Extending Conditional Convolution Structures for Enhancing Multitasking Continual Learning

Cheng-Hao Tu, Cheng-En Wu and Chu-Song Chen

Institute of Information Science, Academia Sinica, Taipei, Taiwan

Research Center for Information Technology Innovation, Academia Sinica, Taipei, Taiwan

MOST Joint Research Center for AI Technology and All Vista Healthcare, Taipei, Taiwan

Outline

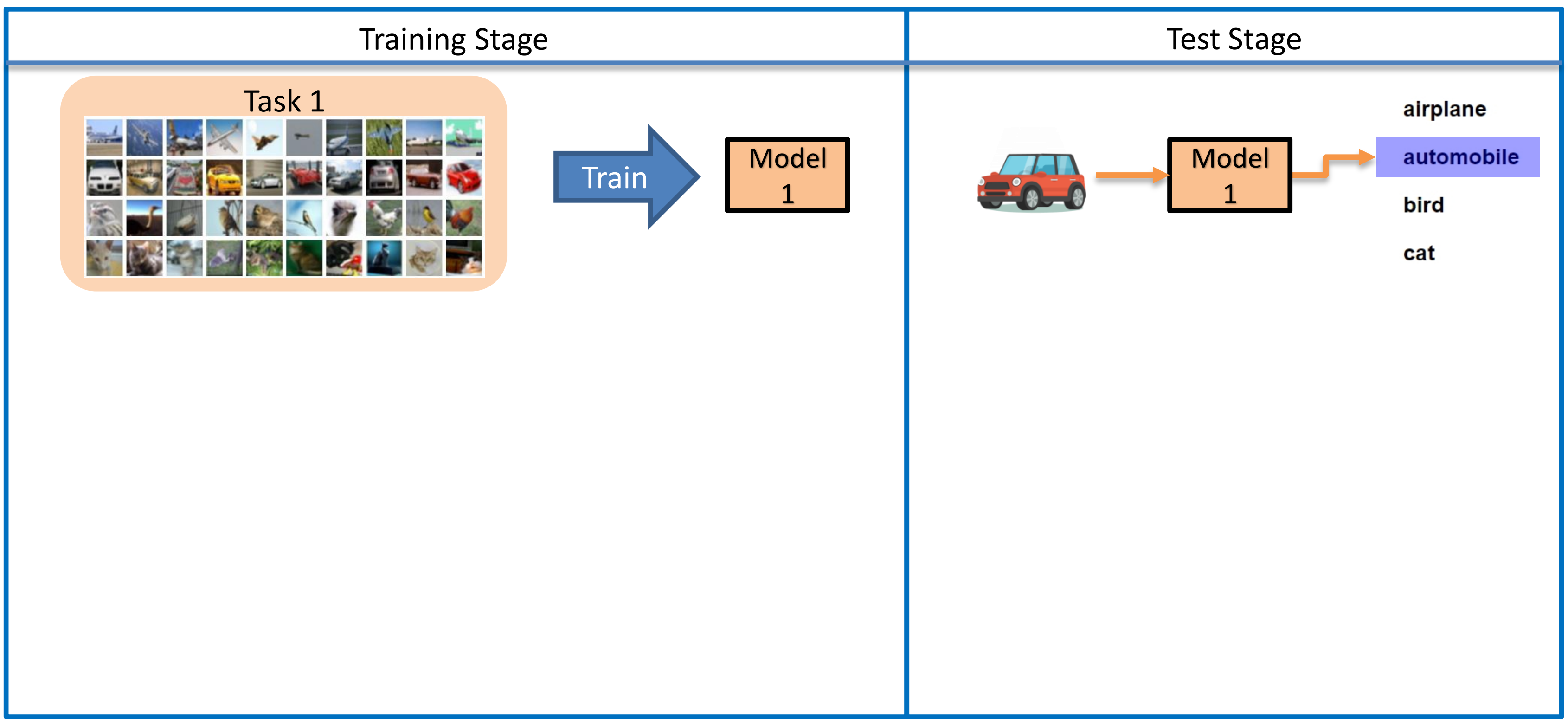
- Introduction
- Related Work
- Conditional Convolution (CondConv)
- CondConv Continual Learning
- Experiments
- Conclusion

