The Supplementary File for Scalable Learning to Optimize: A Learned Optimizer Can Train Big Models

Xuxi Chen[1⋆], Tianlong Chen[1*], Yu Cheng[2], Weizhu Chen[2]
Ahmed Awadallah[2], and Zhangyang Wang[1]

[1] The University of Texas at Austin, Austin TX 78712, USA
{xxchen,tianlong.chen,atlaswang}utexas.edu
[2] Microsoft Research
{yu.cheng,wzchen,hassanam}@microsoft.com

1 More Experiment Settings

For the analytical optimizers, we report the hyperparameter we use in Table 1. The architecture of SL2O we use is stemmed from a one-layer LSTM with a hidden size of 8. The inputs are fed into the LSTM model, followed by an MLP model and a `tanh` function to generate the prediction of the parameters' update.

2 More Experiment Analysis

As mentioned in the main manuscript, we will provide more analysis on experiment results to this supplementary file due to space limitations. The following analysis are **not** based on experiments on new datasets **nor** on new architectures.

2.1 Performance on ResNet-8

To cover a wider range of models, we examine our SL2O on an extremely small architecture ResNet-8, and we present the curves of training loss and testing accuracy in Figure 1. It shows that SL2O has promising results on small networks such as ResNet-8. The training loss converges the most rapidly among all methods, and the testing accuracy remains the highest.

2.2 Superior Performance on GPT

As mentioned in the main manuscript, we successfully deploy SL2O on a giant model GPT-2. We visualize the curves of training and validation loss in Figure 2. For better demonstration, we report both the training loss for the first 2000 and 80000 steps. The figures show 1) Although both methods show great variation in training loss, the overall trend of training loss is descending; 2) SL2O is more

⋆ Equal Contribution.

Table 1: The hyperparameter configurations of different optimizers.

Optimizer	Initial Learning Rate	Learning Rate Scheduler	Other Parameters
SGD	0.1	$\times 0.1$ at 50th,75th epochs	-
Momentum	0.1	$\times 0.1$ at 50th,75th epochs	$\gamma = 0.9$
Adam	0.001	$\times 0.1$ at 50th,75th epochs	$\beta_1 = 0.9, \beta_2 = 0.99$
RMSProp	0.01	$\times 0.1$ at 50th,75th epochs	-
SGD†	1	$\times 0.1$ at 30th,50th epochs	-
Momentum†	1	$\times 0.1$ at 30th,50th epochs	$\gamma = 0.9$

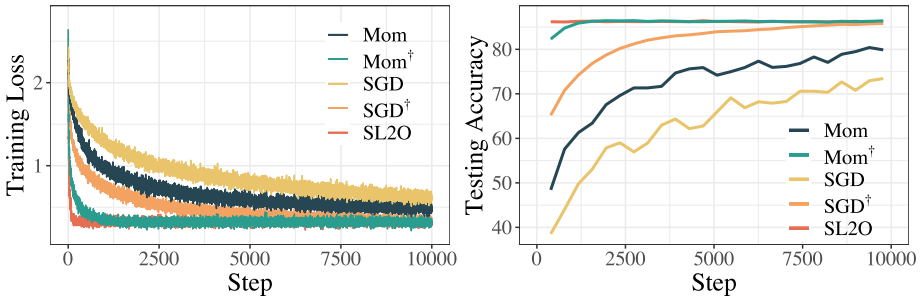

Fig. 1: Training loss and testing accuracy on ResNet-8 on CIFAR-10. We present the training loss and the testing accuracy in the first 10000 steps. The superscription † means that the model is updated in a subspace.

efficient at minimizing the training loss, particularly for the first 250 steps, shown by the steeper training loss curve. 3) The validation losses of SL2O match the validation losses of AdamW† along the training process. These observations show that the scalability roadblock of L2O has been removed.

