Evolving Parameterized Prompt Memory for Continual Learning

Muhammad Rifki Kurniawan[1], Xiang Song[1], Zhiheng Ma[3], Yuhang He[2], Yihong Gong[1,2], Qi Yang[4], Xing Wei[1]

[1]School of Software Engineering, Xi'an Jiaotong University, [2]College of Artificial Intelligence, Xi'an Jiaotong University
[3]Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, [4]School of Computer Science and Technology, Xi'an Jiaotong University

Prompting Foundational Model in Continual Learning (CL)

- **Continual learning** objective involves **finding generalized parameters** through sequential task learning, integrating new concepts in each task while minimizing **catastrophic forgetting**.
- **Learning limited parameters** from foundational Vision Transformer (ViT), instead from *tabula rasa*, becoming new trend in CL, specifically those utilized prompting, selected discretely based on instance query.
- **Pool of key-prompt pair**, establishing new one each task/concept, emerge as prevalent framework in CL from foundational model.

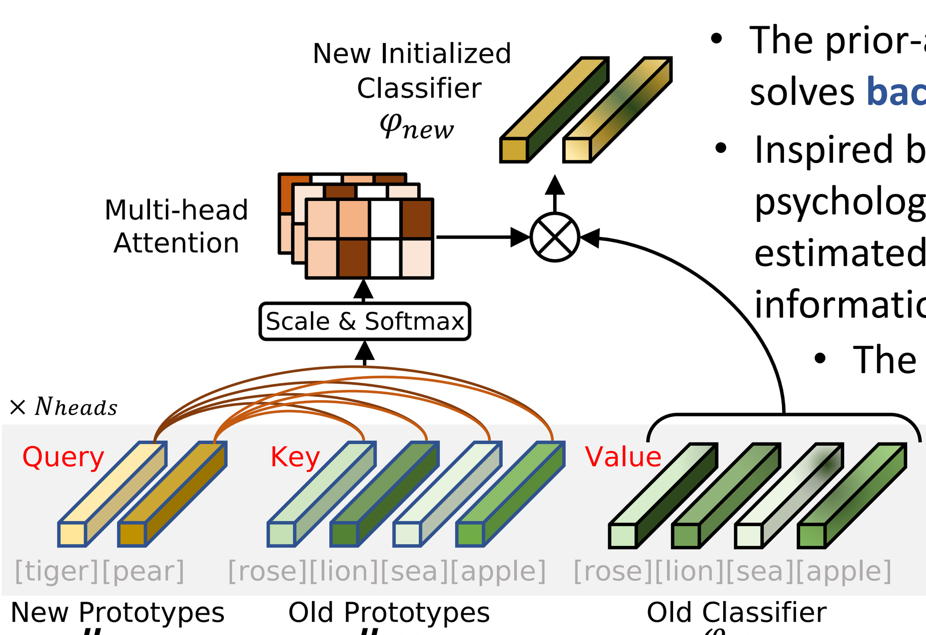
Issues with Previous Prompt-based CL

- **Discrete prompting**: test-time prediction may experience **prompt selection mismatches** and incorrect prompt associations due to a discrete bottleneck in key-prompt within prompt pool.
- **Lack shareability**: orthogonal categories that belong to the same task use an **identical prompt if coming from task**, prevent sharing prompts nearly similar categories.
- **Dynamically expand**: introducing new prompt at each task, thus the **pool dynamically expand**.
- **Backward incompatible**: rely on trick that learning to contrasting intra-task categories only, overlooking discrimination across inter-task classes.

Proposed Ideas

- **Prompt parameterization**: formulating prompting as feed-forward networks (FFNs) with multilayer perceptron (MLP) bottleneck with **soft assignment**.
- **Prompt incremental fusion**: linearly weighted fusion between optimal transport-based (OT) aligned working prompt memory (WPM) with generalized reference prompt memory (RPM).
- **Compositional classifier initialization**: inferring the future classifiers from available old classifiers and prior prototypical relations between classes.

Compositional Classifier Initialization



- The prior-aware classifier initialization solves **backward compatibility issue**.
- Inspired by **bias-adjustment heuristic** in psychology, where the unknown is estimated by leveraging relevant available information
  - The unknown is a future classifiers and the knowns are learned classifiers and old-new tasks prototypical relations from pretrained as **foundational relational presumption**.