Introduction

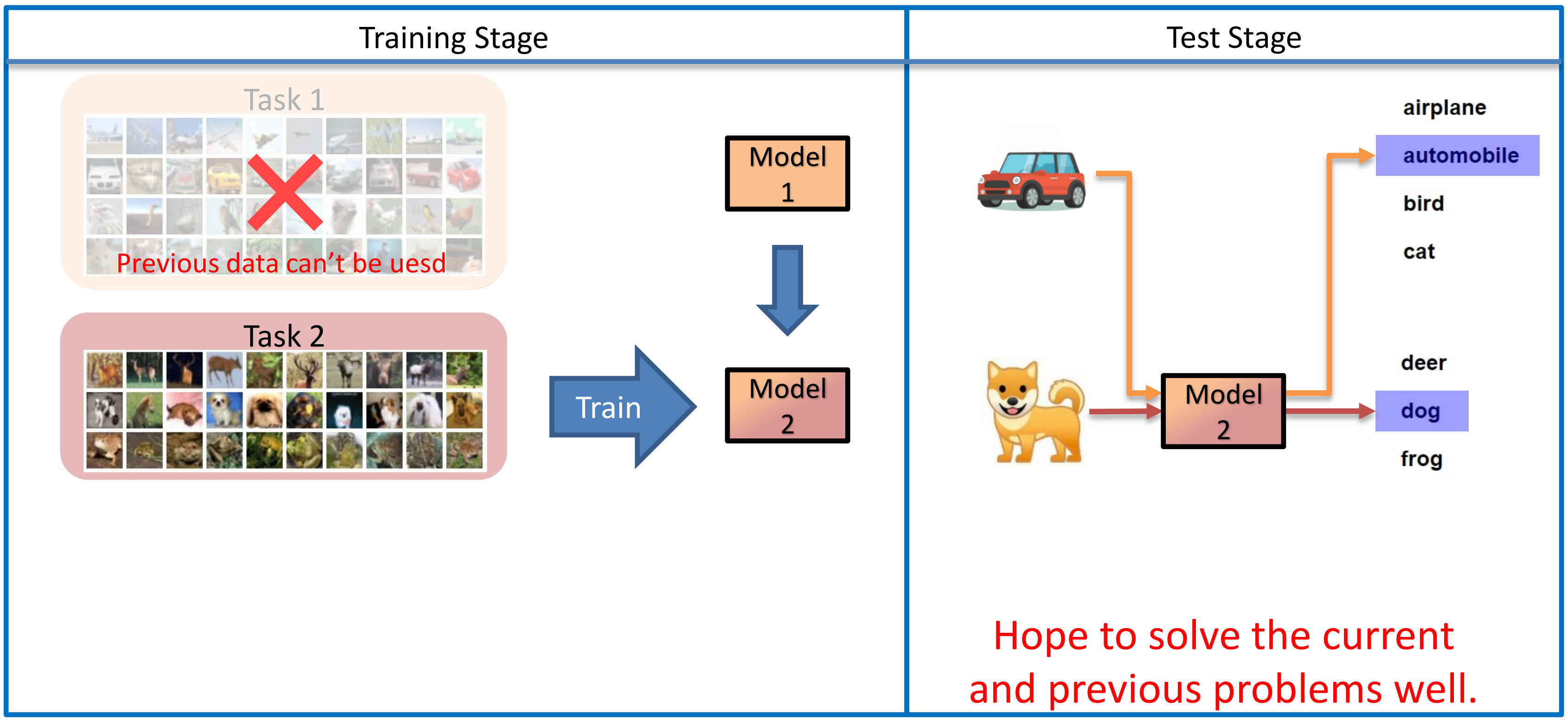
- Continual Learning aims to continuously learn an unknown sequence of tasks while keeping the performance of previously learned ones.

- The training data of previous learned tasks are assumed to be unavailable for new tasks.

