MambaPEFT: Exploring Parameter-EfficientFine-Tuning for Mamba

Masakazu Yoshimura*, Teruaki Hayashi*, and Yota Maeda (Sony Group Corporation) *equally contributed

The Code is available!

Introduction

Mamba is successful in various fields

However, Transformer's eco-system is huge.
Once a large Transformer model is created, it can be efficiently adapted to various tasks with parameter-efficient fine-tuning (PEFT).

PEFT for Mamba is needed to replace Transformer's eco-system.

Our contribution

- Check the availability of existing PEFT methods to Mamba
- Optimize them to Mamba with extensive exploration
- Propose PEFT methods specific to Mamba

Partial tuning - tuning small parameters

A, D, conv1d, cls_embed, bias, …

Additive method - add parameters

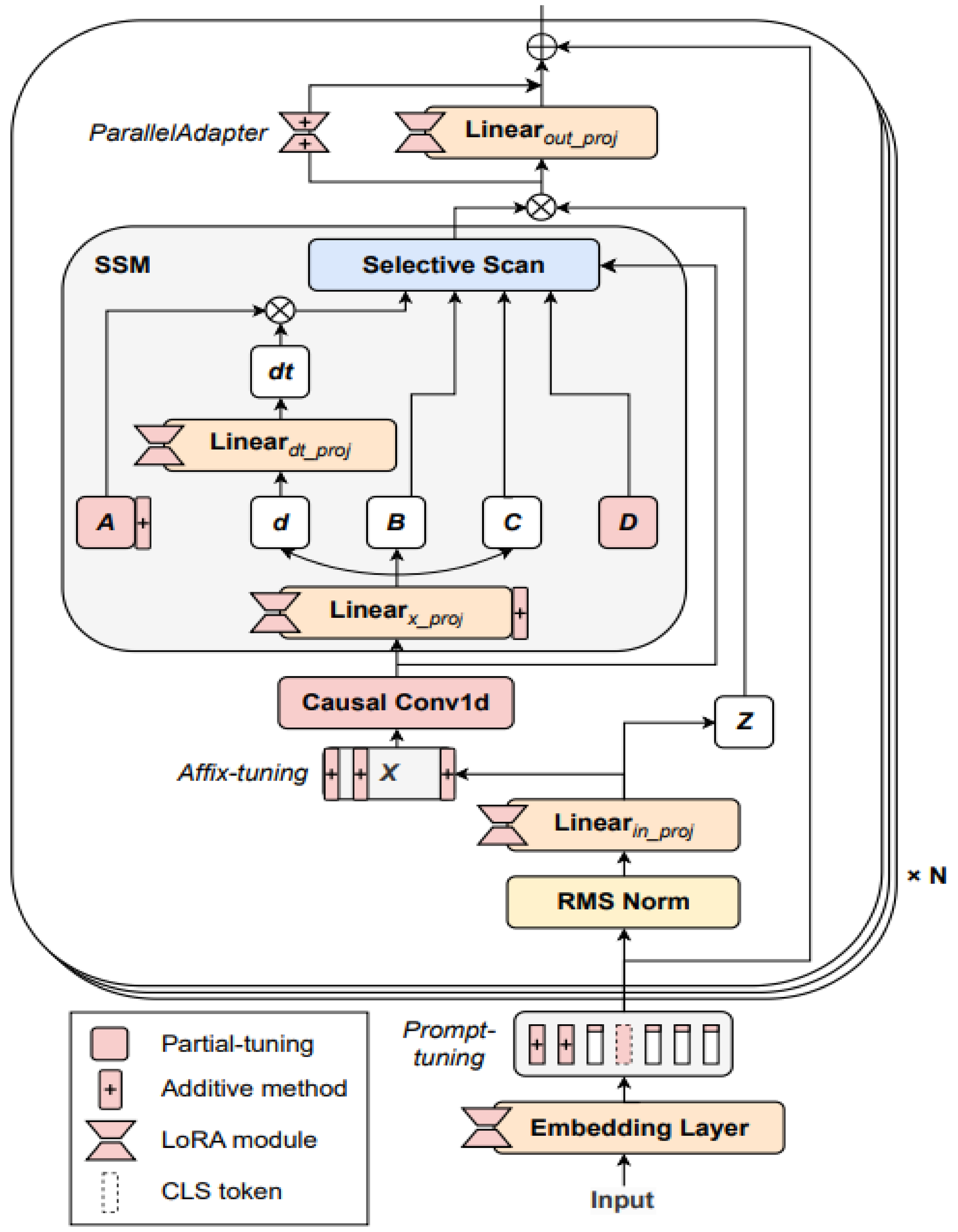
affix-tuning, **additional-scan**, …

Reparameterization method

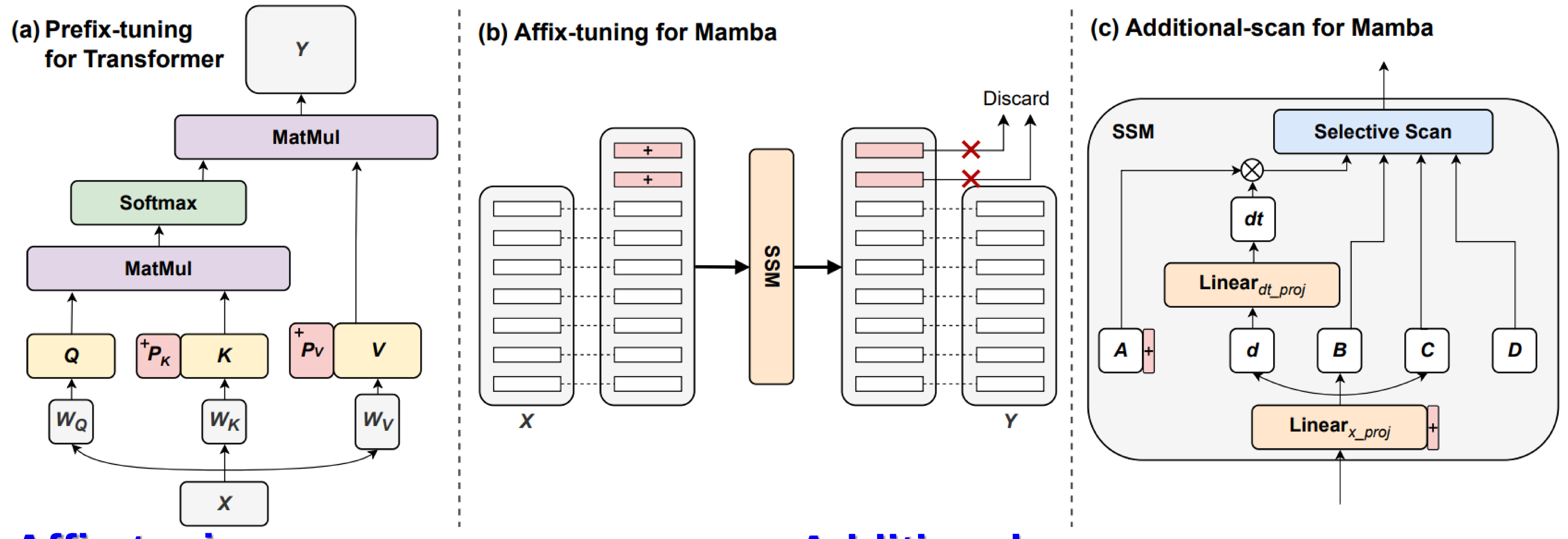
LoRA, **partial-LoRA**

Overview

Investigate, improve, and propose 20 variations of 7 PEFT methods



Proposed PEFT methods specific to Mamba



Affix-tuning

Add tokens before causal conv1d and discard after selective scan

Additional-scan

add learnable dimensions to the hidden state in SSM