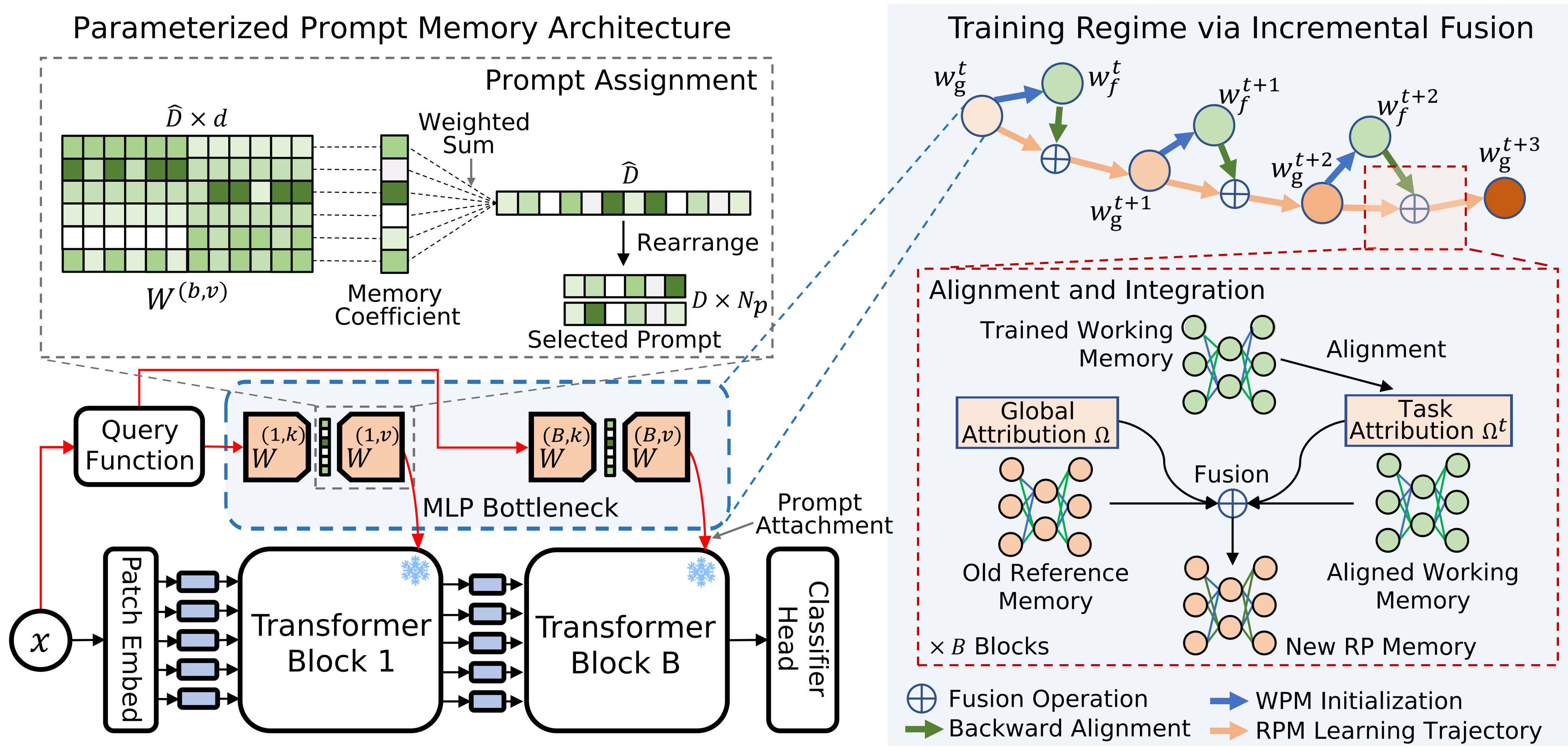
Prototypical Attention (PA):

$$\varphi_{new} = \mathrm{PA} = \mathrm{Softmax}\left(\frac{d\left(\boldsymbol{\mu}_{new}, \boldsymbol{\mu}_{old}\right)}{\tau}\right)\boldsymbol{\varphi}_{old}$$

Multi-head Prototypical Attention: (MPA):

$$\begin{aligned}\varphi_{new} &= \mathrm{MPA}\left(\boldsymbol{\mu}_{new}, \boldsymbol{\mu}_{old}, \boldsymbol{\varphi}_{old}\right)\\ &= \mathrm{Concat}\left(\mathrm{PA}_1\left(\boldsymbol{\mu}_{new,1}, \boldsymbol{\mu}_{old,1}, \boldsymbol{\varphi}_{old,1}\right),\right.\\ &\qquad \left.\ldots, \mathrm{PA}_k\left(\boldsymbol{\mu}_{new,k}, \boldsymbol{\mu}_{old,k}, \boldsymbol{\varphi}_{old,k}\right)\right).\end{aligned}$$