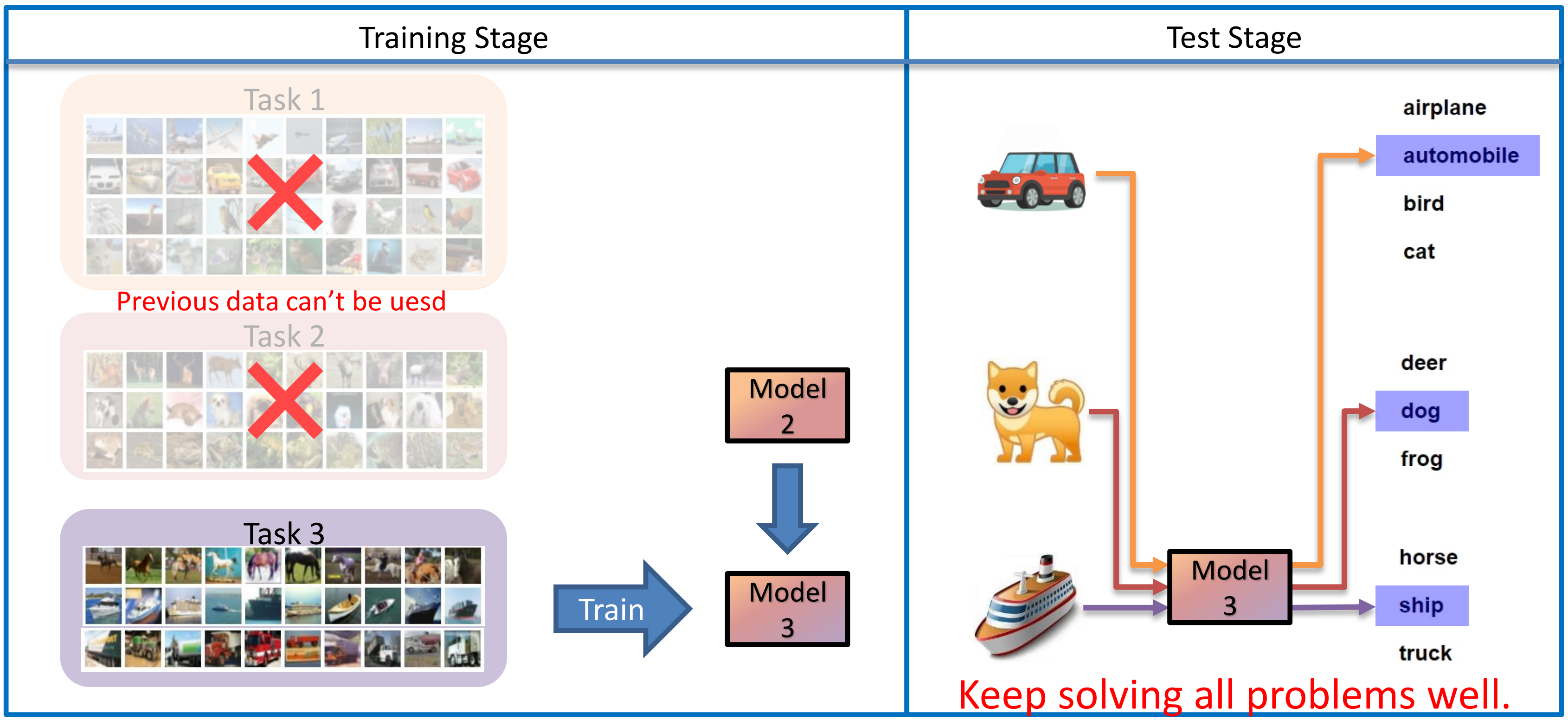
Continual Learning Illustration



Continual Learning Illustration



Continual Learning Illustration



Related Work

- While network expansion is needed to learn multiple tasks, it usually accompanies with increasing inference time.
- **Progressive** [1] progressively expands the network widths to acquire enough capacity for new tasks.
- **CPG** [2] uses iterative expansion and pruning processes to find structures with balance between model accuracy and speed.

