Towards Robust Unroll Generalization in Learned Optimizers

Xiaolong Huang[1,2] Benjamin Thérien[2,3]
Eugene Belilovsky[1,2]

[1]Concordia University; [2]Mila – Quebec AI Institute; [3]Université de Montréal

Motivation

- Learned Optimizers (LOs) can discover superior update rules, achieving faster and better convergence than hand-designed optimizers on certain tasks.
- However, LOs often saturate quickly or diverge when evaluated on very long unrolls (e.g., $10\times$ larger than meta-training horizon).
- Extending unroll length during meta-training increases computational cost at least linearly and frequently leads to divergence due to compounding errors.

Meta-Training Pipeline

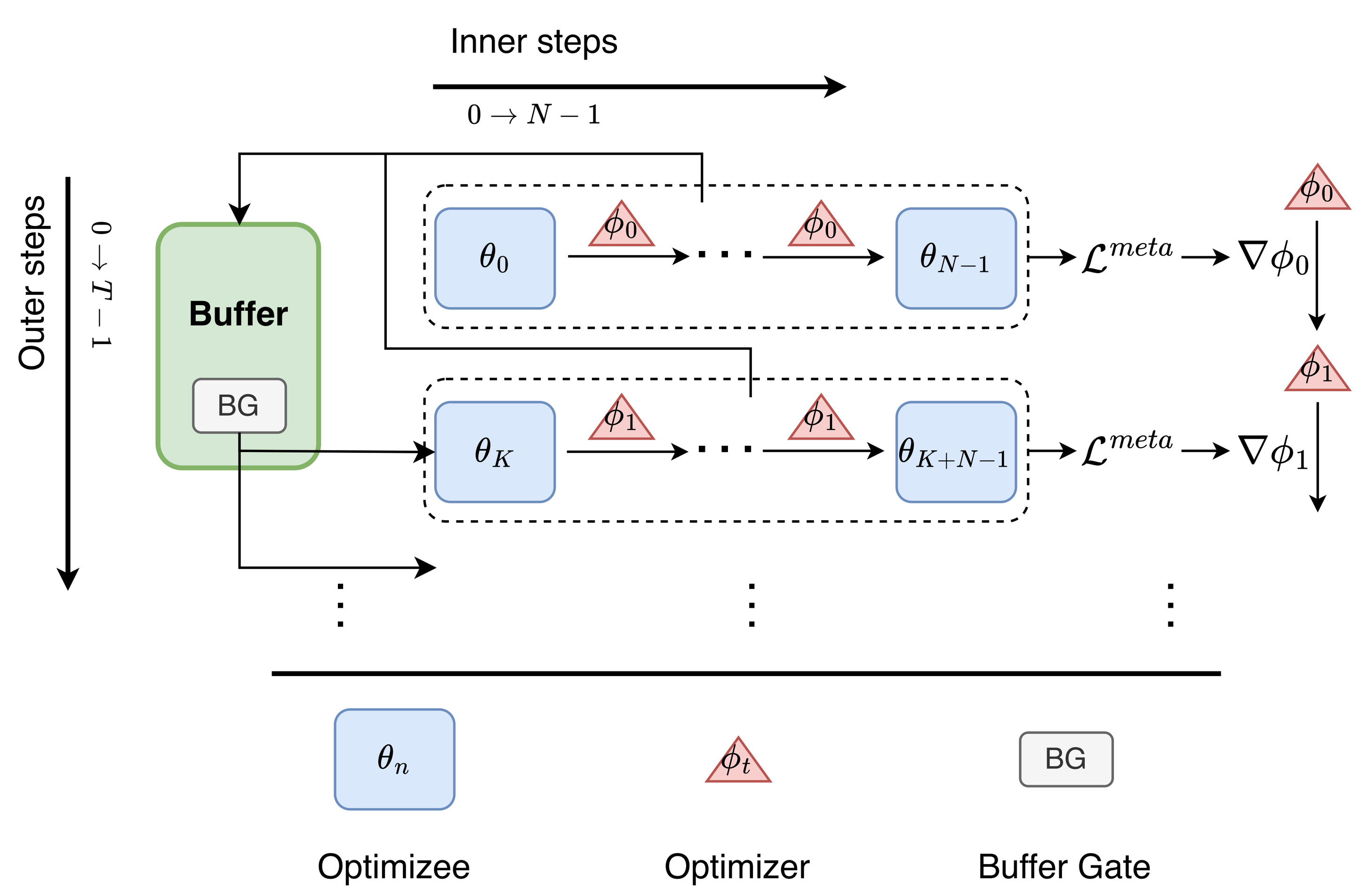

Figure 1. **Overview of meta-training a Learned Optimizer.** At each outer step t, the LO parameters ϕ_t update the optimizee parameters θ_n over N inner steps. Per-step meta-losses $\mathcal{L}_n^{meta}$ are accumulated, and the resulting meta-gradient updates $\phi_t \to \phi_{t+1}$.

ELO Algorithm

We propose **Efficient Long-horizon Learning (ELO)**: a novel meta-training scheme that leverages a buffer to efficiently extend unroll length during meta training without extra cost, along with an online behavior cloning strategy for stable meta-training.

Algorithm 1 Efficient Long-horizon Learning (ELO). Green: buffer operations. Yellow: behavior cloning.