Evaluation & Findings

Vtab-1K (vision, 1K training data)

Model	Method	#Params (K)	Avg.
ViT-S	Scratch	21,704	26.20
	Full	21,704	53.47
	Linear Probing	9	51.74
	FacT-TK	16	66.96
	LoRA	628	68.68
	Adaptformer	333	68.97
	SPT-LoRA	414	69.38
	Adapter+	122	69.87
Vim-S	Scratch	25,450	25.42
	Full	25,450	47.08
	Linear Probing	9	52.75
	Conv1d-tuning	156	69.09
	Prompt-tuning (w/o proj)	12	56.77
	Prompt-tuning	307	62.54
	Affix-tuning (w/o proj)	230	65.04
	Affix-tuning	117,000	70.29
	Additional-scan	672	68.65
	ParallelAdapter	663	70.96
	LoRA(out_proj)	2,663	71.12
	LoRA(in_proj)	1,483	71.25
	LoRA$_p$(X)	1,778	71.52
	Hybrid (w/ proj)	117,236	**72.05**
	Hybrid (w/o proj)	1,044	71.80

Commonsense (language, 170K data)

Model	Method	#Params(%)	Avg.
Pythia 160M	Full	100	42.0
	LoRA	0.72	41.6
Mamba 130M	Full	100	43.8
	SLL LoRA	1.45	42.7
	Additional-scan	0.51	42.7
	Affix-tuning (w/o proj)	0.17	40.6
	Affix-tuning	64.64	43.2
	LoRA(in_proj)	2.23	42.8
	LoRA$_p$(X)	2.67	**43.7**
Pythia 1.4B	Full	100	50.5
	LoRA	0.44	50.5
Mamba 1.4B	Full	100	53.0
	SLL LoRA	4.64	52.7
	Additional-scan	0.26	53.5
	Affix-tuning (w/o proj)	0.09	**53.9**
	LoRA(in_proj)	1.13	52.6
	LoRA$_p$(X)	1.36	53.7