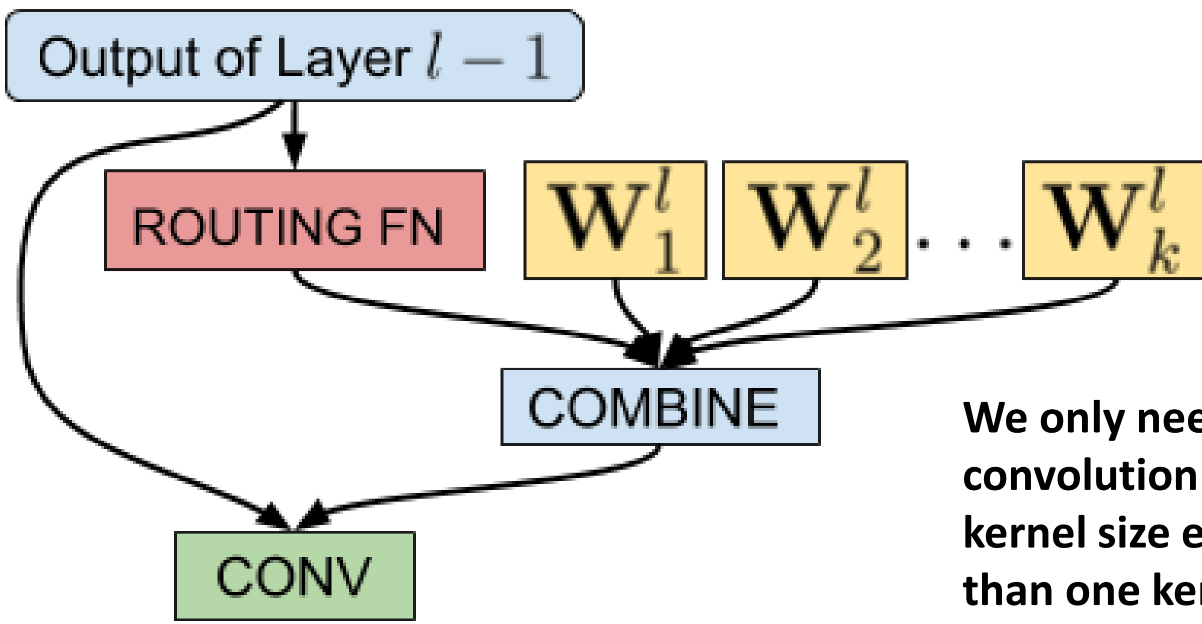
[1] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick,K. Kavukcuoglu, R. Pascanu, and R. Hadsell, “Progressive neuralnetworks,”arXiv, 2016.
[2] S. C. Y. Hung, C.-H. Tu, C.-E. Wu, C.-H. Chen, Y.-M. Chan, and C.-S. Chen, “Compacting, picking and growing for unforgetting continual learning,” in Proceedings of Advances in Neural Information Processing Systems, 2019

Related Work

- These methods adopt inefficient expansion structures (**network channels**) so they usually require network compression to make trade-off between accuracy and inference speed.
- In this paper, we use a more efficient Conditional Convolution (**CondConv**) structure for network expansion to gain the enough model capacity without losing too much efficiency.

Conditional Convolution (CondConv)

- CondConv [3] uses input-dependent routing weights to combine multiple convolutional kernels into a single one.