Input: M Size of replay buffer
P_{th} Threshold for buffer initialization
α_t Mixing coefficient (increases $0 \to 1$)

```
Initialize replay buffer B = {s_1, ..., s_m}, m ≤ M
for outer step t = 0, 1, ..., T − 1 do                         ▷ outer loop
    Sample P_B ~ Uniform(0, 1)
    if P_B > P_th and t > 0 then
        Initialize from buffer: s_* ~ B (K ≠ 0)                 ▷ Apply buffer
    else
        Random initialization (K = 0)
    Sample N_push ∈ (K, N + K) ∩ Z
    for inner step n = K, ..., K + N − 1 do                    ▷ inner loop
        θ^H_{n+1} = θ_n + Δθ^H_n                                ▷ Adam update
        θ^O_{n+1} = θ_n + Δθ^O_n                                ▷ LO update
        θ_{n+1} = (1 − α_t) θ^H_{n+1} + α_t θ^O_{n+1}          ▷ Fused trajectory
        L^meta_n = (1 − α_t) L^bc_n + α_t L^task_n             ▷ Fused meta-loss
        s_{n+1} = (θ_{n+1}, ζ_{n+1})      ▷ Optimizee parameters, momentum, etc.
        if n == N_push then                                     ▷ Update buffer
            B ← { enqueue(B, s_n),             m < M
                { enqueue(dequeue(B), s_n),    m = M
    g_t = G( Σ_{n=K+1}^{K+N} L^meta_n(θ^H_n, θ^O_n; φ_t, α_t) )   ▷ Estimate meta gradient
    φ_{t+1} = U(g_t, t; φ_t)                                    ▷ Update LO
```