Evolving Parameterized Prompt Memory



Continuous Prompting

Non-negative unnormalized coefficient for assigning corresponding prompt in $W^{(b,v)}$:

$$\begin{aligned}\boldsymbol{p}_b &= f_{\mathbf{W}^{(b)}}\left(q\left(x\right);\mathbf{W}^{(b)}\right),\\ &= \mathrm{ReLU}\left(q\left(x\right)\cdot\mathbf{W}^{(b,k)}\right)\cdot\mathbf{W}^{(b,v)}\end{aligned}$$

Functional Alignment

Resolving OT problem to align functionally similar memory segment of WPM to RPM:

$$\hat{\mathbf{W}}^t_{f,\ell} \leftarrow \mathrm{diag}\left(\frac{1}{\beta_\ell}\right)\mathbf{P}_\ell\mathbf{W}^t_{f,\ell}\mathbf{P}_{\ell-1}\mathrm{diag}\left(\frac{1}{\beta_{\ell-1}}\right)$$

Incremental Fusion with Awareness to Attribution

Prompt memory evolution via momentum merging, considering importance λ of each node while merging memory, determining the fusion weight of node parameter vectors:

$$\mathbf{W}^{t+1}_{g,\ell} \leftarrow \lambda_\ell\hat{\mathbf{W}}^t_{f,\ell} + (1-\lambda_\ell)\mathbf{W}^t_{g,\ell}$$

Empirical Results From Experiments

Class Incremental Learning: Evaluation results Accuracy and forgetting on Split ImageNet-R.

Method	Split ImageNet-R: 5 Steps Acc.(↑)	5 Steps Forget.(↓)	10 Steps Acc.(↑)	10 Steps Forget.(↓)	20 Steps Acc.(↑)	20 Steps Forget.(↓)	Avg Acc.(↑)	Avg Forget.(↓)
FT-seq	61.41 ±0.38	5.76 ±0.48	50.28 ±2.29	24.28 ±1.73	39.25 ±0.90	40.38 ±0.77	50.31 (+0.00)	23.48 (−0.00)
LP-seq	59.83 ±0.33	**1.50 ±0.41**	55.30 ±0.12	7.85 ±0.10	51.97 ±0.34	13.87 ±0.21	53.64 (+3.33)	7.74 (−15.74)
NME-seq	61.06	6.64	61.40	**0.76**	61.76	2.89	61.41 (+11.10)	**3.43 (−20.05)**
L2P	66.63 ±0.33	6.65 ±0.38	64.05 ±0.39	10.05 ±0.26	60.34 ±0.17	14.44 ±0.61	63.67 (+13.36)	10.38 (−13.10)
DualPrompt	71.06 ±0.35	4.19 ±0.25	69.71 ±0.25	5.44 ±0.12	66.26 ±0.46	8.74 ±0.33	69.01 (+18.70)	6.12 (−17.36)
ESN	73.42 ±0.40	3.79 ±0.55	71.07 ±0.29	4.99 ±0.49	64.77 ±0.71	6.65 ±1.24	69.75 (+19.44)	5.14 (−18.34)
CODA-P-S	73.80 ±0.40	5.56 ±0.64	71.95 ±0.41	5.92 ±0.35	69.67 ±0.35	6.23 ±0.40	71.81 (+21.50)	5.90 (−17.58)
CODA-P	73.77 ±0.48	6.60 ±0.52	72.42 ±0.40	6.26 ±0.61	70.18 ±0.43	5.53 ±0.21	72.12 (+21.81)	6.13 (−17.35)
EvoPrompt-S	76.79 ±0.23	9.84 ±0.15	76.22 ±0.16	2.33 ±0.24	**74.68 ±0.51**	2.70 ±0.19	75.90 (+25.59)	4.96 (−18.52)
EvoPrompt	**77.16 ±0.18**	9.89 ±0.30	**76.83 ±0.08**	2.78 ±0.06	74.41 ±0.23	**2.56 ±0.22**	**76.13 (+25.82)**	5.08 (−18.40)
Upper-bound†	79.13 ±0.18	-	79.13 ±0.18	-	79.13 ±0.18	-	79.13	-

Online Learning: Results on single epoch on 10 tasks CIL, measuring effectiveness on acquisition.