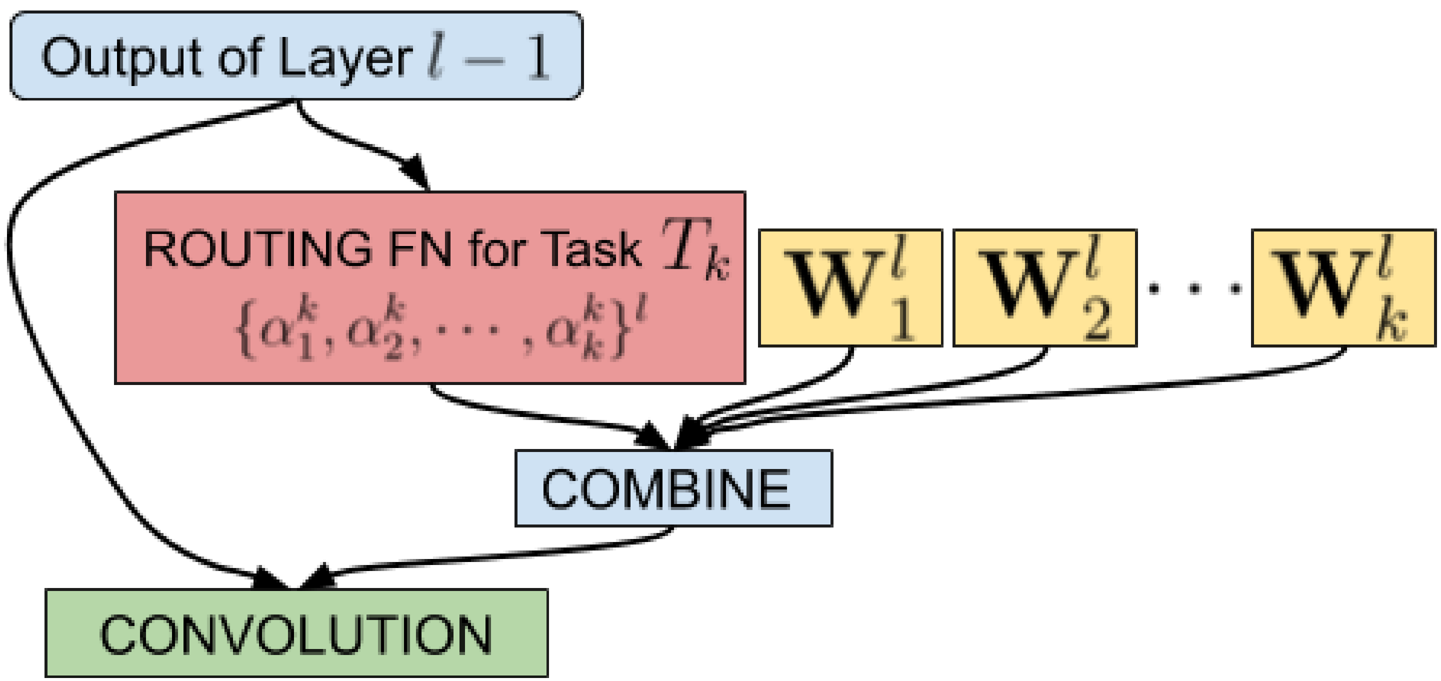


We only need to compute the convolution once with the same kernel size even though more than one kernels.

[3] B. Yang, G. Bender, Q. V. Le, and J. Ngiam, "Condconv: Conditionally parameterized convolutions for efficient inference," in Proceedings of Advances in Neural Information Processing Systems, 2019.