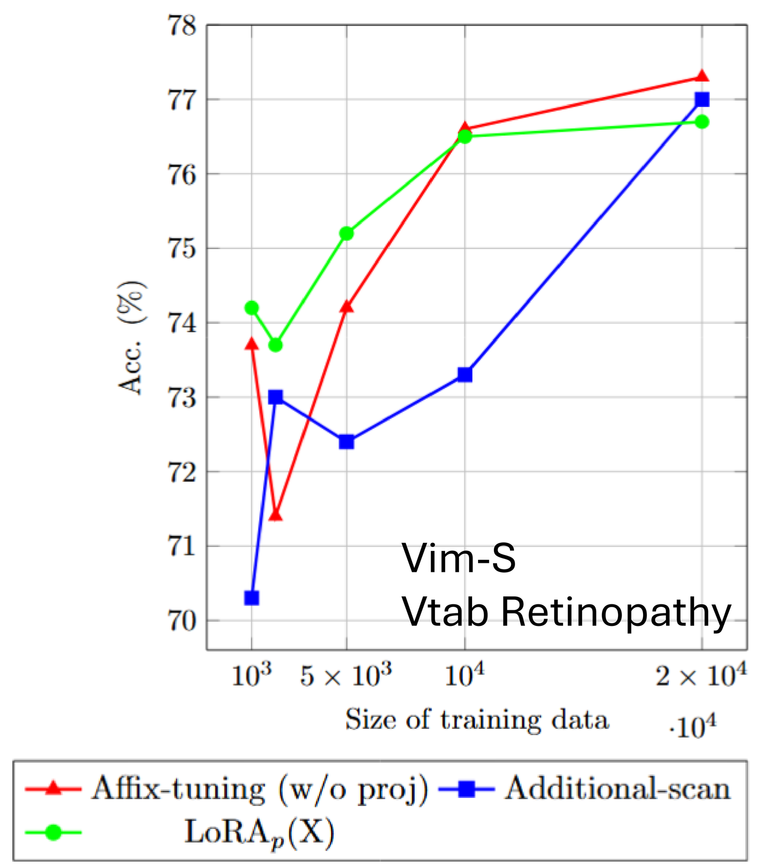
(1) Mamba benefits from PEFT more than Transformers

larger improvement from full fine-tuning

(2) LoRA$_p$(X) is effective with limited data

apply LoRA on the partial weight of in_proj layer to generate feature X

(3) Additional-scan is effective with large data



(4) Affix-tuning is effective for large Mamba models

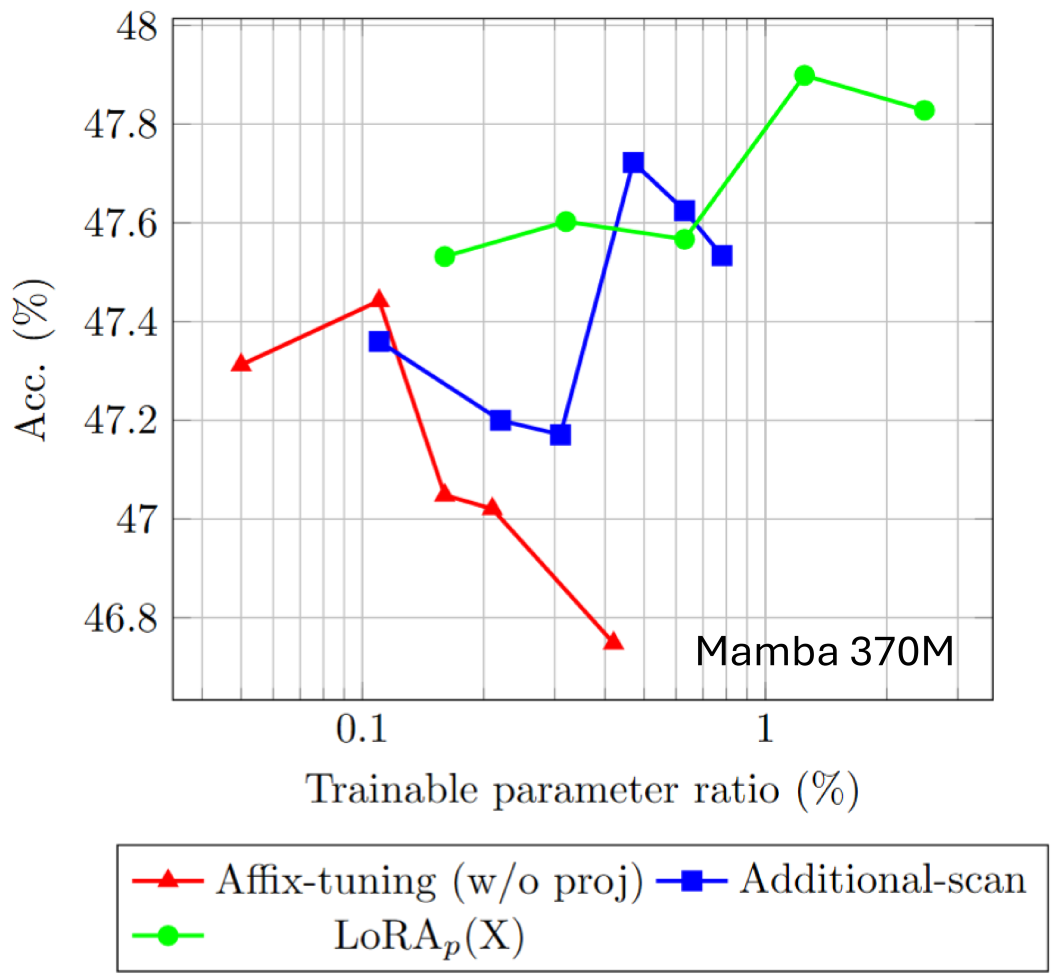
(5) LoRA$_p$(X) is enough, and adding LoRA to other layers is harmful