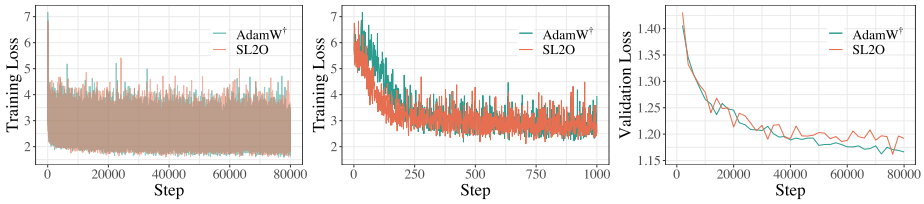

Fig. 2: Training and validation loss of GPT-2 on E2E. We report both the training loss in the full range and the first 2000 steps. The superscription † means that the model is updated in a subspace.

2.3 Superior Performance on DeiT

We prove that SL2O is also effective on different variants of Vision Transformers. We conduct experiments on `DeiT-base-distilled` and present both the training loss and the testing accuracy in Figure 3. From the figures, we can observe that SL2O achieves the highest testing accuracy (98.36%) while the most competitive baseline $\text{Mom}^{\dagger}_{0.1}$ can achieve only 98.24% after 20 training epochs. A bonus point is that SL2O shows high stability, demonstrated by the least fluctuated testing accuracy curve among all baselines. The convergence speed of SL2O continues to be fast, reaching a similar level as $\text{Mom}^{\dagger}_{0.1}$. In summary, these deferred results on ResNet-8, GPT-2, and DeiT further support the validness of applying L2O on models with larger scales that have never been studied in previous works.

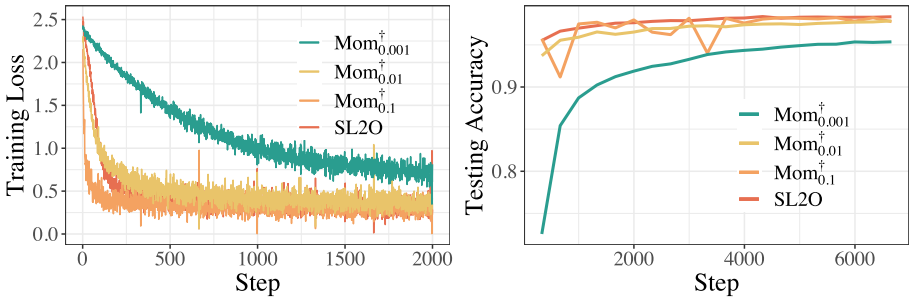

Fig. 3: Training loss and the testing accuracy on DeiT. We present the training loss in the first 2000 steps and the testing accuracy in the first 20 epochs. The superscription † means that the model is updated in a subspace, and the subscription indicates the learning rate.

2.4 Training ViT from Scratch?

We train ViT-B from scratch with SL2O, $\text{Mom}^{\dagger}$, and AdamW [2] on ImageNet. We report an intermediate result at 2k-th steps, where SL2O is 1.6% and 7.1% better than $\text{Mom}^{\dagger}$ and AdamW, showing a positive trend.

2.5 Symbolic Regression

We sample 10000 samples from the training iterations as the training set, and 2000 samples as the testing set. The input features for symbolic regression are $\tilde{\boldsymbol{m}}_t$ and $\tilde{\boldsymbol{g}}_t$, and the output feature is the predicted update $\Delta\boldsymbol{\theta}_t$ generated by SL2O. The atomic operators we use for symbolic regression are $+, -, \times, \div, **$ (power function),`abs`,`sign`,`logm`, `sqrtm`,`relu`,`exp`,`tanh`,`sinh`, `asinh` and `erfc`. . The first

Table 2: The mean-squared error on the training set and the Pearson's R on the testing set of different symbolic expressions.