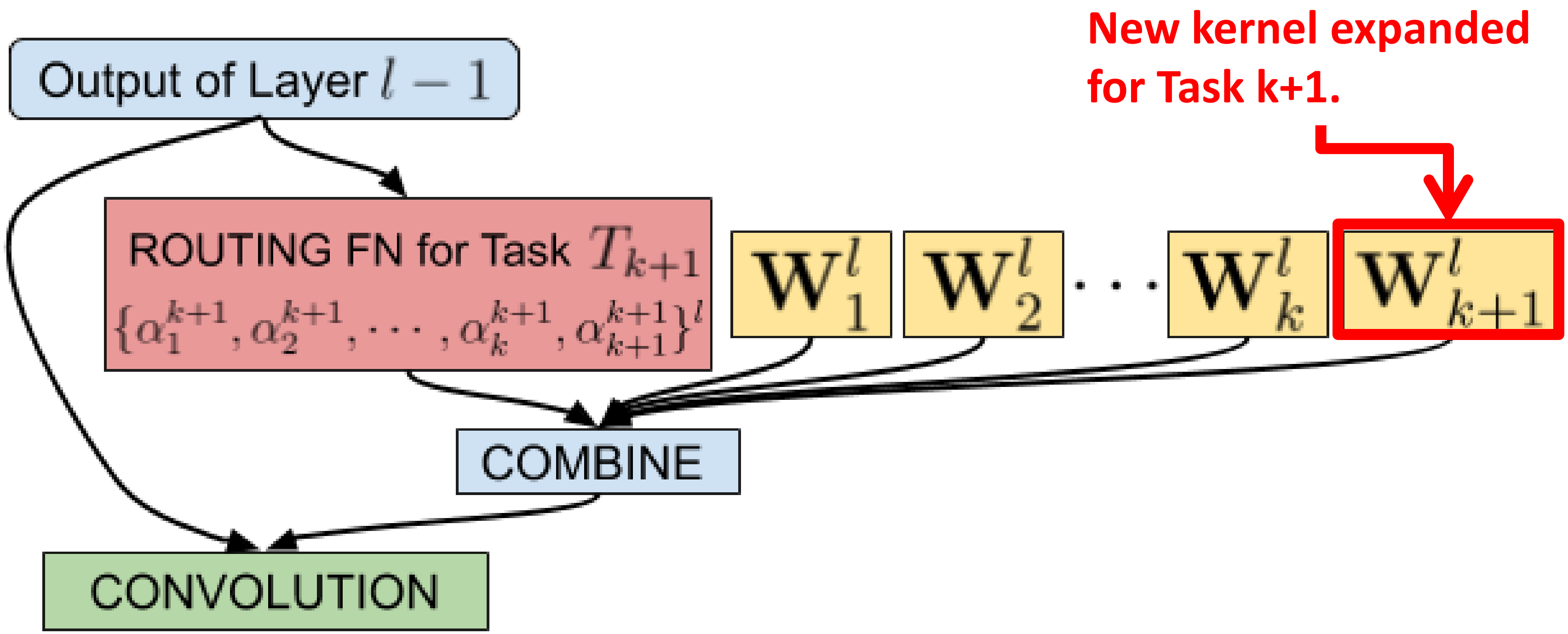
CondConv Continual Learning

- We incorporate CondConv structures into Continual Learning by progressively expanding a new kernel in each CondConv layer when a new task arrives.



