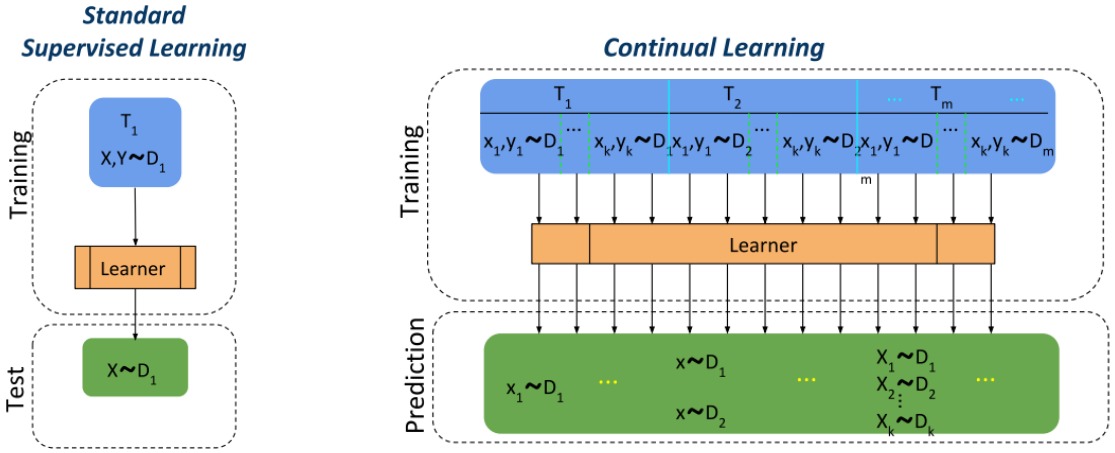
Representational Continuity for Unsupervised Continual Learning

Divyam Madaan[1*] **Jaehong Yoon**[2,3 †] **Yuanchun Li**[5,6] **Yunxin Liu**[5,6] **Sung Ju Hwang**[2,4]
New York University[1] KAIST[2] Microsoft Research[3] AITRICS[4]
Institute for AI Industry Research (AIR)[5] Tsinghua University[6]
divyam.madaan@nyu.edu, {jaehong.yoon,sjhwang82}@kaist.ac.kr
liyuanchun@air.tsinghua.edu.cn, liuyunxin@air.tsinghua.edu.cn

ICLR 2022 (Oral)

Background

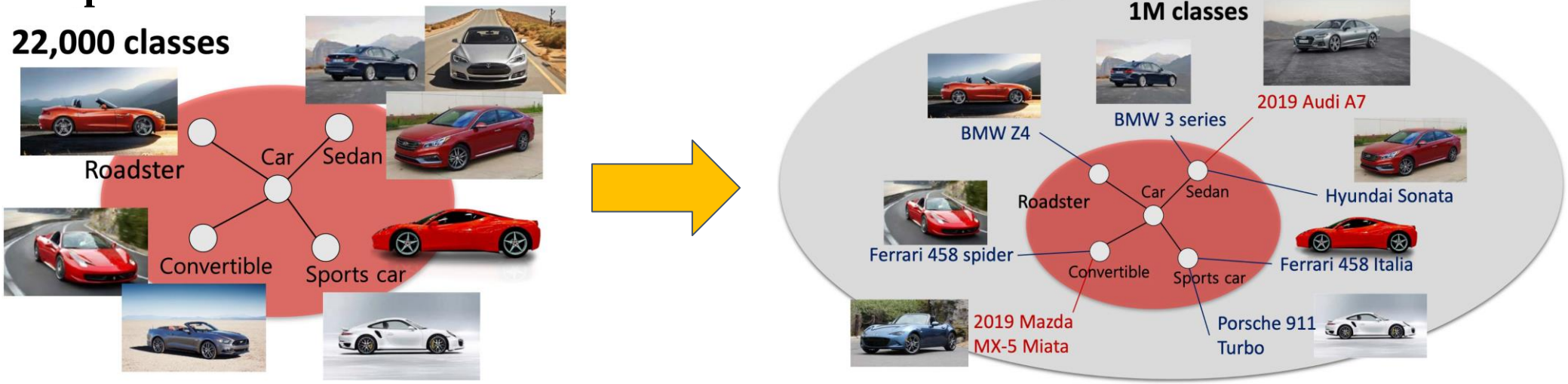
- As humans live, they acquire more knowledges through new experiences.
- In machine perspective, continual learning focuses on **how to learn new knowledge from new incoming data with preserving previous knowledge**.



