Heterogeneous Continual Learning

Divyam Madaan, Hongxu Yin, Wonmin Byeon, Jan Kautz, and Pavlo Molchanov

CVPR 2023 Highlight

WED-PM-346

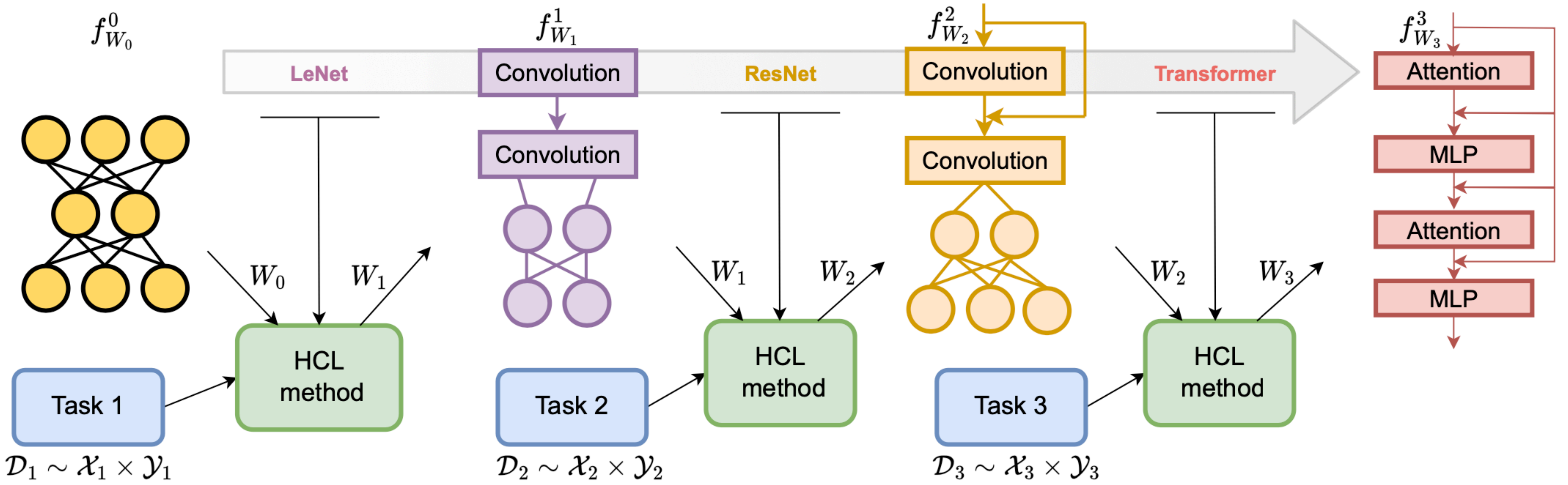
NYU

JUNE 18-22, 2023 CVPR VANCOUVER, CANADA

NVIDIA

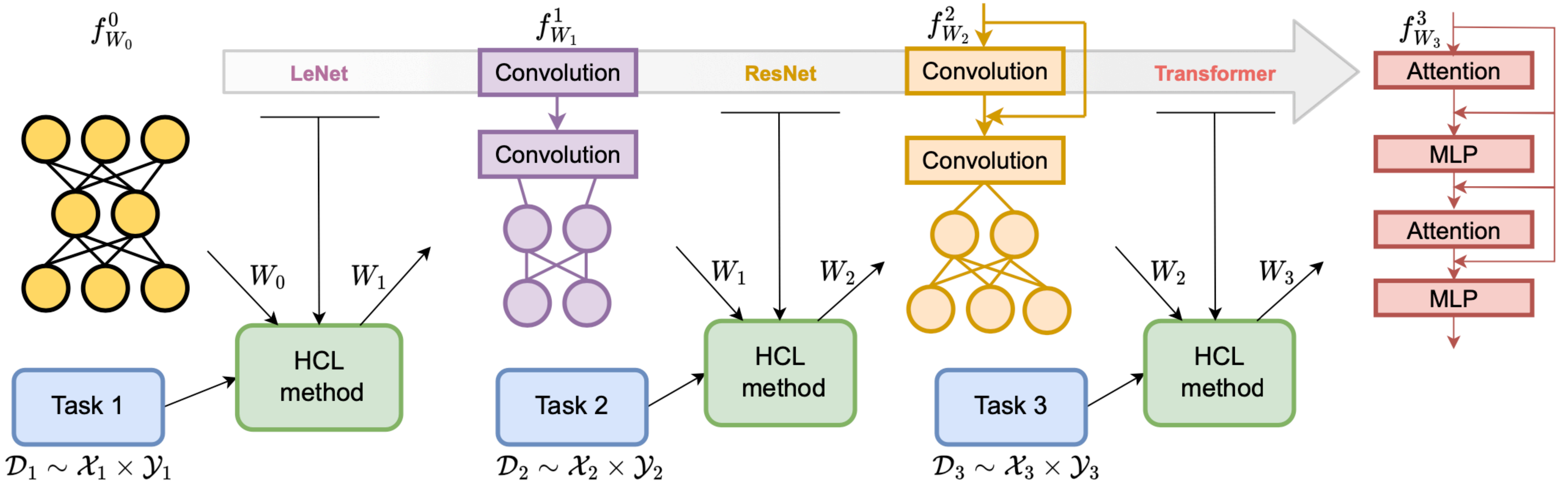
Heterogeneous Continual Learning

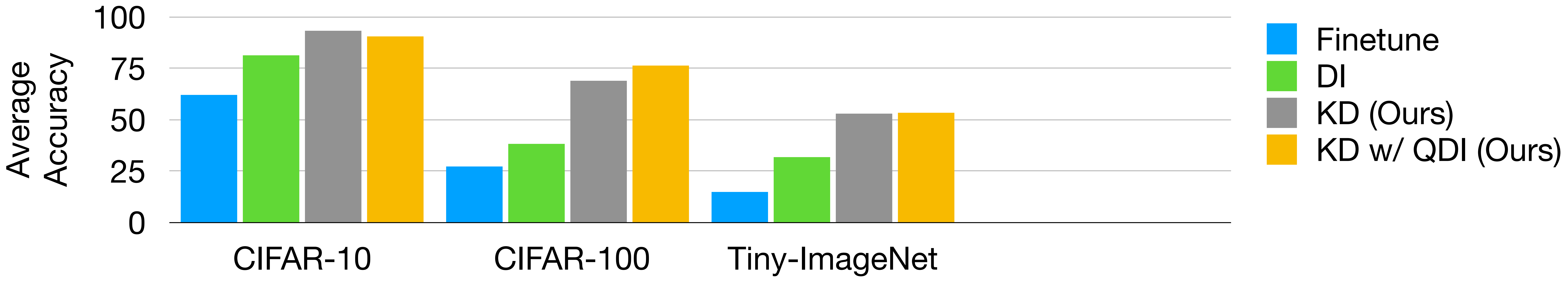
We learn continual representations ***without any constraints*** on the training configurations network architecture.



Heterogeneous Continual Learning

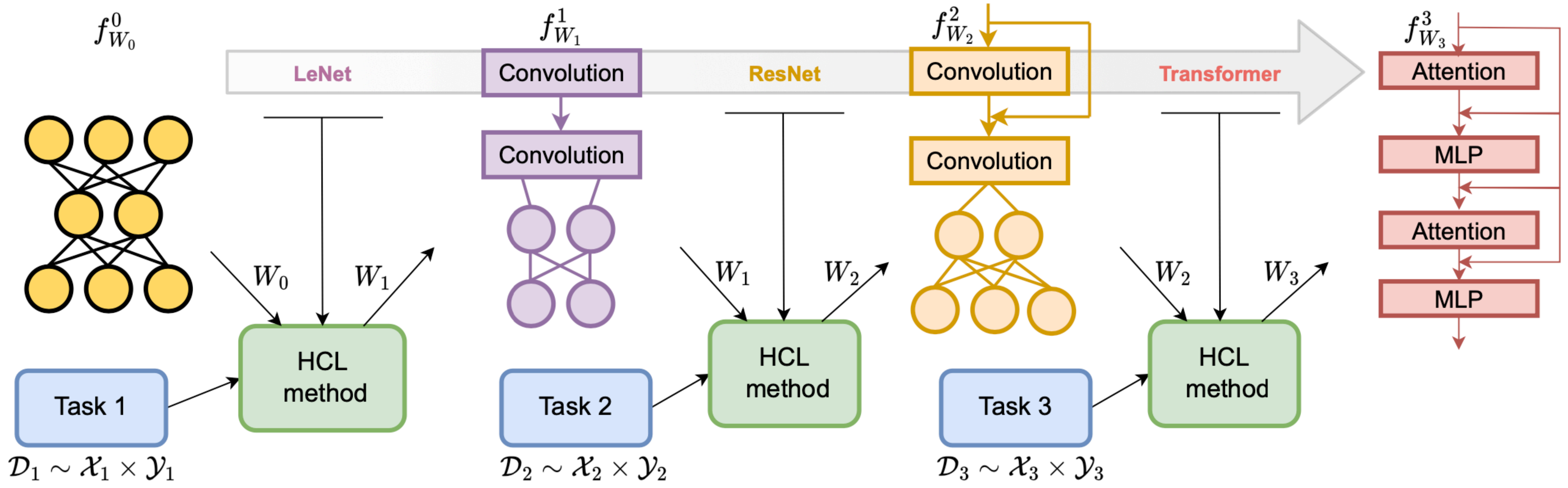
We outperform prior works on ***diverse stream of architectures*** for both task-incremental and class-incremental continual learning.