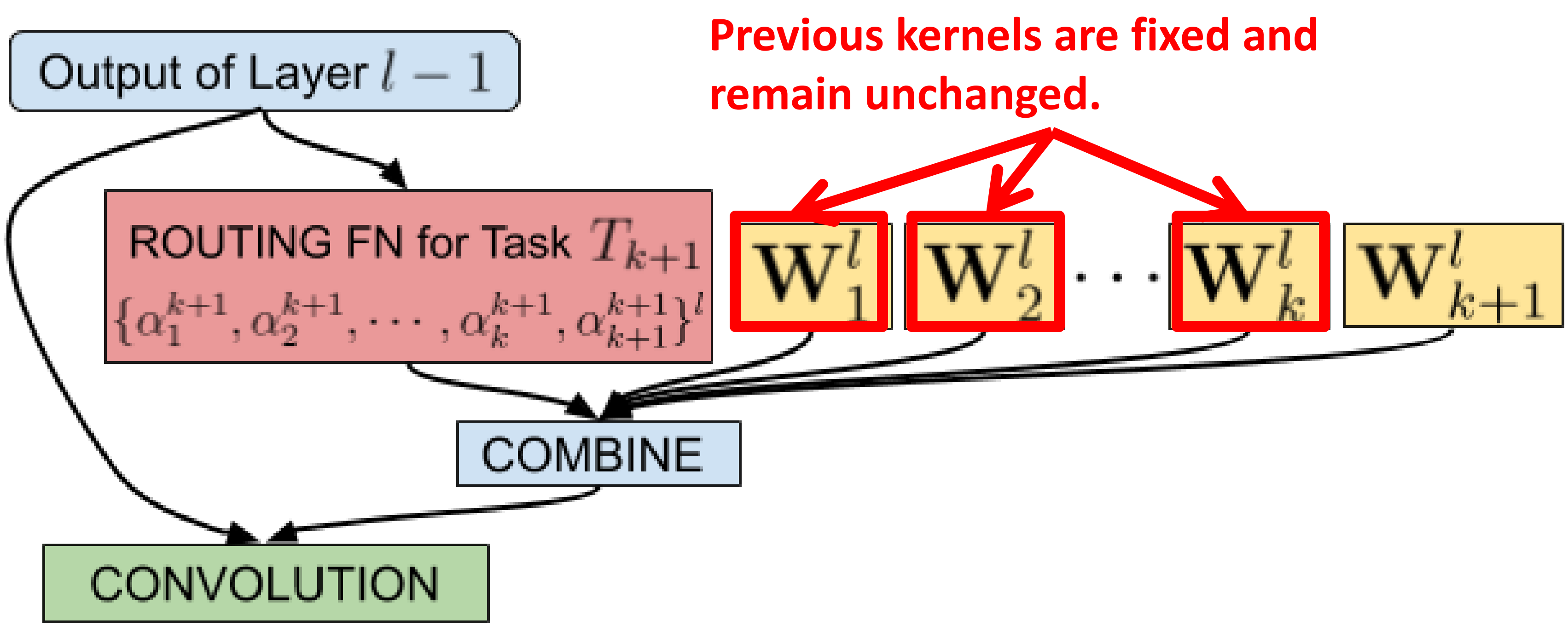
CondConv Continual Learning

- We incorporate CondConv structures into Continual Learning by progressively expanding a new kernel in each CondConv layer when a new task arrives.