Non-stationary data comes one example at a time in a stream:
$(x_1, y_1, t_1), \ldots, (x_i, y_i, t_i), \ldots, (x_{i+j}, y_{i+j}, t_{i+j})$

Data is locally i.i.d. - Samples for a task are drawn from the same unknown joint probability distribution
$x_i, y_i \sim P_t(x, y)$

Examples



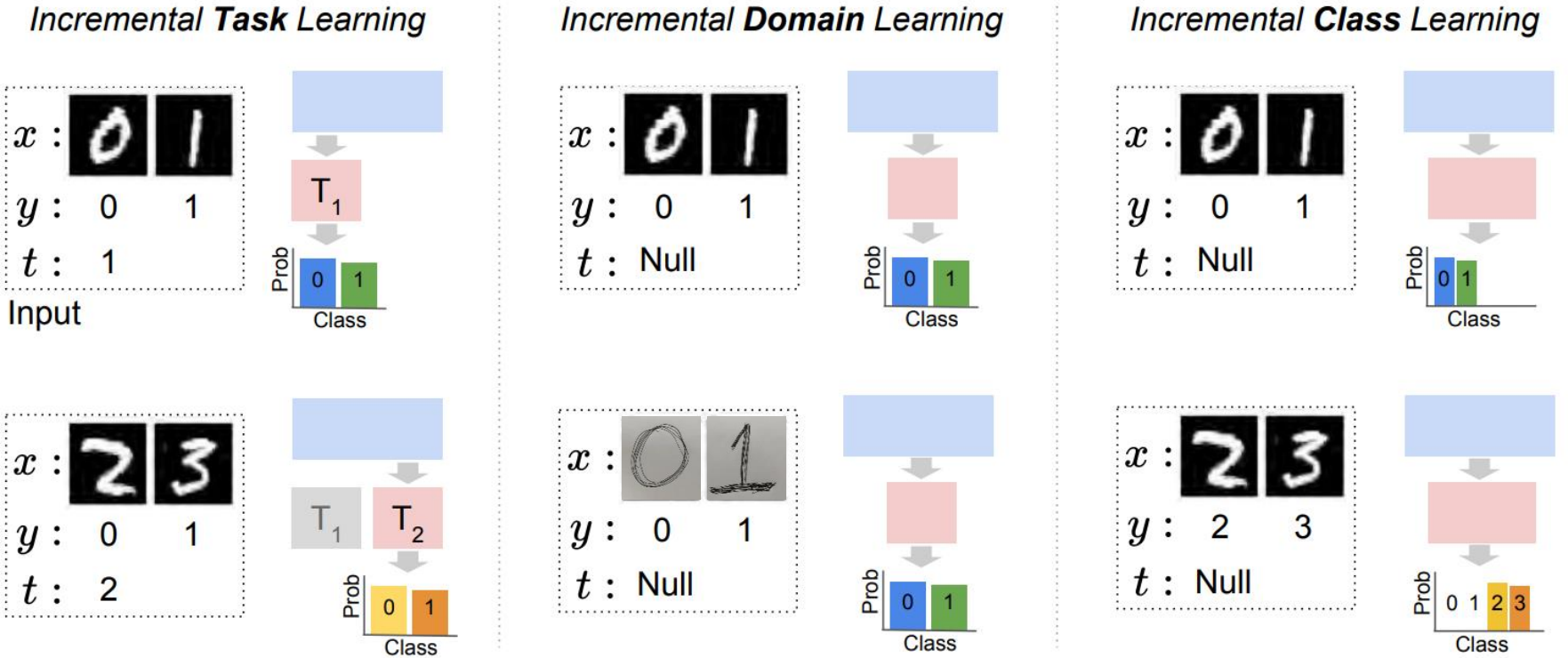
Background

Scenario	Difference between D_{i-1} and D_i $P(X_{i-1}) \neq P(X_i)$	$P(Y_{i-1}) \neq P(Y_i)$	$\{Y_{i-1}\} \neq \{Y_i\}$	Task-ID	Online