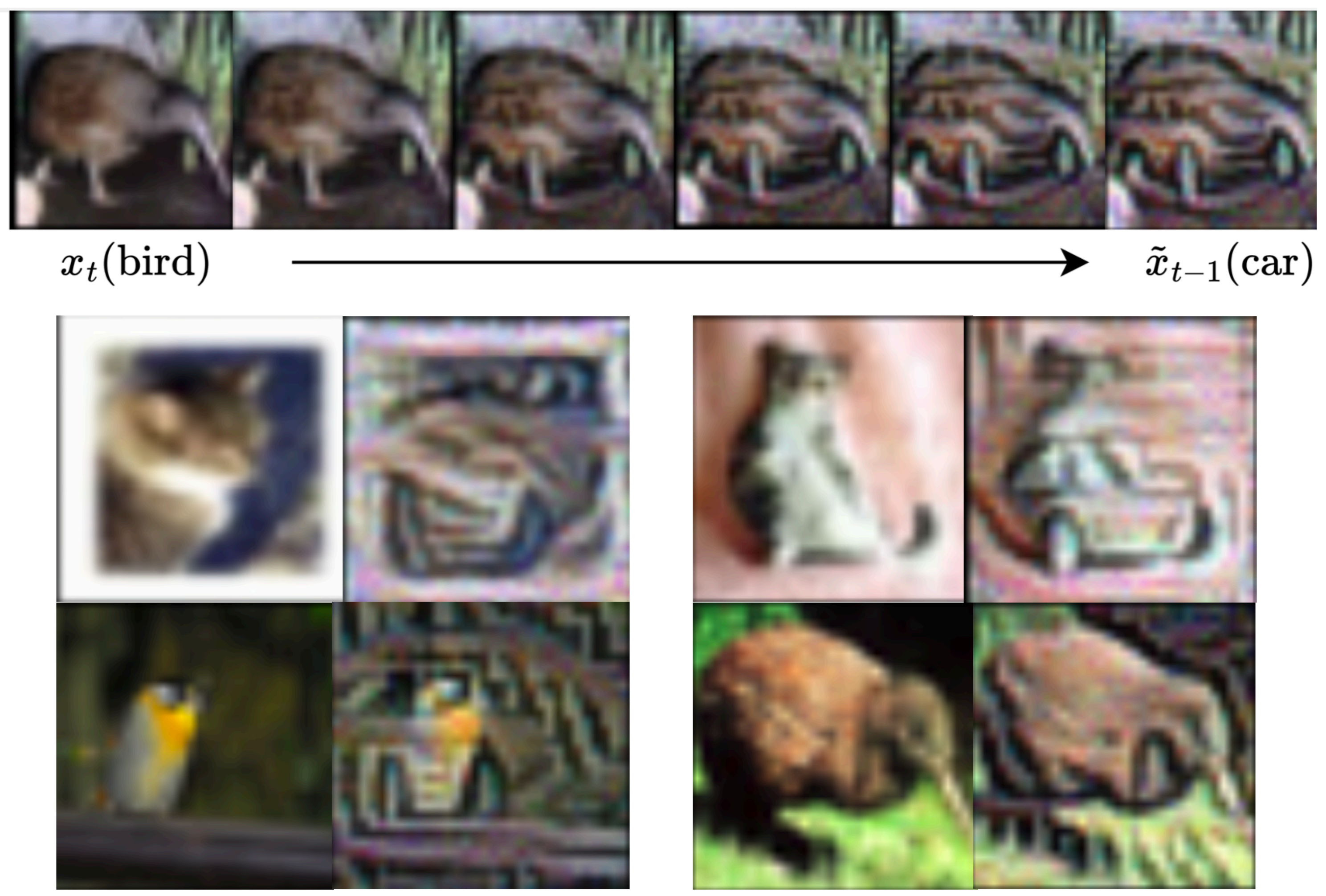


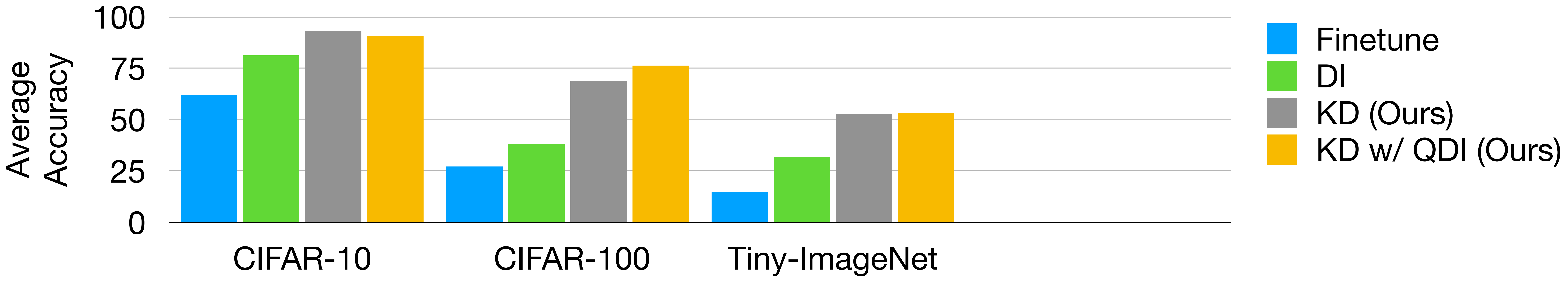


Heterogeneous Continual Learning

We ***interpolate*** between the ***current task*** and ***previous task instances***, which promotes current task adaptation while minimizing catastrophic forgetting.

