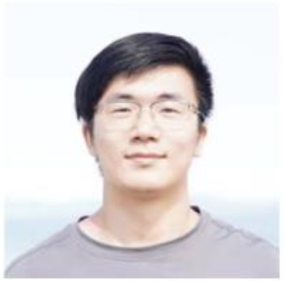





Zhizhong Li   Sina Sajadmanesh   Jingtao Li   Lingjuan Lyu

StelLA: Subspace Learning in Low-rank Adaptation using Stiefel Manifold

06.11.2025 Sony AI

NEURAL INFORMATION PROCESSING SYSTEMS
2025

What is StelLA?

LoRA

- Given a pre-trained weight matrix $W \in R^{m\times n}$, LoRA computes the adapted weight as

$$W + BA^\top$$

where $B \in \mathbb{R}^{m\times r}$ and $A \in \mathbb{R}^{n\times r}$. W is frozen during training.

StelLA

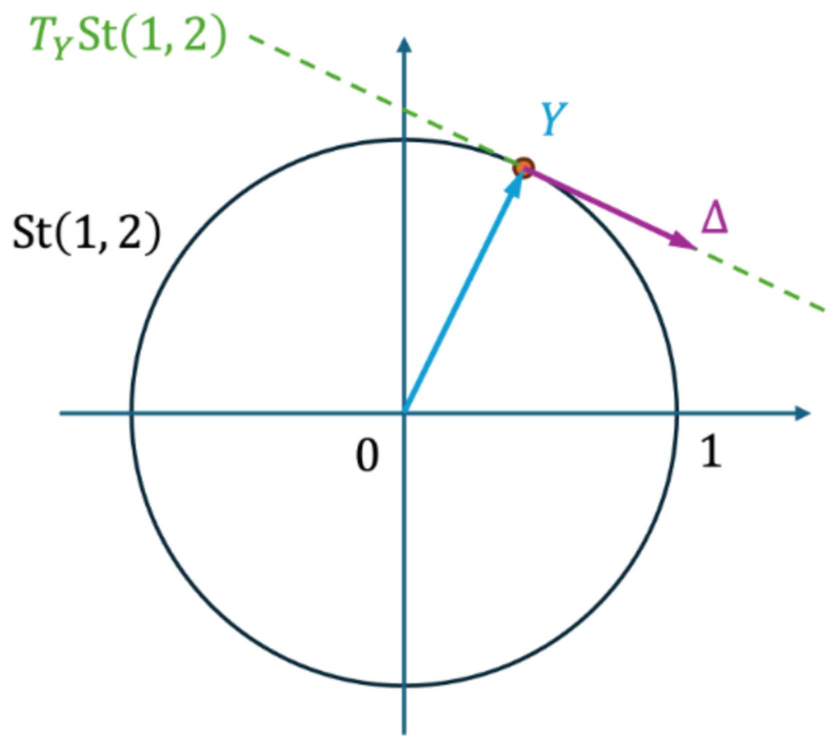
- Represent the low-rank adaptation in the SVD-style

$$W + USV^\top$$

where $U \in \mathrm{St}(r, m)$ and $V \in \mathrm{St}(r, n)$ lie in the Stiefel manifold. $S \in \mathbb{R}^{r\times r}$.