different from Transformers

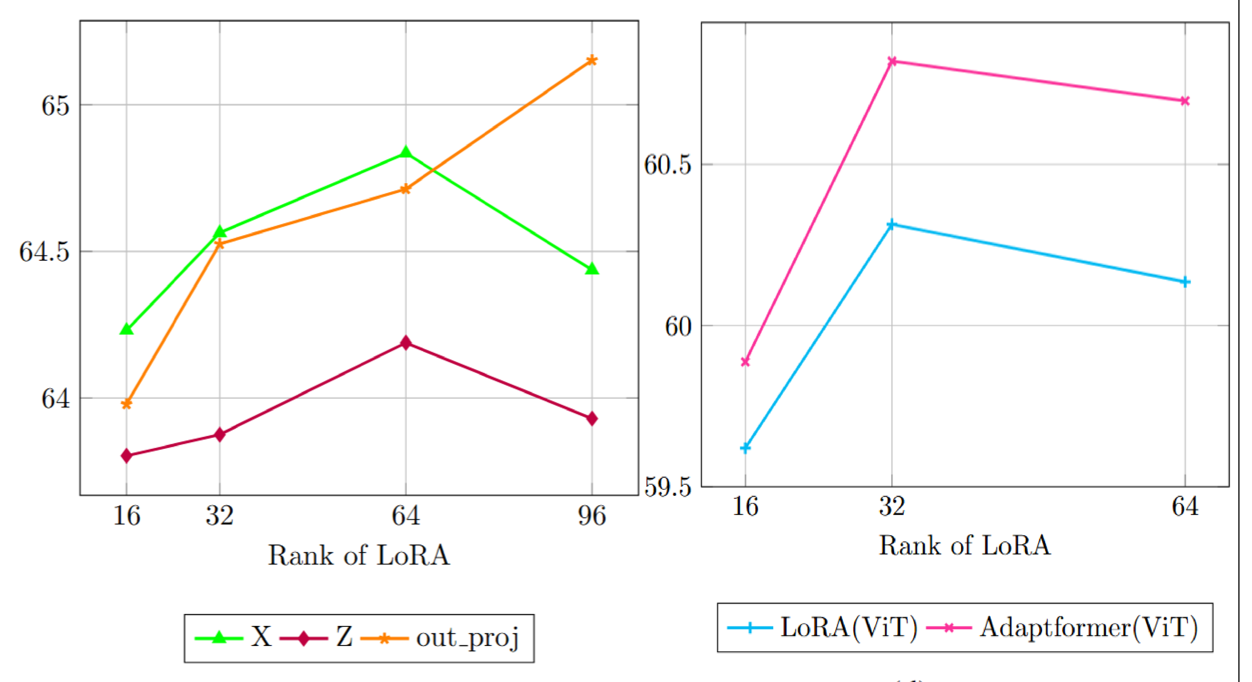
Method	Rank	Avg.
LoRA(in_proj + out_proj)	8	71.33
LoRA(in_proj + out_proj)	16	70.68
LoRA(in_proj + out_proj)	32	69.77
LoRA(ALL)	8	71.02
LoRA(ALL)	16	70.42
LoRA(ALL)	32	68.74
LoRA$_p$(X)	64	**71.52**

Vim-S VTab Avg.

(6) We should choose a suitable PEFT method depending on the computational budget



(7) PEFT for Mamba can be improved by adding more parameters



(8) In Hybrid PEFT, simply combining individually high-performance methods, as many previous works did, is inefficient