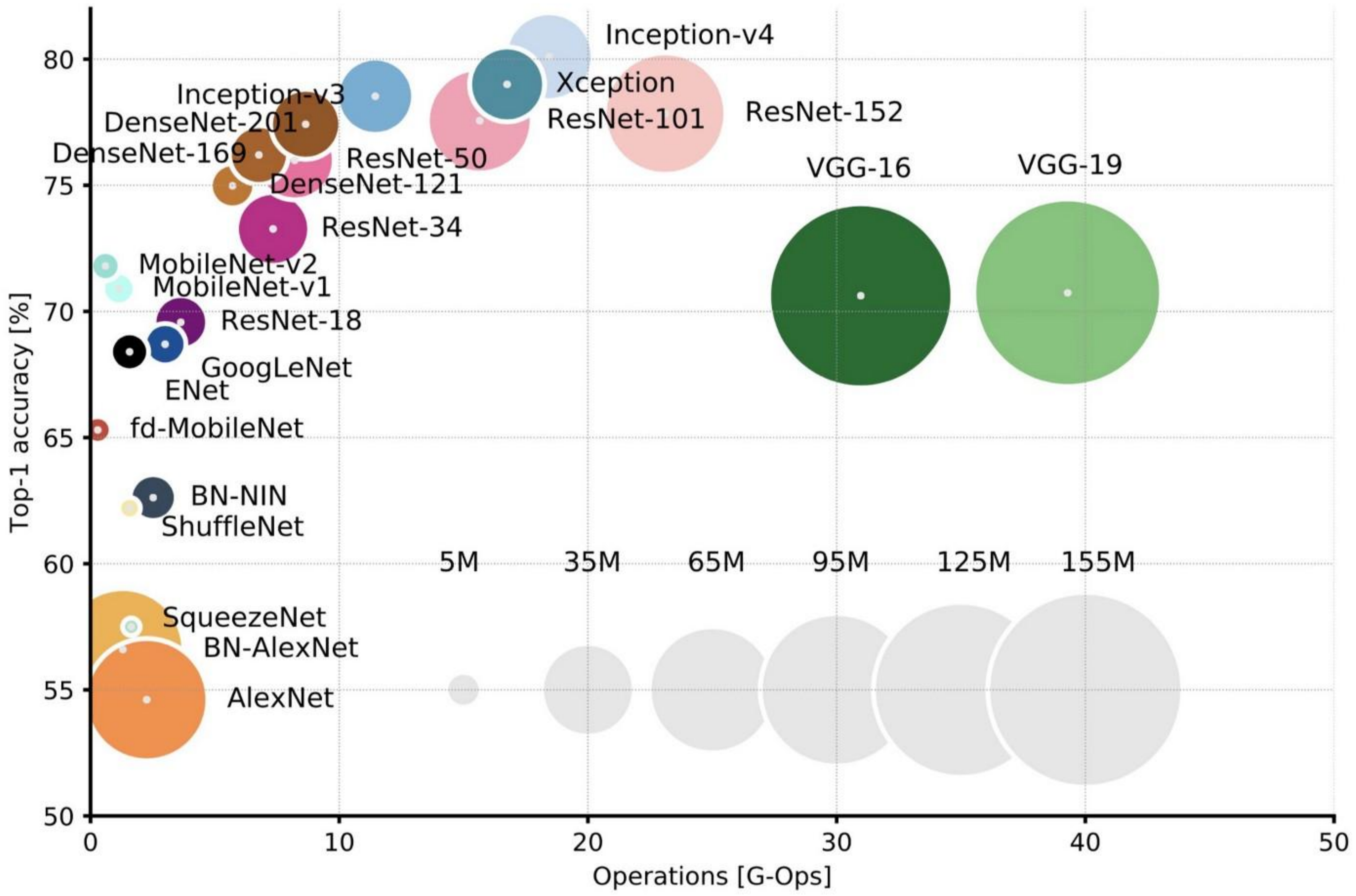






Continual Change in Model Architecture

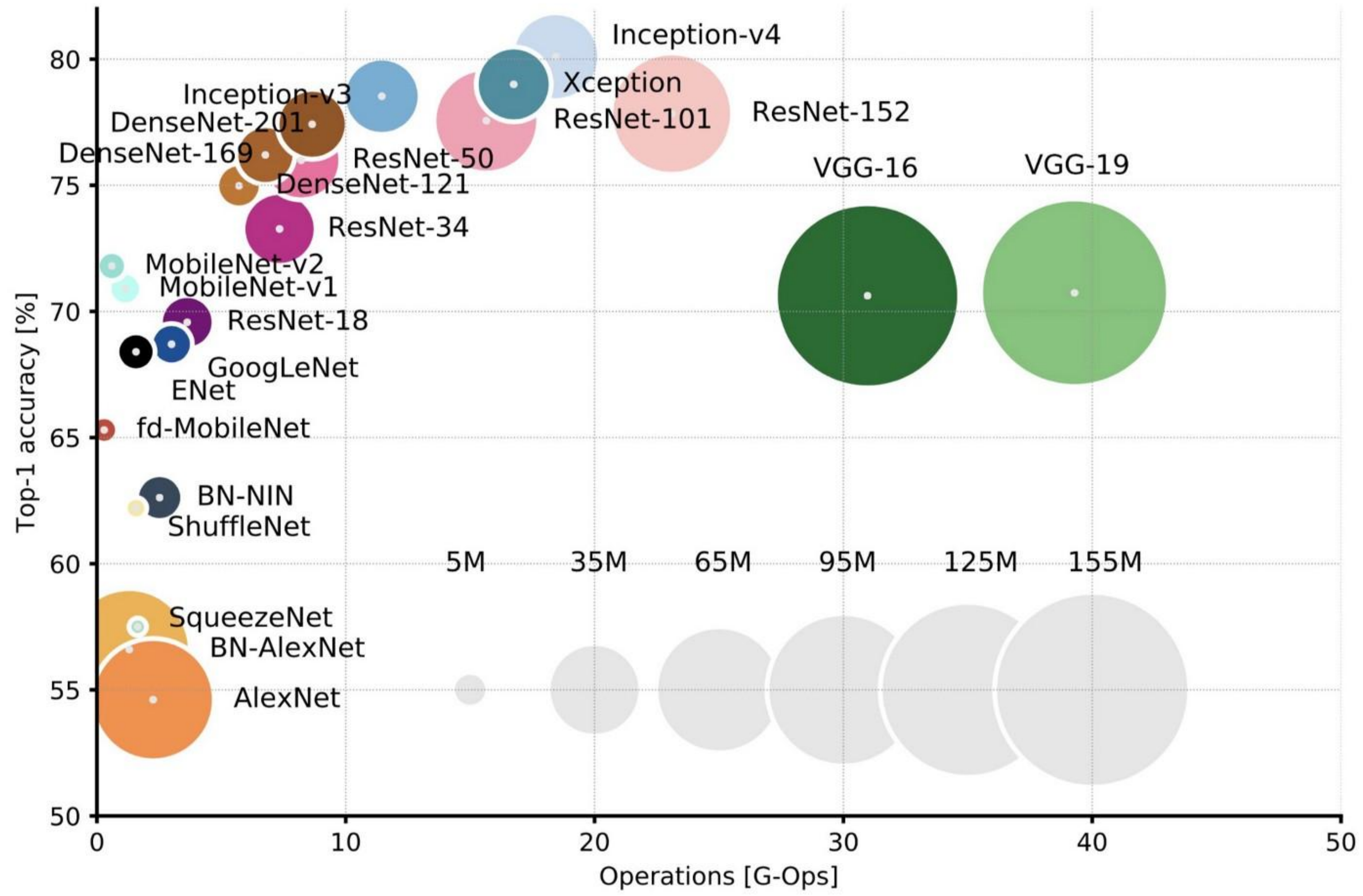
In recent years, many ***new architectures*** have been developed and deep learning has been applied to ***various tasks*** and ***modalities***.

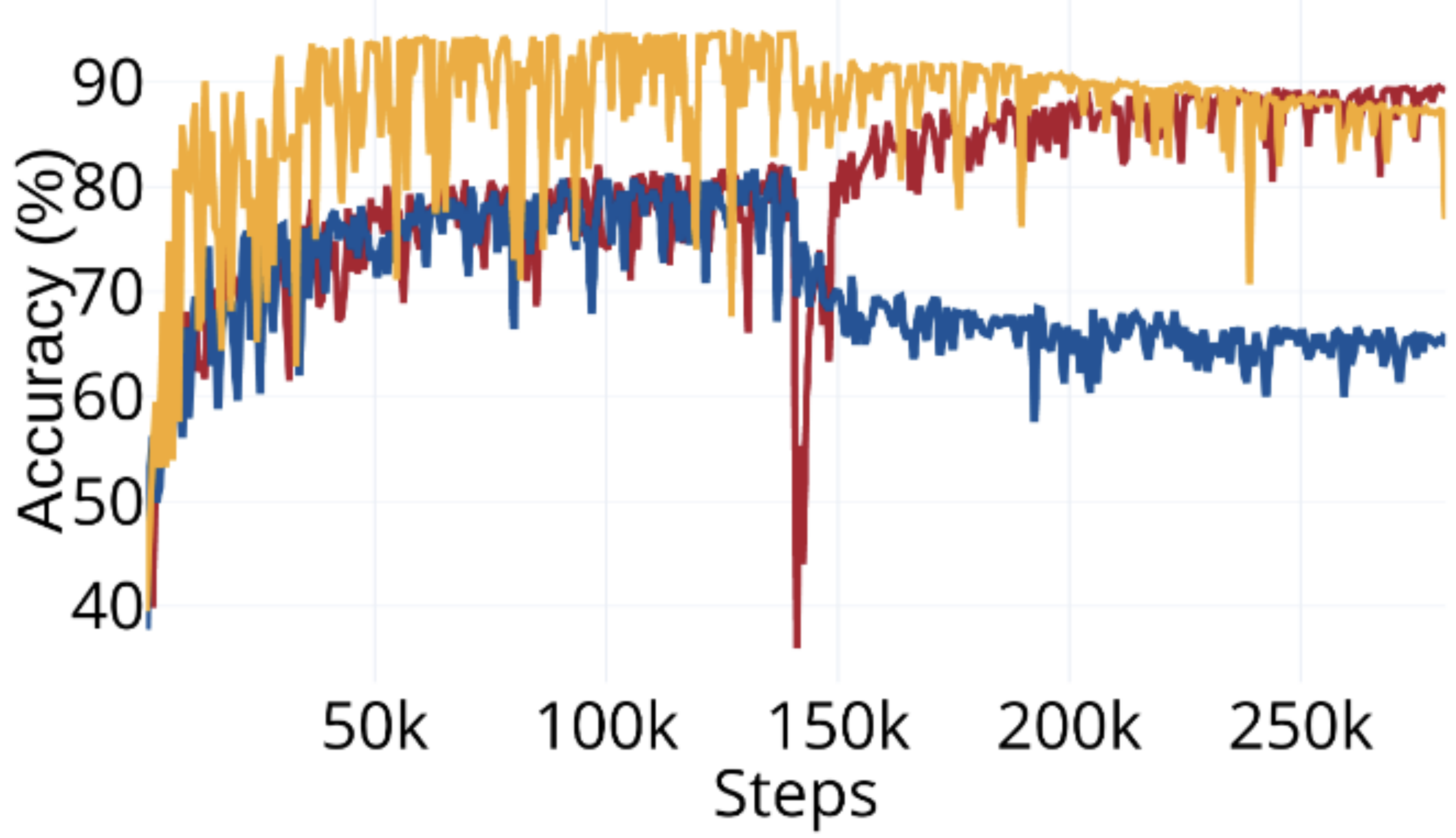


[Canziani et a. 2016] An Analysis of Deep Neural Network Models for Practical Applications, 2016

Continual change in model architecture

In recent years, many ***new architectures*** have been developed and deep learning has been applied to ***various tasks*** and ***modalities***.