Stiefel Manifold

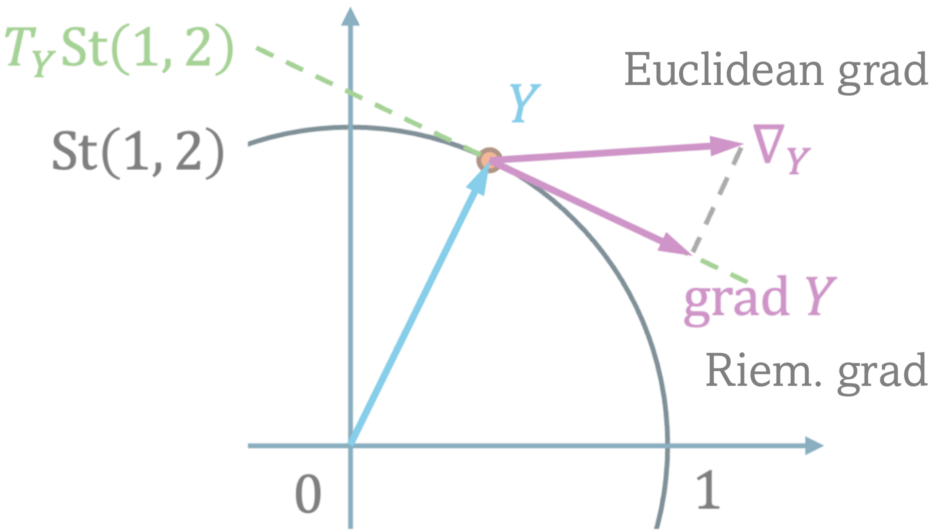
- The Stiefel manifold $\mathrm{St}(k, n)$ is the set of all $n \times k$ matrices with orthonormal columns.
- For example, $\mathrm{St}(1, 2)$ consists of all unit vectors in the 2D plane. It is the unit circle as shown in the right figure.



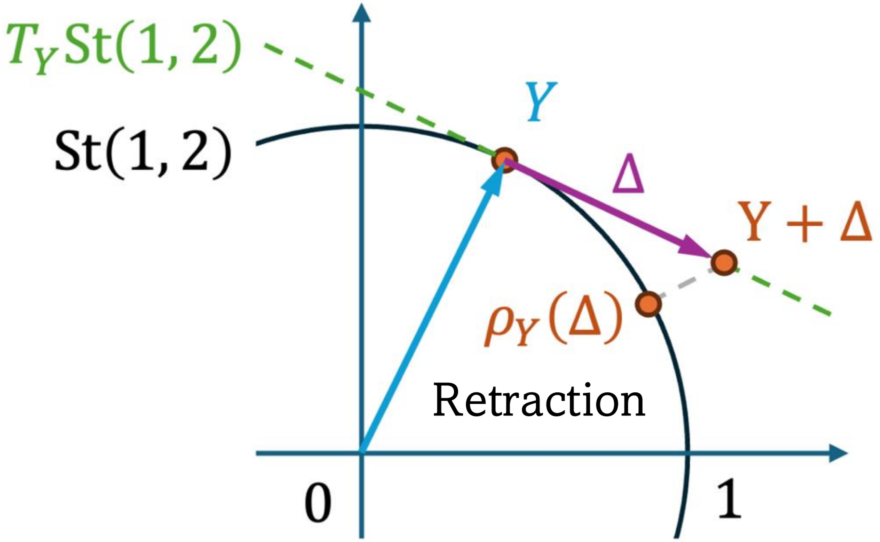
(a) The manifold $\mathrm{St}(1, 2)$ and the tangent space $T_Y\mathrm{St}(1, 2)$ at point Y.

Optimization on Stiefel Manifold

Riemannian optimization:



Convert gradient from Euclidean to Riemannian by projection



Retract $Y + \Delta$ back to Stiefel manifold by QR decomposition

StelLA Implementation

Algorithm 1 StelLA: Stiefel Low-Rank Adaptation

Require: Pre-trained weight $W \in \mathbb{R}^{m \times n}$, loss function $\mathcal{L}$, a Euclidean optimizer's step function 'step', rank r, scale factor α, number of iterations T.

1: Randomly initialize $U_0 \in \mathrm{St}(r, m)$ and $V_0 \in \mathrm{St}(r, n)$, set $S_0 \leftarrow I_r$.

2: **for** $t \leftarrow 0$ to $T - 1$ **do**

3: Compute loss: $\mathcal{L}_t \leftarrow \mathcal{L}(W + \frac{\alpha}{r} U_t S_t V_t^\top)$.

4: Compute Euclidean gradients: $\nabla_{U_t}, \nabla_{S_t}, \nabla_{V_t}$.