Sampling Efficiency & Meta-Training Stability

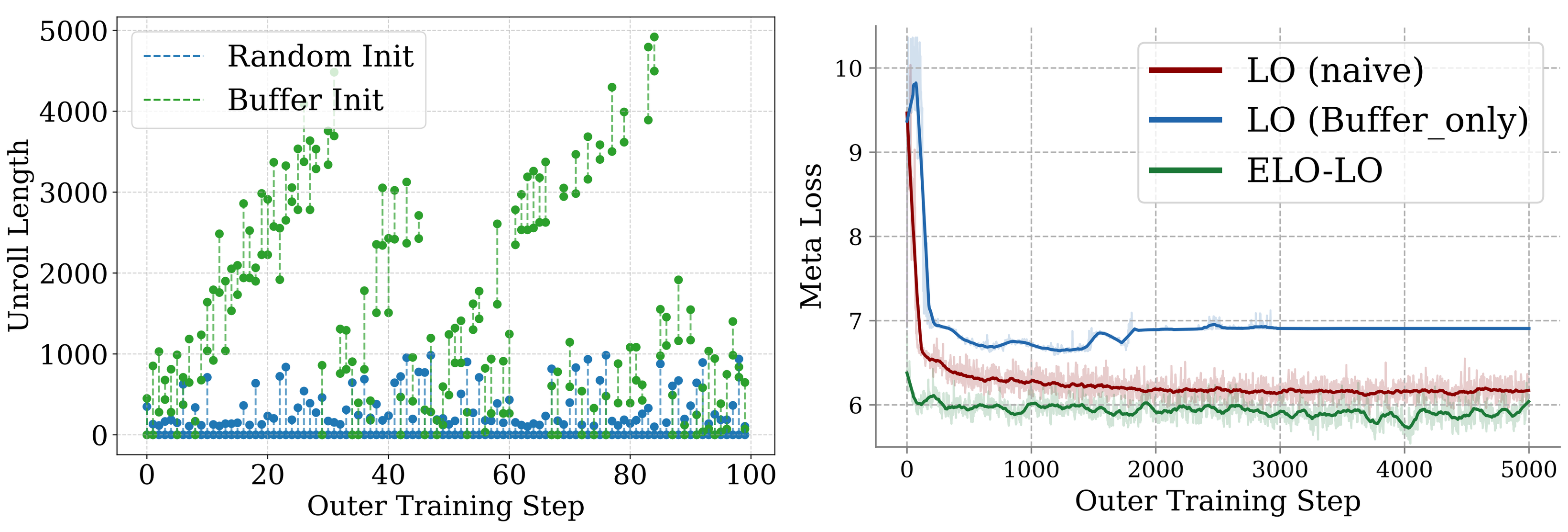

Figure 2. **Left:** Buffer initialization achieves longer effective unrolls without extra cost. **Right:** Buffer alone causes collapse; adding behavior cloning ensures stable meta-training.