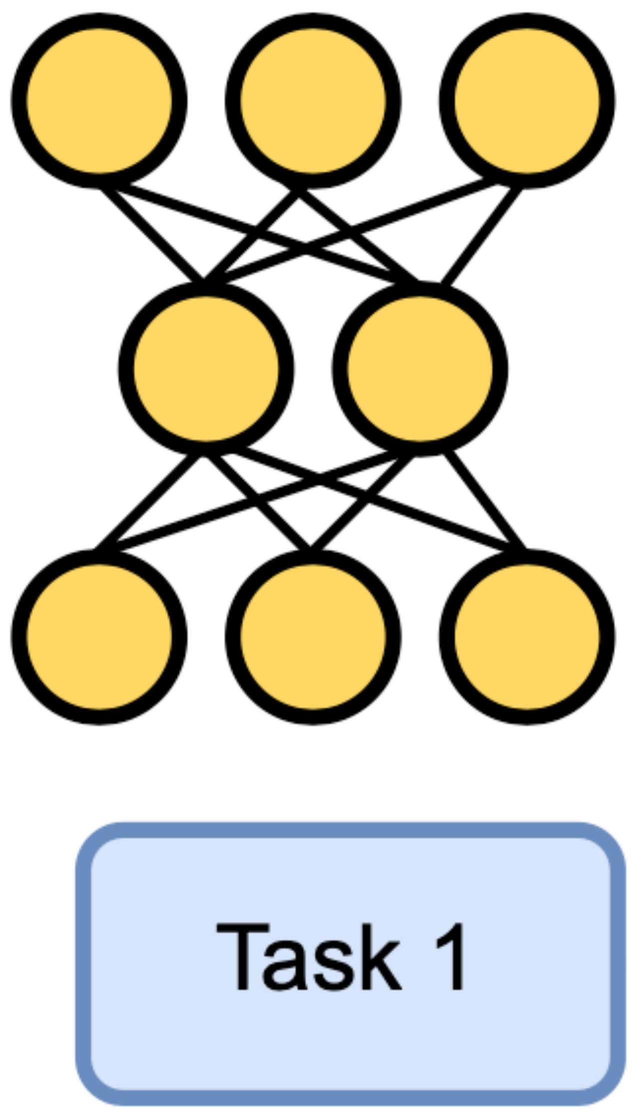
Comparison between standard CL -- ResNet18 and LeNet w/ buffer, **ours HCL** w/o buffer on two tasks.

[Canziani et a. 2016] An Analysis of Deep Neural Network Models for Practical Applications, 2016

Standard Continual Learning

Continually learn f_{W_t} with a ***fixed representation structure*** on a sequence of T tasks.

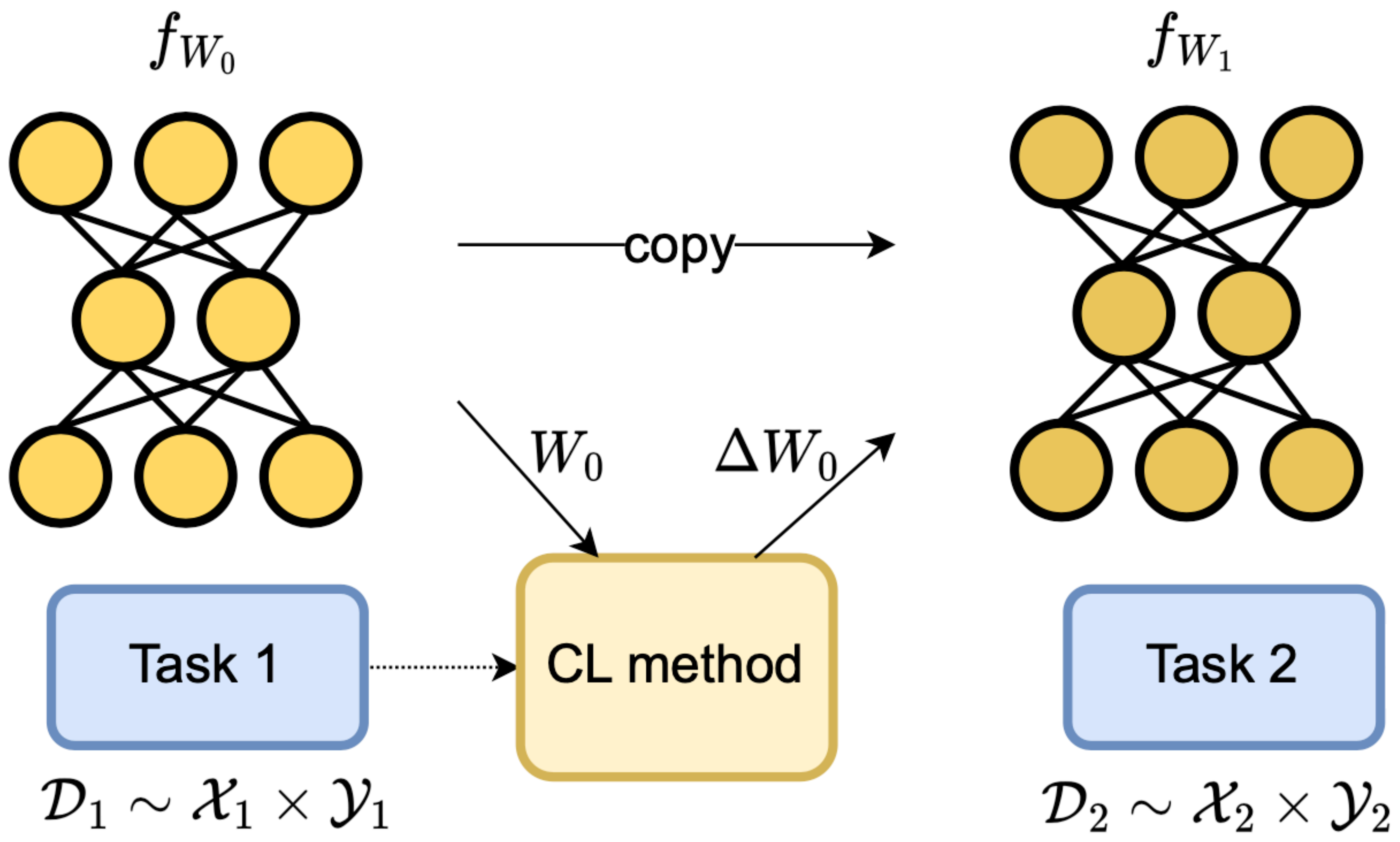
f_{W_0}



$\mathcal{D}_1 \sim \mathcal{X}_1 \times \mathcal{Y}_1$

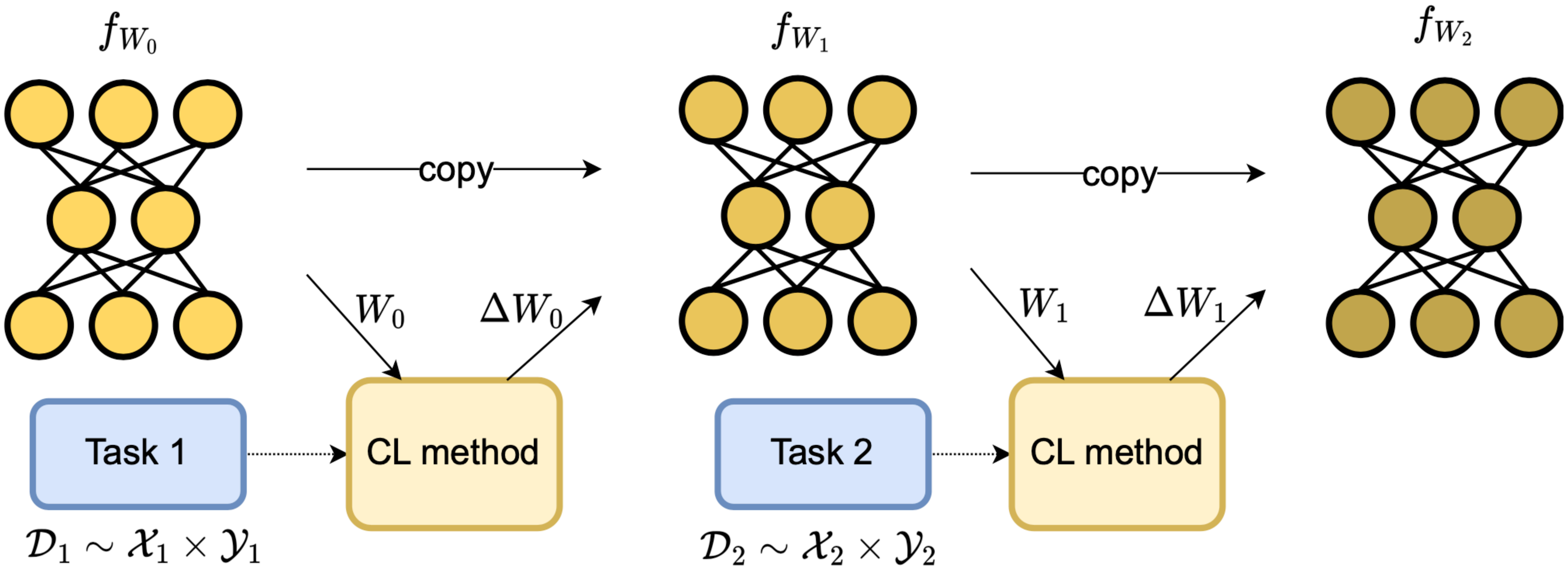
Standard Continual Learning

Continually learn f_{W_t} with a ***fixed representation structure*** on a sequence of T tasks.