Expression	Training Set Loss	Testing Pearson's R
$\Delta\boldsymbol{\theta}_t = 0.2326\texttt{tanh}(\tilde{\boldsymbol{g}}_t - 0.1094)$	0.0027	0.9495
$\Delta\boldsymbol{\theta}_t = 0.2097\texttt{exp}(\texttt{tanh}(\tilde{\boldsymbol{g}}_t))$ - 0.2523	0.0016	0.9689
$\Delta\boldsymbol{\theta}_t = 0.0627\texttt{exp}(\texttt{tanh}(\tilde{\boldsymbol{g}}_t + \tilde{\boldsymbol{m}}_t^3))$	0.0015	0.9697

seven operators are fundamental blocks in traditional optimizers and the rest are common in L2O; hence we adopt them to perform symbolic regression. On ResNet-20, we derive three symbolic expressions with different complexity from the training set and report the loss (*i.e.,* MSE loss) on the training set as well as the Pearson's R on the testing set. The results are shown in Table 2 and Figure 5 The extracted expressions extracted on the training samples can be generalized onto testing samples, demonstrated by the correlation values on the testing set. The presence of hyperbolic functions in the symbolic expressions (*i.e.,* `tanh`) is reasonable: 1) hyperbolic functions limit the response when the input features are large (refer to Figure 4) so they have a higher chance to reduce gradient noise and make these operators be more easily selected; 2) the `tanh` functions are being used as the non-linear activation function in the LSTM we deploy for SL2O, so its presence correctly reflects the non-linearity structure. From a perspective of expressions, these extracted equations are in general *linear* when the inputs are small and become *bounded and stable* when the inputs are large. Some non-linear structures are embedded in the extracted symbolic rules, which help stabilize the predicted gradient to limit them to a reasonable range.

In summary, the learned update rules can be greatly interpreted by symbolic expressions; hence our proposed optimizer benefits from high interpretability.

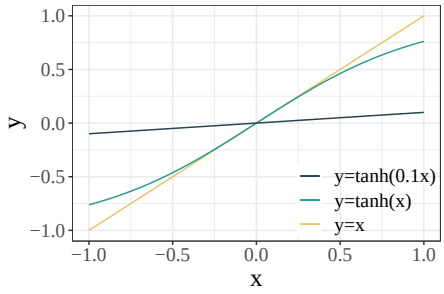

Fig. 4: Comparison between hyperbolic functions and linear functions. Hyperbolic functions limit the response when input features are large.

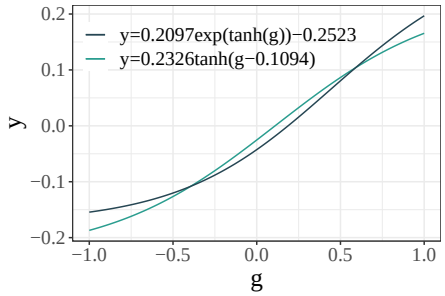

Fig. 5: Visualization of symbolic regression equations extracted from the training set. Both rules are deviated from the origin $(0, 0)$.

2.6 Transferability Between Datasets and Architectures

As shown below, our SL2O is transferable across various datasets. SL2O achieves a test accuracy of {93.54%, 74.39%} on CIFAR-100/10, which is only {0.07%, 0.28%} lower compared to its variants meta-trained on the same datasets. Moreover, SL2O trained on ImageNet also demonstrate an impressive transfer performance on CIFAR-10/100. We also conduct new experiments of transferring the learned optimizer among VGG-16, ResNet-18 and MobileNet as below. The performance gap after transfer is negligible ($< 0.1\%$), implying the sound transferability of SL2O across various types of architectures.

Table 3: Transfer performance between datasets and architectures. Transfer results are marked.

Meta-Training (ResNet-18)	Meta-Testing		Meta-Training (CIFAR-10)	Meta-Testing	
	CIFAR-10	CIFAR-100		VGG-16	ResNet-18
CIFAR-10	93.61%	74.39%	VGG-16	92.74%	93.54%
CIFAR-100	93.54%	74.67%	ResNet-18	92.66%	93.61%
ImageNet	93.58%	74.47%	MobileNet	92.64%	93.58%

2.7 Comparison With Different L2O methods

We compare SL2O with L2O-DM [1] to the same subspace of ResNet-20 on CIFAR-10. We have observed that SL2O reaches over 50% accuracy advantages in the meta-testing phase.