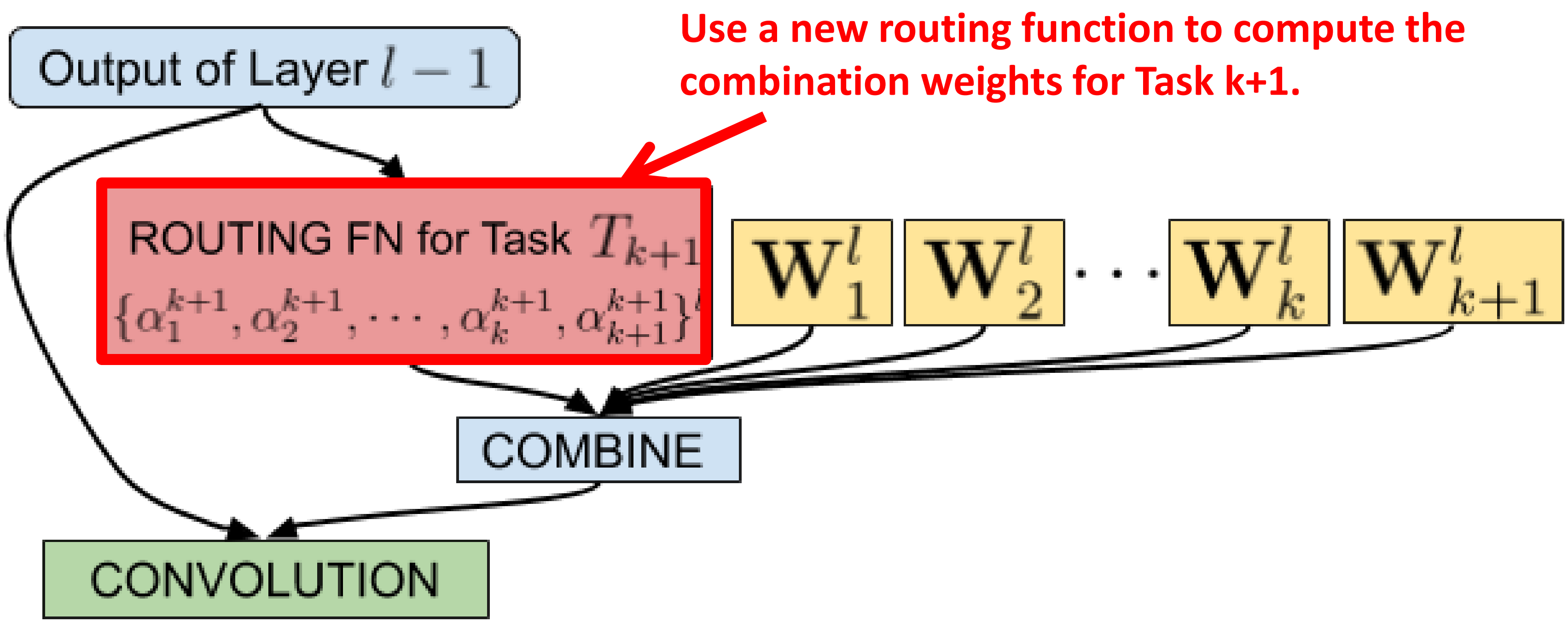
CondConv Continual Learning

- We incorporate CondConv structures into Continual Learning by progressively expanding a new kernel in each CondConv layer when a new task arrives.



CondConv Continual Learning

- We incorporate CondConv structures into Continual Learning by progressively expanding a new kernel in each CondConv layer when a new task arrives.



CondConv Continual Learning

- Although our model size of our model size is linearly proportional to the number of tasks, our model runs efficiently in inference time.
- In [3], CondConv remains efficient even when there are 32 kernels.

[3] B. Yang, G. Bender, Q. V. Le, and J. Ngiam, “Condconv: Conditionally parameterized convolutions for efficient inference,” in Proceedings of Advances in Neural Information Processing Systems, 2019.

Experiments

- On CIFAR100 Twenty Tasks
  - Use a 4-layer convolutional network as backbone
  - Eventually, CPG becomes 16.34x of the original model size, and Our method becomes 20.0x
  - But, in inference time, Our method is 33% faster than CPG

	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}	T_{13}	T_{14}	T_{15}	T_{16}	T_{17}	T_{18}	T_{19}	T_{20}	Avg.
Scratch	65.4	76.0	75.0	78.0	83.0	77.8	79.2	81.8	82.2	86.8	83.4	79.4	84.2	78.4	48.0	68.2	63.8	70.2	85.8	88.6	76.8
Finetuning	65.4	75.4	74.5	74.7	81.2	77.2	73.2	80.4	81.0	84.8	86.0	76.6	81.6	77.5	46.6	67.2	63.2	69.7	84.4	88.6	75.5
CPG[8]	63.6	76.8	76.2	74.4	83.0	79.6	79.2	82.2	80.6	87.0	85.2	77.6	82.4	81.6	51.0	67.8	68.4	67.2	85.8	90.2	77.0
Ours	65.4	77.4	75.2	78.4	81.4	77.6	77.6	82.2	82.2	86.8	85.4	77.8	83.8	80.2	50.6	71.0	67.8	69.8	86.8	91.2	77.4

Experiments

- Fine-grained Six Tasks
  - Use ResNet50 as backbone
  - We only use the 1^{st} ImageNet task to combine the 2^{nd} ~ 6^{th} tasks, and thus we only need to load 2x model size for these tasks.

Dataset	**ImageNet**	**CUBS**	**Stanford Cars**	**Flowers**	**WikiArt**	**Sketch**	**Total Gain**
Finetuning	-	83.41	92.85	97.12	74.19	79.7	-
Scratch	76.16	42.03	62.94	46.24	55.12	69.48	-151.46
ProgressiveNet[10]	76.16	78.94	89.21	93.41	74.94	76.35	-14.42
PackNet[11]	76.16	81.59	89.62	94.77	71.33	79.91	-10.05
Piggyback[12]	76.16	81.59	89.62	94.77	71.33	79.91	-10.05
CPG[8]	75.81	83.59	92.80	96.62	77.15	80.33	$+2.87$
Ours	76.16	84.26	92.61	97.16	78.32	80.77	$+5.85$

Experiments

- ImageNet50 Five Tasks
  - Use ResNet18 as backbone
  - We extend our model to no-task-boundary settings using the observation that images from the distribution similar in training time tend to produce peaked probabilities; otherwise they produce uniform probabilities.

Method	Accuracy
DGMw[5]	17.82
DGMa[5]	15.16
CCGN[14]	35.24
Ours	61.32

Conclusion

- We propose to use CondConv structures in Continual Learning to enhance the inference efficiency under network expansion.
- Our method achieves competitive or better performance compared with others in both task-boundary and no-task-boundary settings.