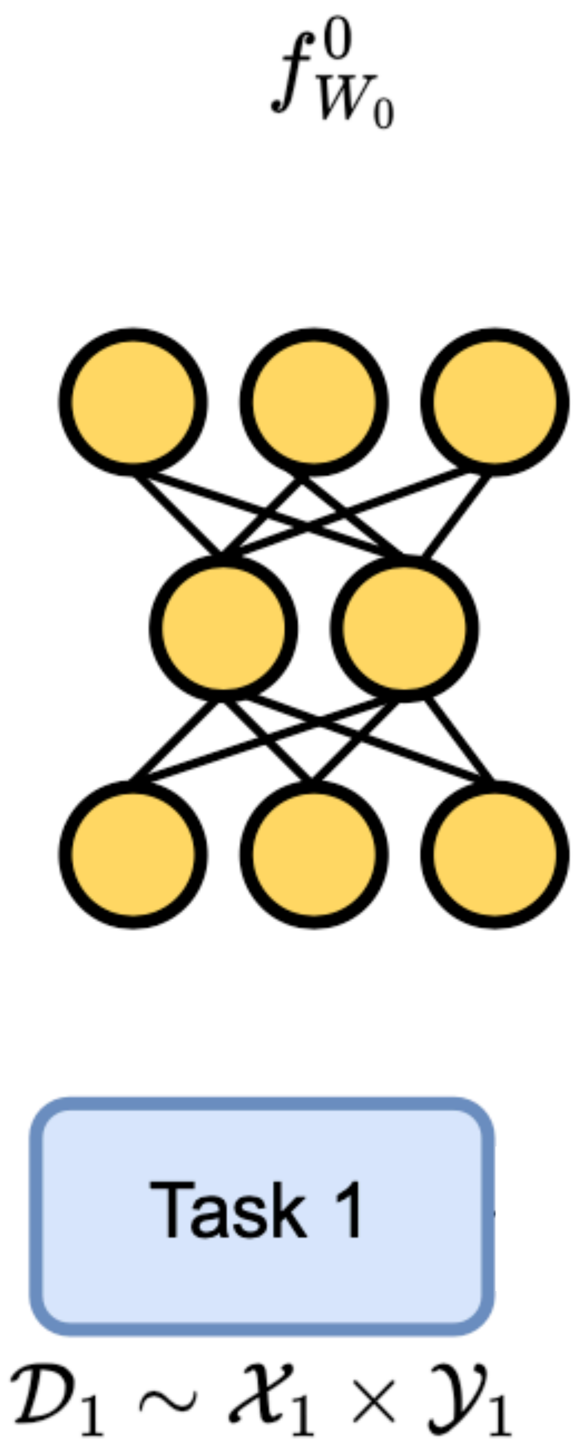
Standard Continual Learning

Continually learn f_{W_t} with a ***fixed representation structure*** on a sequence of T tasks.



