly enhances the interpretability of SRNet+.

Dataset	$f_\theta^5(\hat{\mathbf{h}}^4)$	$f_\theta^6(\hat{\mathbf{h}}^5)$
K0	$\left(\hat{h}^4\right)_1 / \left(\hat{h}^4\right)_5$	$-\left(\hat{h}^5\right)_{13} + \left(\hat{h}^5\right)_5^2$
K1	$-\left(\hat{h}^4\right)_9 + tan\left(\left(\hat{h}^4\right)_7\right)$	$-sin\left(\left(\hat{h}^5\right)_{13} - \left(\hat{h}^5\right)_4\right)$
F0	$-\left(\hat{h}^4\right)_{14} + cos\left(\left(\hat{h}^4\right)_9\right)$	$\left(\hat{h}^5\right)_5 - 1.23$
F1	$0.5 *\ log\left(\left(\hat{h}^4\right)_9\right)$	$\left(\hat{h}^5\right)_{13} / \left(\hat{h}^5\right)_0$
Datasets	$f_\theta^7(\hat{\mathbf{h}}^6)$	$\hat{y}$
K0	$-\left(\hat{h}^6\right)_2 + cos\left(\left(\hat{h}^6\right)_3\right)$	$-\left(\hat{h}^7\right)_1 + \left(\hat{h}^7\right)_7$
K1	$\left(\hat{h}^6\right)_0 - 1.3$	$-\left(\hat{h}^7\right)_1 + \left(\hat{h}^7\right)_6$
F0	$\left(\hat{h}^6\right)_1^2 + \left(\hat{h}^6\right)_1 + \left(\hat{h}^6\right)_2 + \left(\hat{h}^6\right)_6$	$tan\left(\left(\hat{h}^7\right)_4\right)$
F1	$tan\left(\left(\hat{h}^6\right)_5^2\right)$	$\left(\hat{h}^7\right)_1 * \left(\left(\hat{h}^7\right)_3 - \left(\hat{h}^7\right)_6\right)$

Table 3. Explanatory expressions for the **last 4** symbolic layers in SRNet

Dataset	$f_\theta^1(\mathbf{x})$	$f_\theta^2(\hat{\mathbf{h}}^1)$	$\hat{y}$
K0	$cos\left(Abs\left(x_1\right)^{0.5}\right)$	$cos\left(Abs\left(\left(\hat{h}^1\right)_5\right)^{0.5}\right)$	$Abs\left(\left(\hat{h}^2\right)_1\right)^{0.5}$ $/Abs\left(log\left(Abs\left(\left(\hat{h}^2\right)_3\right)\right)\right)$
K1	$0.7/Abs\left(x_0\right)$	$\left(\hat{h}^1\right)_4 * \left(\hat{h}^1\right)_5$	$\left(\hat{h}^2\right)_0$
F0	$0.16\ -\ x$	$\left(\hat{h}^1\right)_4 /Abs\left(\left(\hat{h}^1\right)_0\right)$	$\left(\hat{h}^2\right)_2 log\left(Abs\left(\left(\hat{h}^2\right)_3\right)\right)$ $/Abs\left(\left(\hat{h}^2\right)_3\right)$
F1	$cos\left(x_0 - cos\left(x_1\right)\right)$	$sin\left(\left(\hat{h}^1\right)_5\right)$	$-\left(\hat{h}^2\right)_2 + \left(\hat{h}^2\right)_4$

Table 4. Explainable expressions for the symbolic layers in SRNet+

References

1. Luo, Y., Lu, Q., Hu, X., Luo, J., Wang, Z.: Exploring hidden semantics in neural networks with symbolic regression. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 982–990 (2022)

Dataset	$S_{\mathbf{W}}(\mathbf{x})$
K0	$0.37\left(0.19\left(0.25x+0.01\right)/Abs\left(0.51x+0.66\right)+0.93\right)$ $log(Abs(1.02\left(0.25x+0.01\right)/Abs\left(0.51x+0.66\right)-0.43))$ $/Abs(1.02\left(0.25x+0.01\right)/Abs\left(0.51x+0.66\right)-0.43)-0.52$
K1	$1.85\ sin\left(1.08cos\left(x0-cos\left(x1\right)\right)-0.46\right)-0.12$
F0	$(3.89Abs\Big(0.76cos\Big(Abs\left(1.34*cos\left(Abs\left(x1\right)^{0.5}\right)+1.16\right)^{0.5}\Big)+0.55\Big)^{0.5}$ $+0.81Abs\Big(log\Big(Abs\Big(4.31cos\Big(Abs\left(1.34cos\left(Abs\left(x1\right)^{0.5}\right)+1.16\right)^{0.5}\Big)+5.59\Big)\Big)\Big))$ $/Abs\Big(log\Big(Abs\Big(4.31cos\Big(Abs\left(1.34cos\left(Abs\left(x1\right)^{0.5}\right)+1.16\right)^{0.5}\Big)+5.59\Big)\Big)\Big)$
F1	$-0.73*\left(Abs\left(x2\right)+0.07\right)^{0.5}/Abs\left(x2\right)^{0.5}+0.82$

Table 5. The overall explainable expressions of SRNet+