5: Convert Euclidean gradients to Riemannian:

$$\mathrm{grad}_{U_t} \leftarrow \nabla_{U_t} - U_t \nabla_{U_t}^\top U_t, \quad \mathrm{grad}_{V_t} \leftarrow \nabla_{V_t} - V_t \nabla_{V_t}^\top V_t.$$

6: Take tentative steps using the given optimizer's step function: ▷ *e.g.*, using Adam

$$\tilde{U}_{t+1} \leftarrow \mathrm{step}(U_t, \mathrm{grad}_{U_t}), \quad \tilde{V}_{t+1} \leftarrow \mathrm{step}(V_t, \mathrm{grad}_{V_t}), \quad S_{t+1} \leftarrow \mathrm{step}(S_t, \nabla_{S_t}).$$

7: Project the perturbed gradients $\tilde{U}_{t+1} - U_t, \tilde{V}_{t+1} - V_t$ back to the tangent space:

$$\Delta_{U_t} \leftarrow \pi_{U_t}(\tilde{U}_{t+1} - U_t), \quad \Delta_{V_t} \leftarrow \pi_{V_t}(\tilde{V}_{t+1} - V_t).$$

8: Update and retract back to the manifold: $U_{t+1} \leftarrow \rho_{U_t}(\Delta_{U_t})$, $V_{t+1} \leftarrow \rho_{V_t}(\Delta_{V_t})$.

9: **end for**

10: **return** Adapted weight: $\tilde{W} \leftarrow W + \frac{\alpha}{r} U_T S_T V_T^\top$.

Highlights

- Easy to implement
  - By using optimizer pre/post-hooks
  - Work with PyTorch optimizers
- Easy to use
  - Integrated to the peft library
- Efficient
  - Use fast svd driver (1.5x speed up)
  - Batched svd (15x speed up)
  - Only 15% slower than vanilla LoRA (train commonsense on LLaMA3-8B)

Results

Table 1: Accuracy on the commonsense reasoning benchmark. All results are averaged over 3 runs.

Model	Method	Params (%)	BoolQ	PIQA	SIQA	HellaS.	WinoG.	ARC-e	ARC-c	OBQA	Avg.
LLaMA2-7B	LoRA	0.826	72.02	83.46	<u>79.87</u>	90.44	82.69	84.83	71.19	81.53	80.76
	DoRA	0.838	<u>72.67</u>	83.48	79.82	90.82	<u>83.58</u>	85.16	<u>71.27</u>	81.20	<u>81.00</u>
	PiSSA	0.826	71.16	<u>83.89</u>	79.19	<u>91.00</u>	82.87	85.09	69.48	<u>83.93</u>	80.83
	OLoRA	0.826	71.11	82.70	78.64	89.41	81.48	83.58	68.17	80.20	79.41
	TriLoRA	0.828	71.23	80.96	78.33	80.91	77.59	81.76	66.69	79.80	77.16
	MoSLoRA	0.828	71.54	83.84	79.60	90.50	83.19	84.40	69.96	80.47	80.44
	ScaledAdamW	0.826	72.20	83.86	79.67	90.80	82.43	<u>85.55</u>	70.59	81.93	80.88
	StelLA	0.828	**73.62**	**84.87**	**80.64**	**91.44**	**84.50**	**86.43**	**72.84**	**84.33**	**82.33**
LLaMA3-8B	LoRA	0.700	75.16	88.14	80.18	95.41	<u>86.74</u>	90.84	78.70	<u>87.00</u>	85.27
	DoRA	0.710	<u>75.38</u>	88.01	79.94	95.35	86.29	90.54	79.69	86.07	85.16
	PiSSA	0.700	74.67	88.12	<u>80.50</u>	94.98	85.22	90.15	78.87	85.60	84.76
	OLoRA	0.700	74.41	87.68	79.55	94.79	85.40	90.04	78.24	85.00	84.39
	TriLoRA	0.702	73.09	86.64	78.64	93.40	82.88	87.76	75.26	84.30	82.74
	MoSLoRA	0.702	74.88	88.43	80.31	95.50	86.26	90.00	79.86	85.80	85.13
	ScaledAdamW	0.700	75.24	<u>88.57</u>	80.21	<u>95.81</u>	85.11	<u>91.09</u>	<u>80.55</u>	86.60	<u>85.40</u>
	StelLA	0.702	**75.91**	**89.86**	**81.68**	**96.41**	**87.82**	**91.98**	**82.34**	**87.80**	**86.72**

Thank you!

https://github.com/SonyResearch/stella