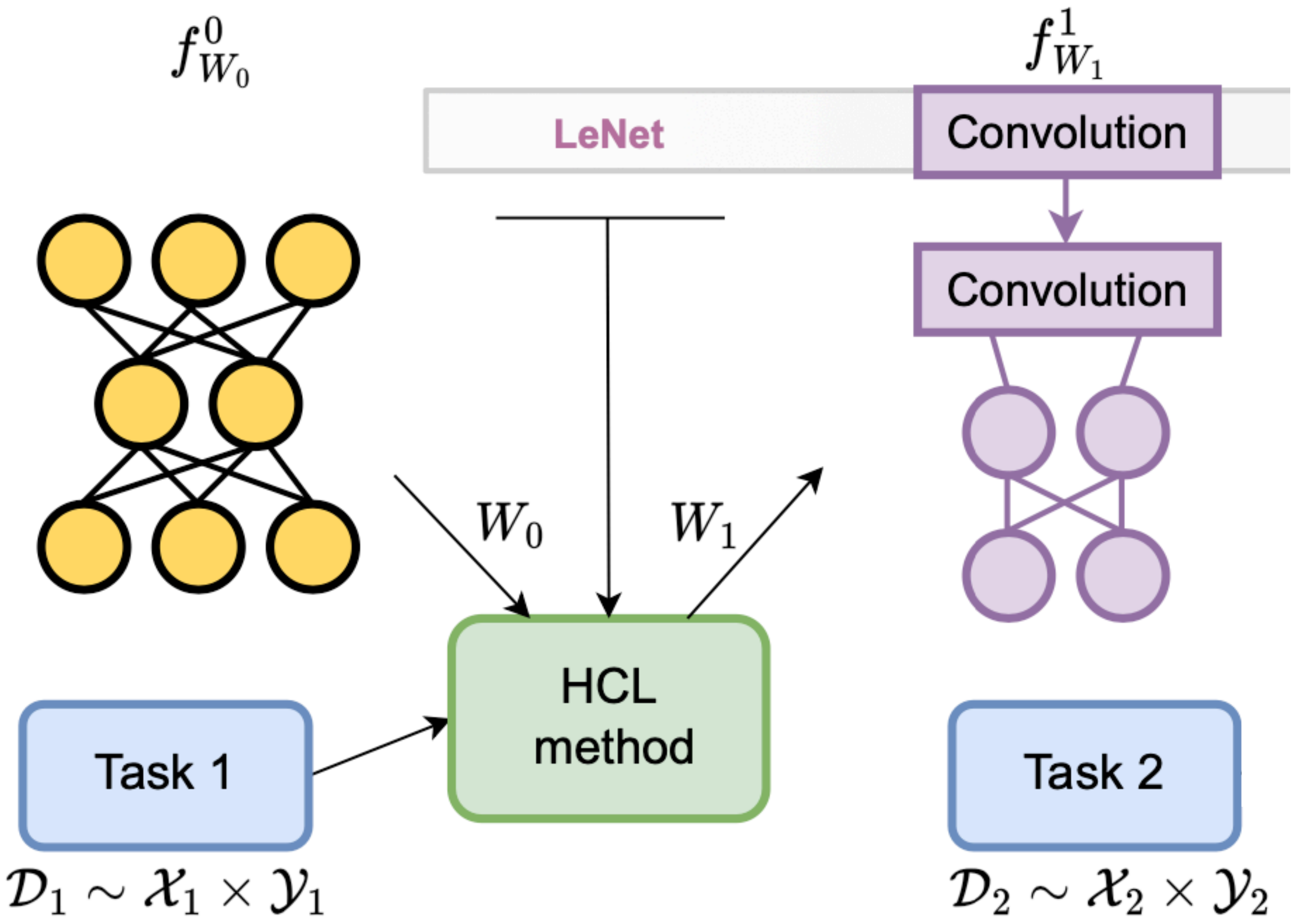
Heterogeneous Continual Learning

Consider continual learner as a ***stream of architectures***



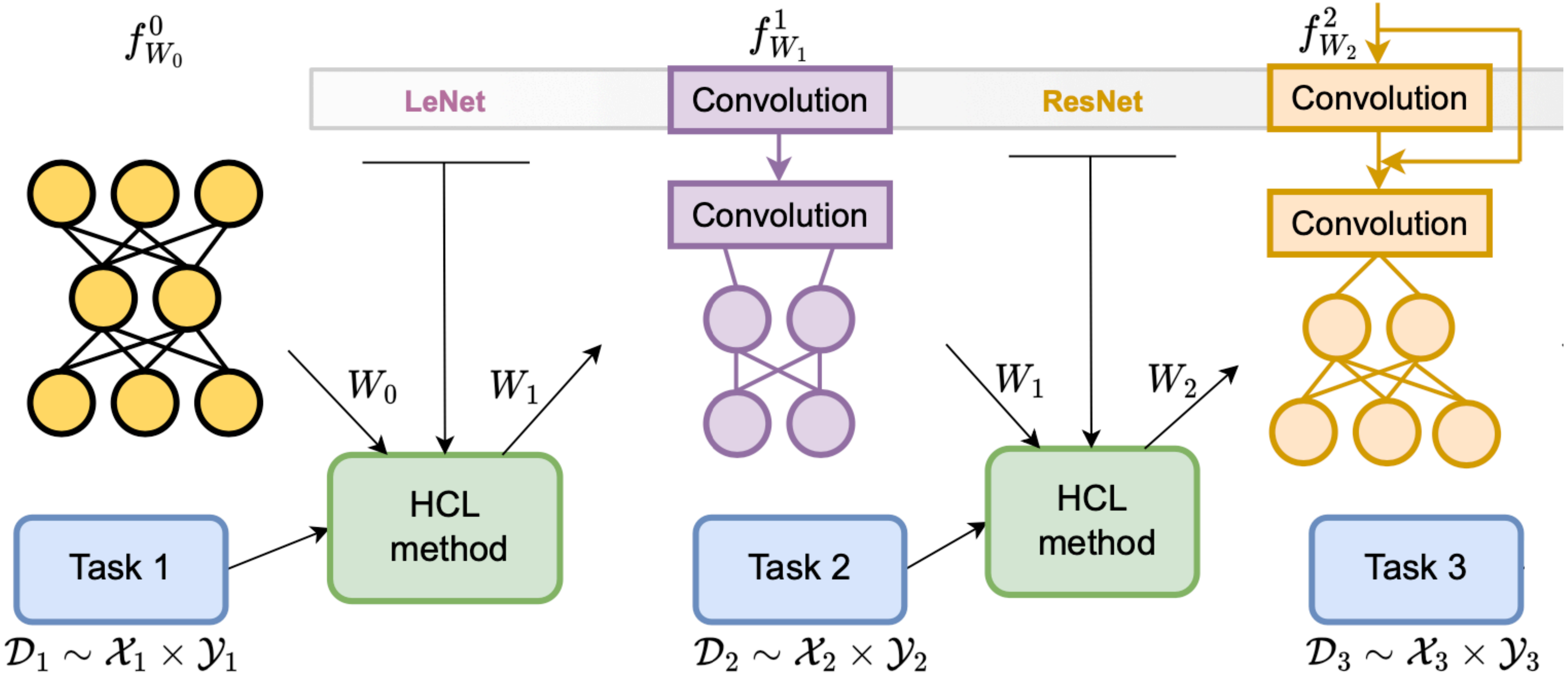
Heterogeneous Continual Learning

Consider continual learner as a ***stream of architectures***, where it can ***change*** the backbone architecture



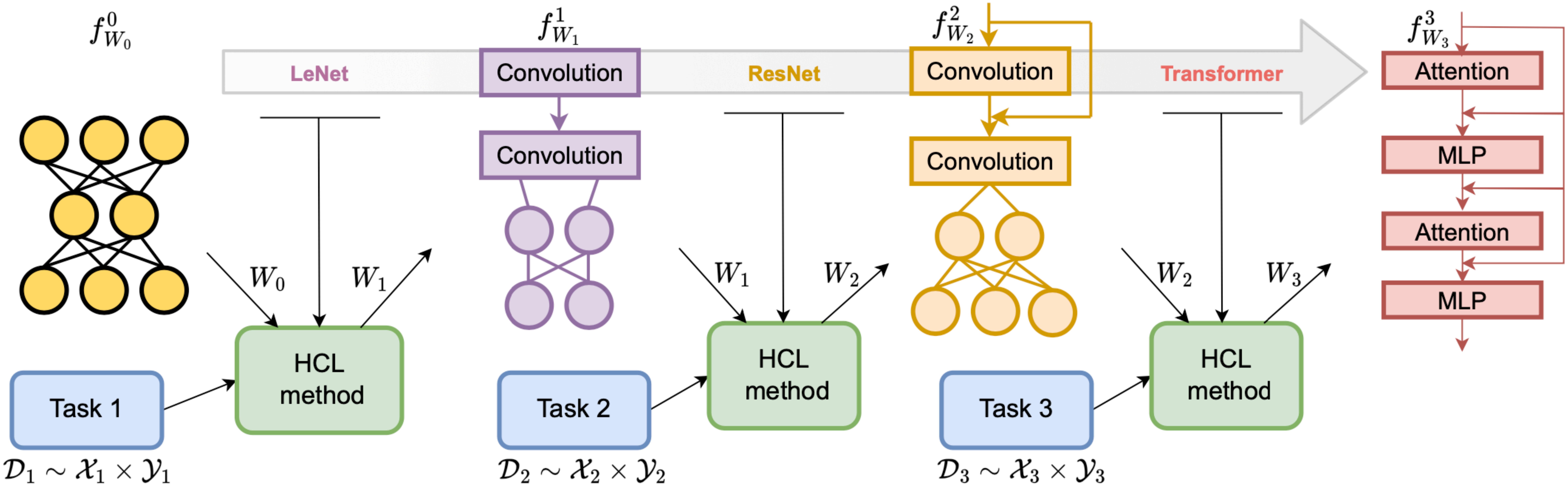
Heterogeneous Continual Learning

Consider continual learner as a ***stream of architectures***, where it can ***change*** the backbone architecture to incorporate ***recent architectural advancements***.



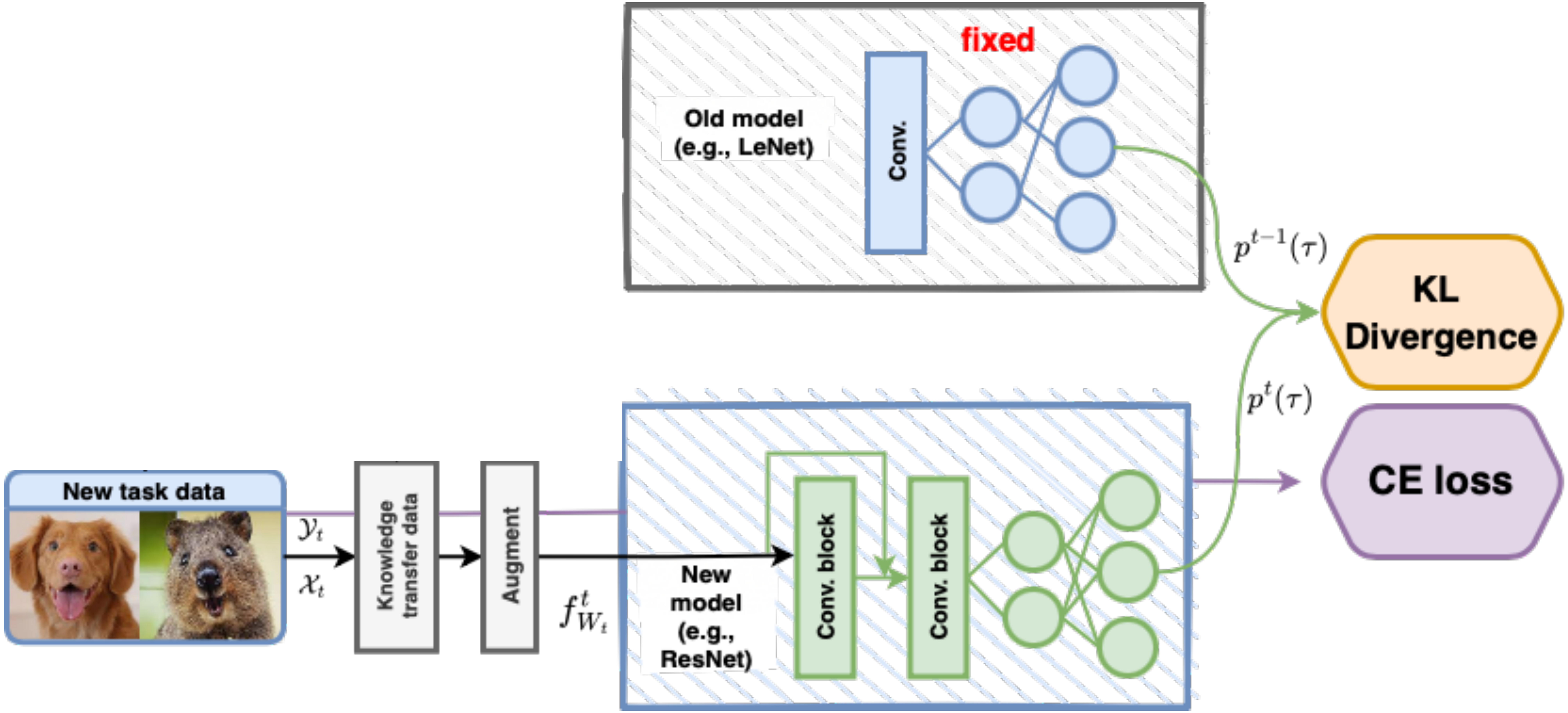
Heterogeneous Continual Learning

Consider continual learner as a ***stream of architectures***, where it can ***change*** the backbone architecture to incorporate ***recent architectural advancements***.