2.8 Detailed Analysis to Ablation Studies

The effects of unroll length. We study the effects of different unroll lengths T and we have observed that different unroll lengths T yield similar results. The experiments are conducted on ResNet-20 with four different $T = [5, 10, 15, 20]$. From Figure 6 we can see that, the unroll length T does not have a dominating effect on the performance in the meta-testing stage since the curves of training loss generated by SL2O with different unroll lengths are highly similar. Table 4 also validates that longer unroll lengths do not bring extra performance gain, while a smaller unroll length would result in slightly weak performance.

The effects of different training iterations. We study the effects brought by different training iterations on a certain optimization task and show the performance in the meta-testing stage by varying $N \in \{200, 500, 1000, 2000\}$. The

training loss curves are shown in Figure 6. Similarly, the training curves only exhibit a mild difference in the first 100 steps. In Table 4 we notice that a smaller N leads to an inferior result, showing that shorter training iterations probably make SL2O overfit to shorter training horizons, become "short-sighted", and thereby harms the optimizee's generalization ability.

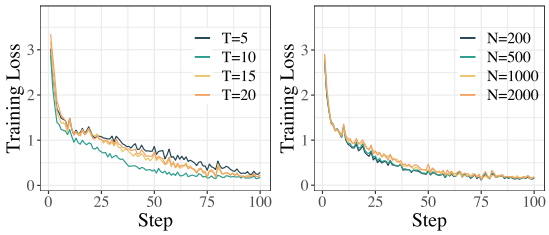

Fig. 6: Left: Training loss in the first 100 training steps with different T; Right: Training loss in the first 100 training steps with different N.

Table 4: The average of best testing accuracy with different unroll length T and training iterations N.

$T =$	5	10	15	20
Acc.	90.93%	91.00%	90.97%	90.92%
$N =$	200	500	1000	2000
Acc.	90.20%	90.70%	91.00%	90.97%

3 More Visualizations

We use two optimizers, Momentum† and SL2O, to optimize ResNet-20 on CIFAR-10 from the same initialization for 100 steps. For each optimizer, we record the updates on the coefficients of the first three principal directions ($\boldsymbol{u}_1$, $\boldsymbol{u}_2$, and $\boldsymbol{u}_3$) which account for most variability. The results are shown in Figure 7. Firstly, we can see that the updates generated by the Momentum† optimizer are small in magnitude, while our SL2O can generate updates with various scales. Secondly, SL2O can surprisingly predict overall decaying updates for the first three principal directions even without an explicit learning rate scheduler. The predicted updates are getting close to zeros after 75 steps, showing that SL2O tends to update models more cautiously when the training loss is getting low. Lastly, SL2O seems to prioritize the updates of the coefficient of $\boldsymbol{u}_2$ and $\boldsymbol{u}_3$ firstly, gradually switches to update the coefficient of $\boldsymbol{u}_1$, and eventually decades its prediction of updates to values of small magnitudes. This behavior suggests that SL2O can learn sophisticated update rules rather than always optimizing towards the largest principal direction.

References

1. Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M.W., Pfau, D., Schaul, T., Shillingford, B., De Freitas, N.: Learning to learn by gradient descent by gradient descent. In: Advances in neural information processing systems (NeurIPS) (2016)
2. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)

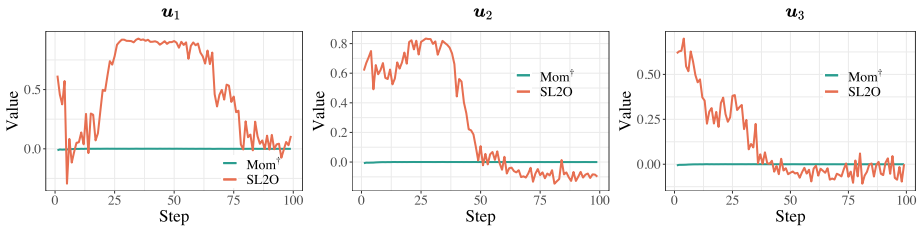

Fig. 7: The learned optimization rules for the largest three principal directions of extracted tiny subspaces.