Method	Split CIFAR-100 Acc.(↑)	Split CIFAR-100 Forget.(↓)	Split ImageNet-R Acc.(↑)	Split ImageNet-R Forget.(↓)
FT-seq	35.39±1.00	32.98±1.53	7.51±3.73	11.22±0.71
L2P	80.49±0.28	8.74±0.44	57.52±0.18	6.54±0.34
DualPrompt	82.17±0.34	7.52±0.21	61.09 ±0.18	4.40 ±0.62
ESN	74.17 ±1.14	10.59 ±1.39	-	-
CODA-P-S	79.46 ±0.06	11.92 ±1.27	64.60 ±0.99	6.09 ±1.18
CODA-P	81.07 ±0.38	10.10 ±0.84	66.47 ±0.33	5.42 ±0.87
EvoPrompt-S	84.23 ±0.57	1.64 ±0.29	73.56 ±0.42	3.82 ±0.24
EvoPrompt	**84.72 ±0.94**	**0.89 ±0.72**	**74.05 ±0.48**	**3.66 ±0.36**

Domain Incremental Learning: Results on CORe50 dataset, measuring generalization to unseen domains.

Method	Test Acc. (↑)	Δ Acc. (↑)
NME-seq	78.20	+00.00
EWC †	74.82 ±0.60	−3.38
LwF †	75.45 ±0.40	−2.75
L2P†	78.33 ±0.06	+0.13
S-iPrompts‡	83.13 ±0.51	+4.93
S-liPrompts‡	89.06 ±0.86	+10.86
ESN‡	91.80 ±0.31	+13.60
EvoPrompt-S	94.77 ±0.50	+16.57
EvoPrompt	**95.27 ±0.15**	**+17.07**
Upper-bound	91.32 ±0.23	-

Not Quite Clear? See If This Helps

Code | Blog | Paper | Video

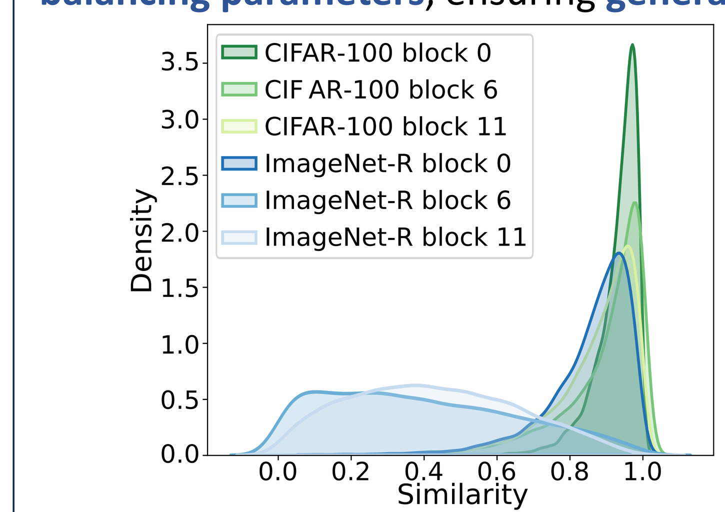
Architectural Comparison from Previous Methods

Our approach uses minimal parameters, **5×and 13× smaller** than CODA-P, without dynamically expanding parameters for new tasks like L2P, DualPrompt, or ESN.

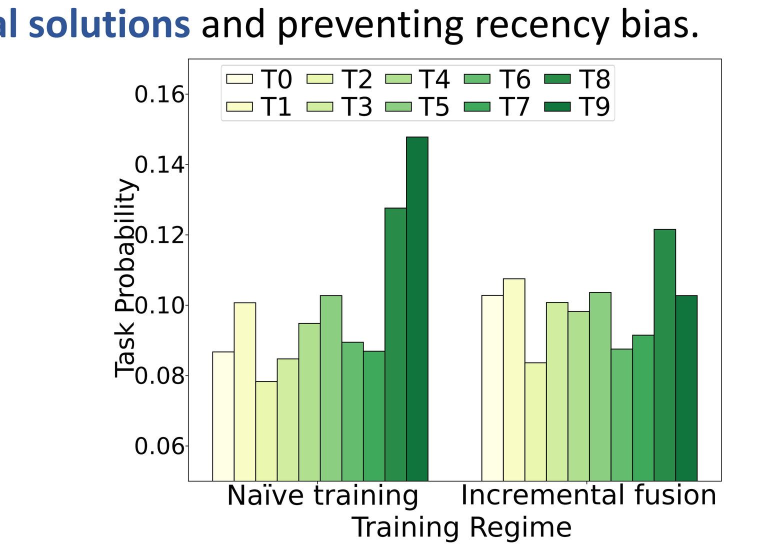
Method	Prompt selection	Dynamically expand	Acc.(↑)	Additional Params M	Additional Params %
L2P	discrete	✔	64.05	0.89	1.21%
DualPrompt	discrete	✔	69.71	0.95	1.28%
ESN	-	✔	71.07	3.67	3.52%
CODA-P-S	continuous	✔	71.95	0.92	1.25%
CODA-P	continuous	✔	72.42	3.84	4.65%
EvoPrompt-S	continuous	✘	76.22	**0.29**	**0.69%**
EvoPrompt	continuous	✘	**76.83**	0.74	1.21%