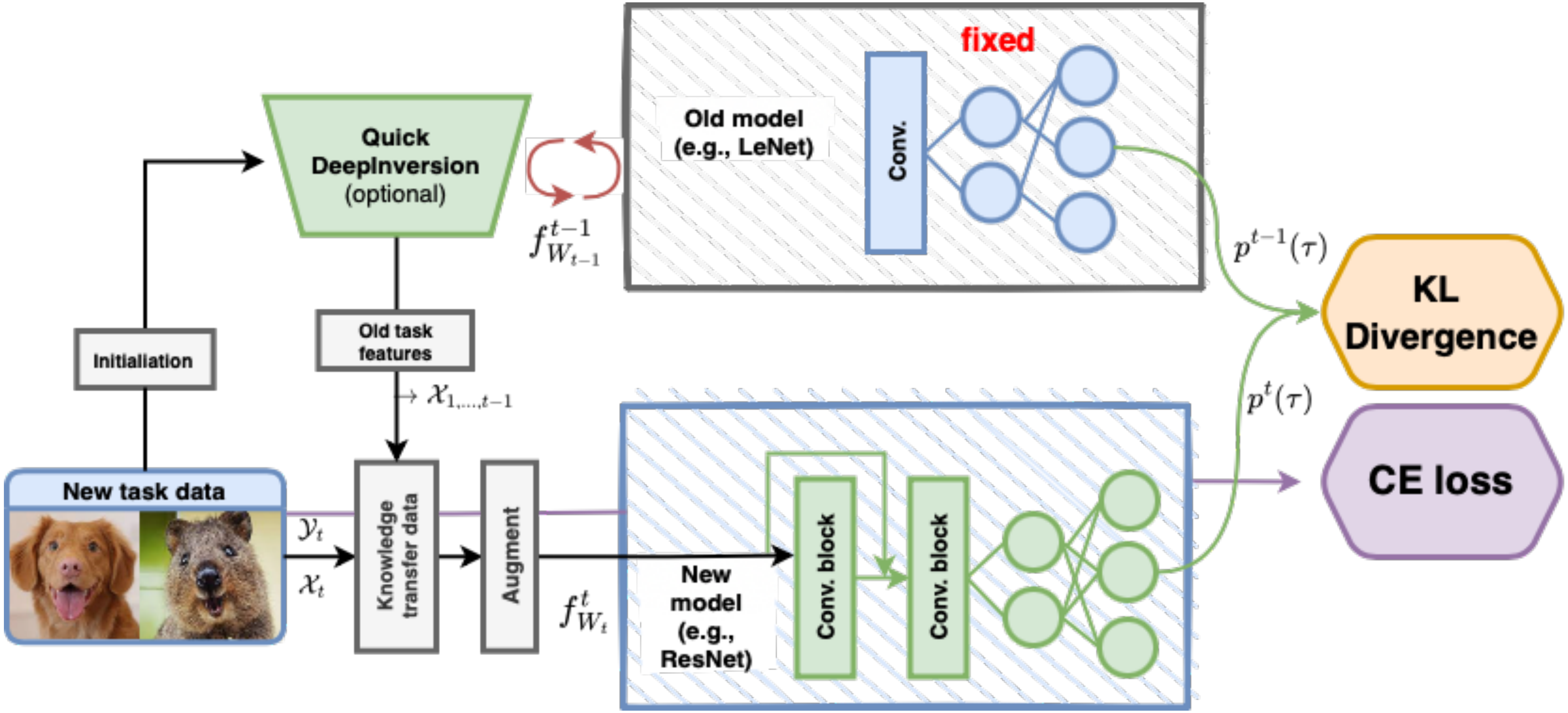
Knowledge Transfer Module

We ***enforce similarity*** between previous and current task model using current task data.



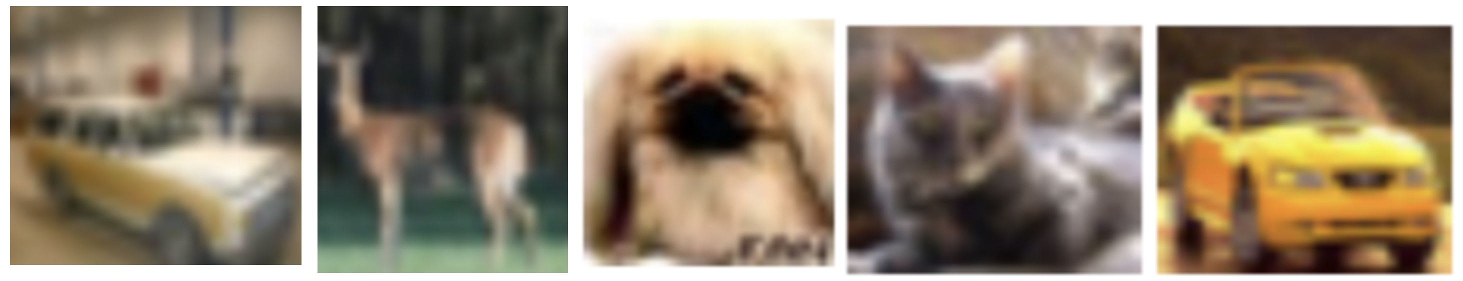
Knowledge Transfer Module

We ***enforce similarity*** between previous and current task model using current task data.



Experimental Setup

Datasets



1) **Split CIFAR-10 [Krizhevsky 2012]** We consider ***five tasks*** with a ***different architecture*** for each task for HCL.

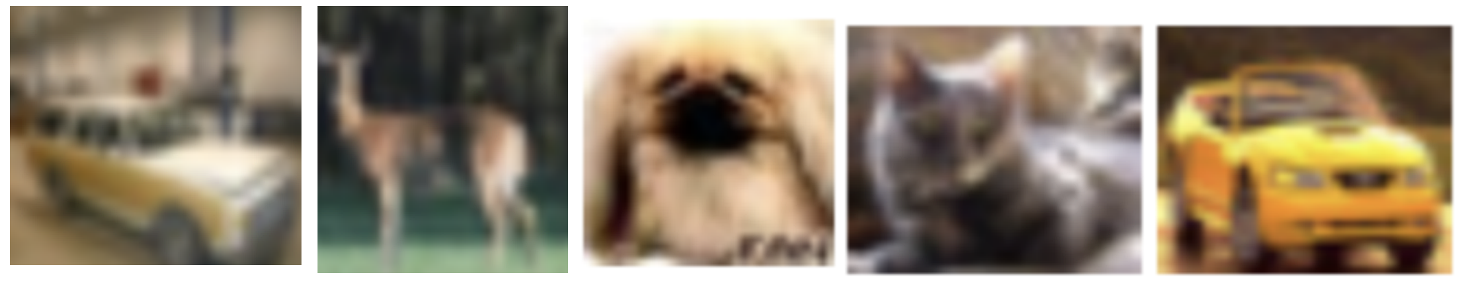
2) **Split CIFAR-100 [Krizhevsky 2012]** We consider ***20 tasks***, each with five classes with ***nine architectures*** for HCL setup.

3) **Split Tiny-ImageNet [Russakovsky 2015]** We consider ***10 tasks***, each with ***20 classes*** with ***five architectures***, each with two tasks.

[Krizhevsky 2012] Krizhevsky, A. Learning multiple layer of features from tiny images. University of Toronto 2012
[Ruakovsk 2015] Imagenet large scale visual recognition challenge. International journal of computer vision, 2015

Experimental Setup

Datasets



1) **Split CIFAR-10 [Krizhevsky 2012]** We consider ***five tasks*** with a ***different architecture*** for each task for HCL.

2) **Split CIFAR-100 [Krizhevsky 2012]** We consider ***20 tasks***, each with five classes with ***nine architectures*** for HCL setup.

3) **Split Tiny-ImageNet [Russakovsky 2015]** We consider ***10 tasks*** with ***20 classes*** each. We use ***five architectures*** for two tasks each.