Vision Tasks

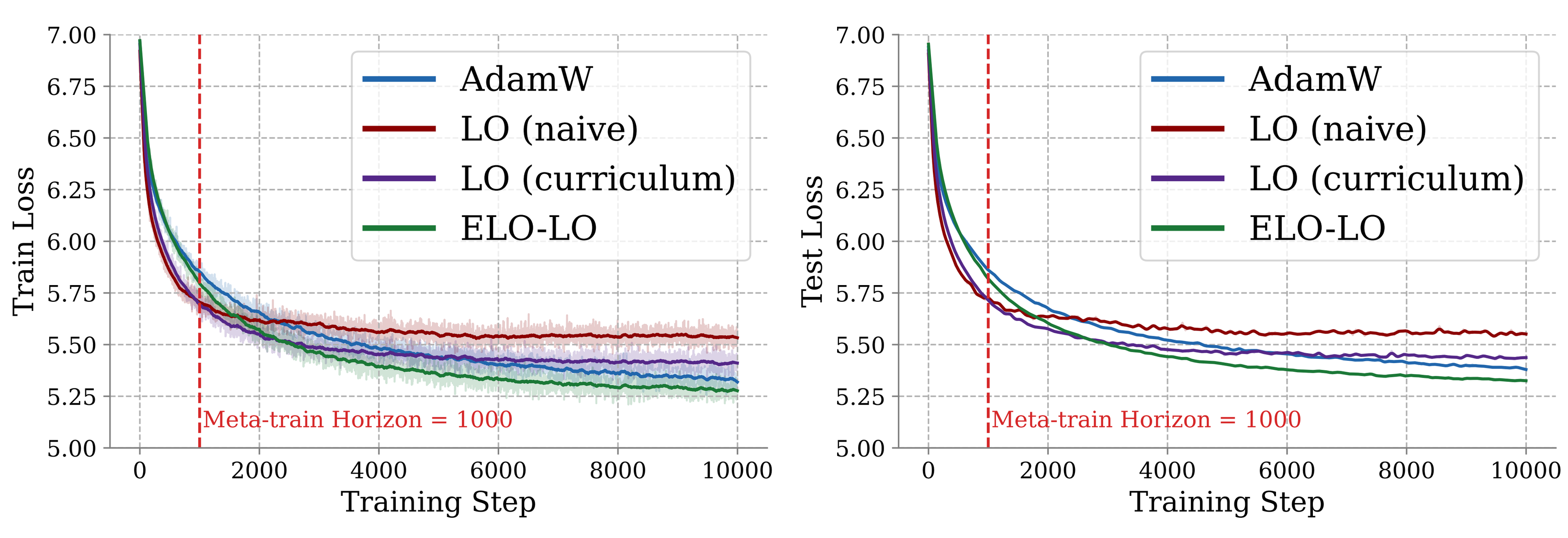

Figure 3. Evaluation on ImageNet-1K ($32 \times 32 \times 3$) using MLP-d3-w128.

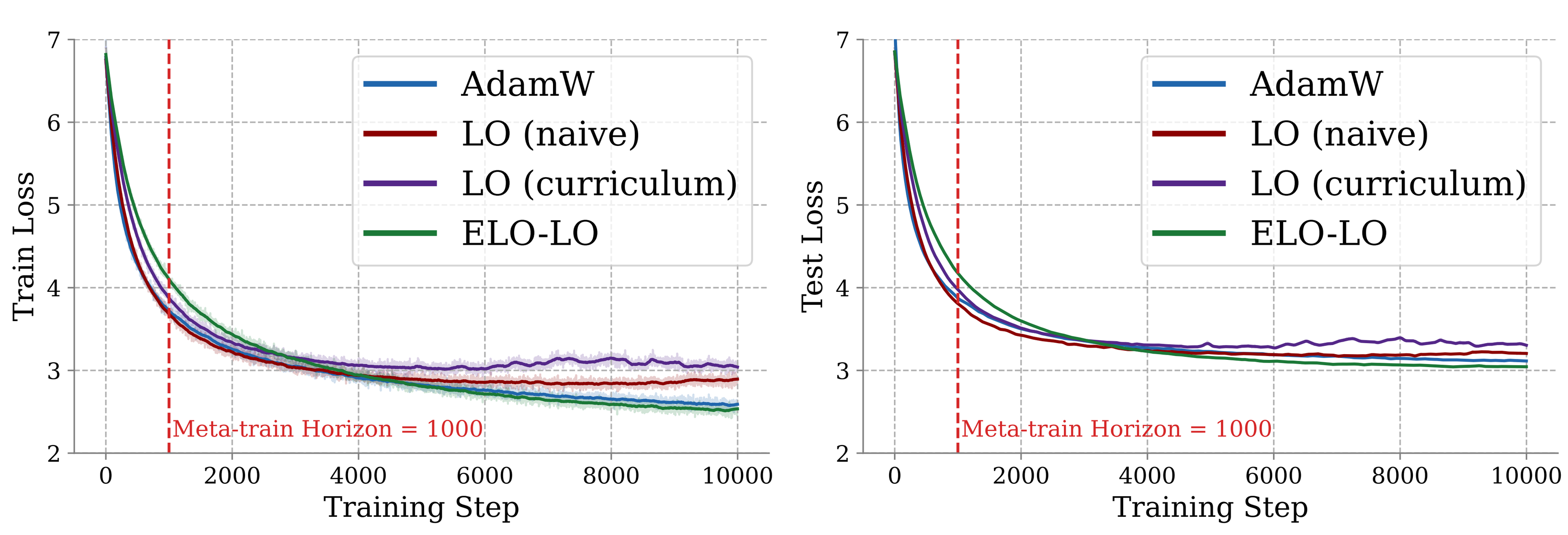

Figure 4. Evaluation on ImageNet-1K ($32 \times 32 \times 3$) using ResNet18.