Analysis on Prompt Diversity and Recency Bias

The adaptive prompt coefficient **varies with the dataset**, being more diverse for datasets like Split ImageNet-R than Split CIFAR-100. Incremental fusion is crucial for **balancing parameters**, ensuring **general solutions** and preventing recency bias.



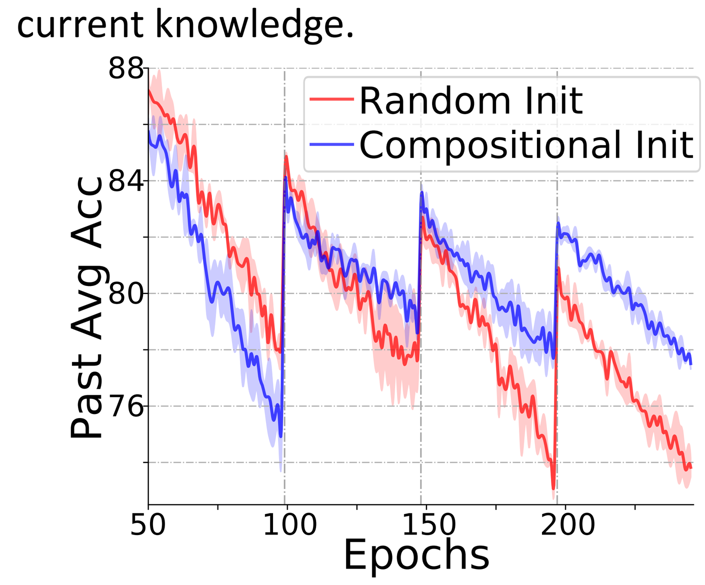
Prompt assignment diversity



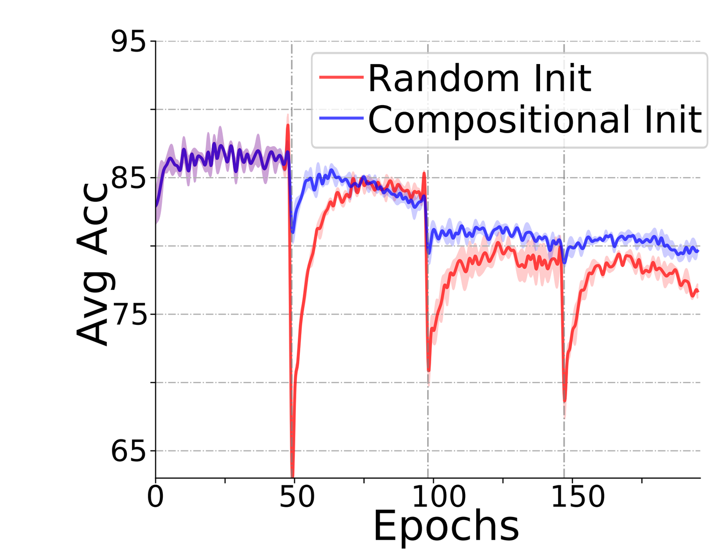
Prediction probability

Does Ours Suffer from Stability Gap?

Both random and CCI show **no signs of a stability gap**. Nevertheless, CCI showcases stability in performance, **smooth task transitions**, and **accelerated** acquisition of current knowledge.



Past average accuracy



Global average accuracy

Embedding Separability and Backward Compatibility

Our initialization reduces intra-class point distances, signifies **greater intra-class compactness**, and balances inter-class margins, leading to smaller inter-class distances than random initialization, **better backward-compatibility**.