Metrics

1) **Accuracy** is the average test accuracy of all the tasks completed until the continual learning of task τ

$$A_\tau = \frac{1}{\tau}\sum_{i=1}^{\tau} a_{\tau,i}$$

1) **Forgetting** is the average performance decrease of each task between its maximum accuracy and accuracy at the completion of training

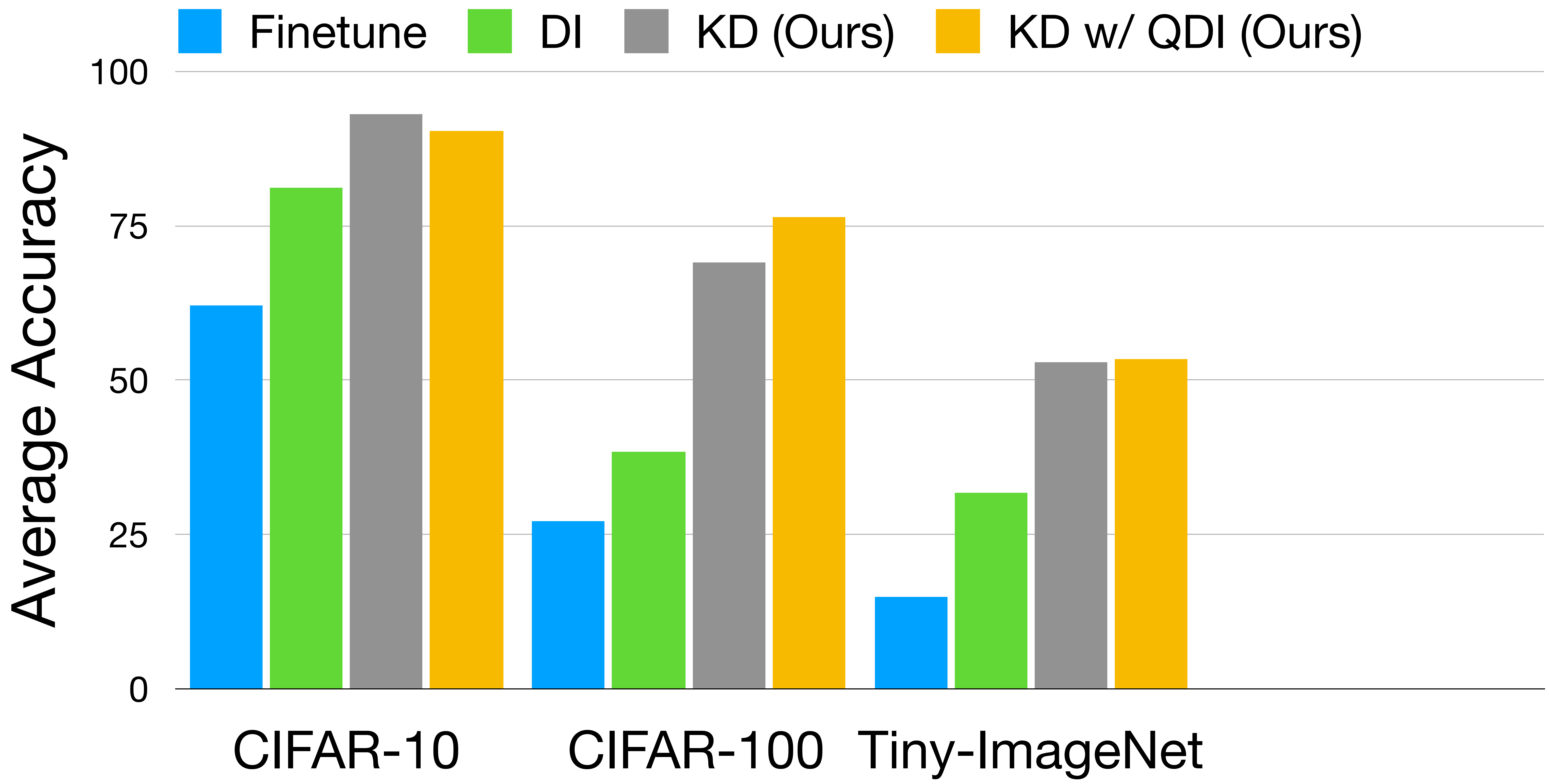
$$F = \frac{1}{T-1}\sum_{i=1}^{T-1} \max_{\tau\in\{1,\ldots,T\}} \left(a_{\tau,i} - a_{T,i}\right)$$

$a_{\tau,i}$ is the test accuracy of task i after learning task τ

[Krizhevsky 2012] Krizhevsky, A. Learning multiple layer of features from tiny images. University of Toronto 2012
[Ruakovsk 2015] Imagenet large scale visual recognition challenge. International journal of computer vision, 2015

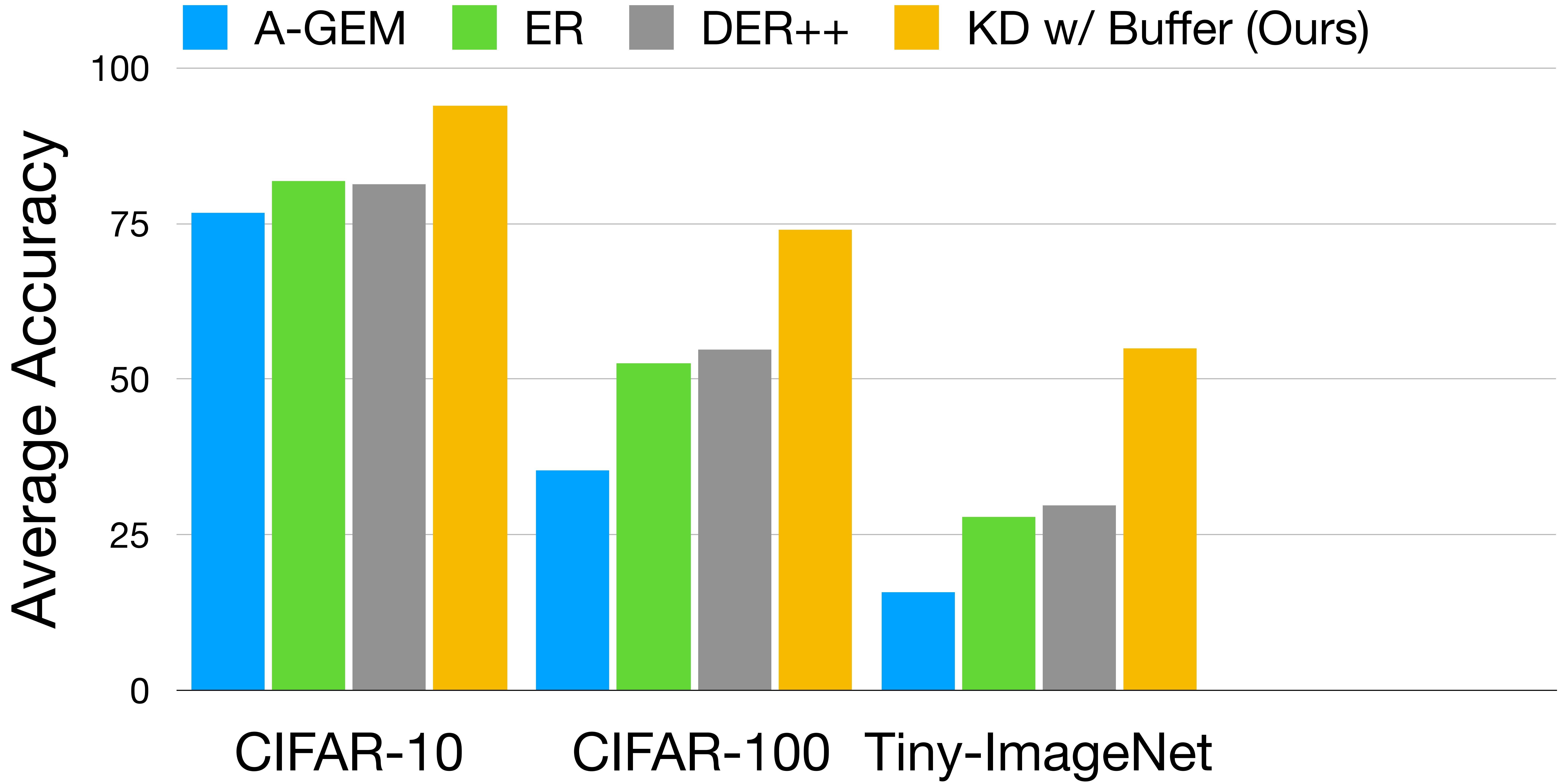
Heterogeneous continual learning evaluation

We obtain ***similar gains*** for HCL when compared with methods without replay buffer.



Heterogeneous continual learning evaluation

We obtain ***similar gains*** for HCL when compared with methods with replay buffer.



Heterogeneous continual learning evaluation

Generated examples have a ***combination*** of current task (cat or bird) and past task (car).

x_t(bird) $\longrightarrow$ $\tilde{x}_{t-1}$(car)

DI (Yin *et al.*)
2k optimization steps

QDI (ours)
0.5k optimization steps **4x faster**

Conclusion

- We propose ***Heterogeneous Continual Learning (HCL)*** where the lifelong learner learns on a sequence of tasks while ***adopting*** state-of-the-art deep-learning techniques and architectures.
- We revisit ***knowledge distillation*** and propose a ***modified paradigm*** inspired by the recent advances in knowledge distillation.
- Additionally, we propose ***Quick Deep Inversion (QDI)*** that generates synthetic examples using current task instances to enhance knowledge distillation performance in the ***data-free continual learning*** setup at a additional cost.
- We believe that our paper can be an essential part towards training ***continually changing network architectures***.

Code will be available at **https://github.com/NVlabs/HCL**

Thank you