Language Task

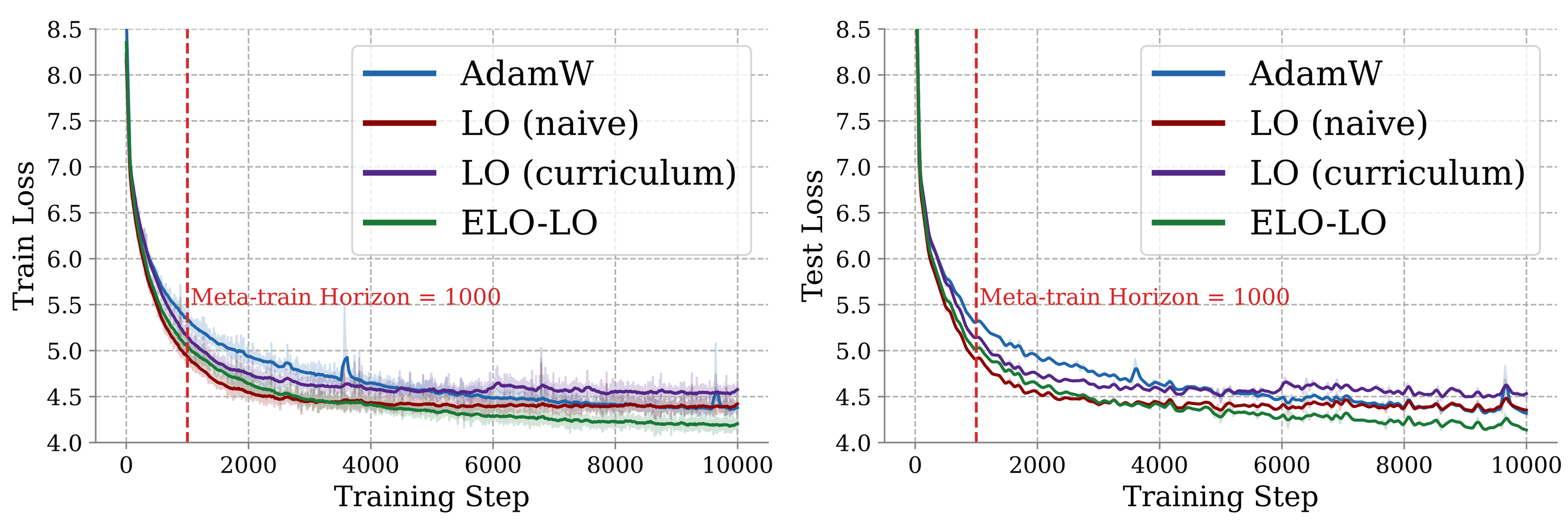

Figure 5. Evaluation on FineWeb-10B using gpt2-mini.

Results summary

Table 1. Best performance of different methods on vision and language tasks. For vision tasks (ImageNet-1K (32×32)), we report test accuracy (%). For language tasks (FineWeb-10B), we report test cross entropy loss.

Method	ImageNet-1K MLP	ImageNet-1K ResNet18	FineWeb-10B GPT2-mini
Metric	Acc. (%)	Acc. (%)	CE-Loss
AdamW	8.26	35.62	4.31
LO (naive)	6.69	33.83	4.29
LO (curriculum)	7.77	31.96	4.44
ELO-LO	**8.77**	**36.51**	**4.13**

Conclusion

We introduced **ELO**, an efficient meta-training scheme for learned optimization. ELO consistently outperforms AdamW and naive LO baselines across vision and language tasks, achieving strong generalization to $10\times$ longer unrolls.