Task Incremental	✓	✓	✓	Train & Test	No
Class Incremental	✓	✓		No	Optional
Domain Incremental	✓			No	Optional

Task-Free CL

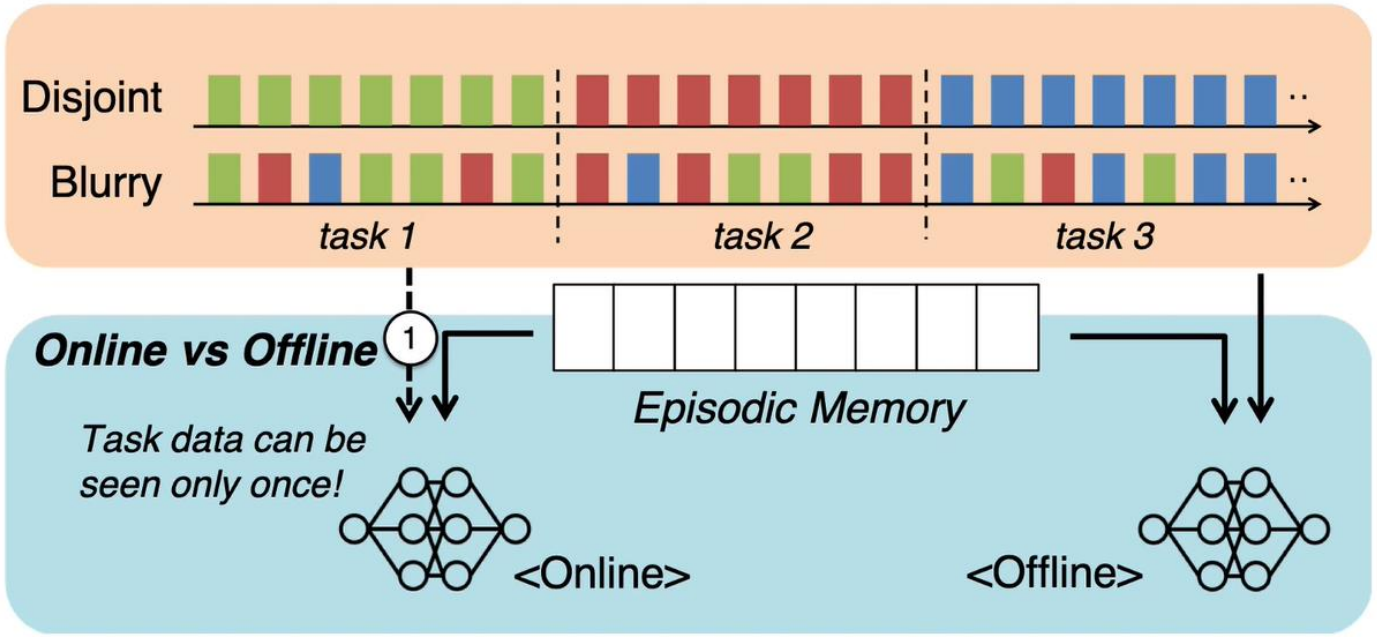
- Task-Incremental
- Domain-Incremental
- Class-Incremental
- Task-Agnostic CL

?



Setup for Continual Learning Methods

- **Online vs Offline**
  - **Online:** allow only **once** to see the incoming data **except** for examples in the memory.
  - **Offline**: allow to see the examples in current task and episodic memory **many times.**

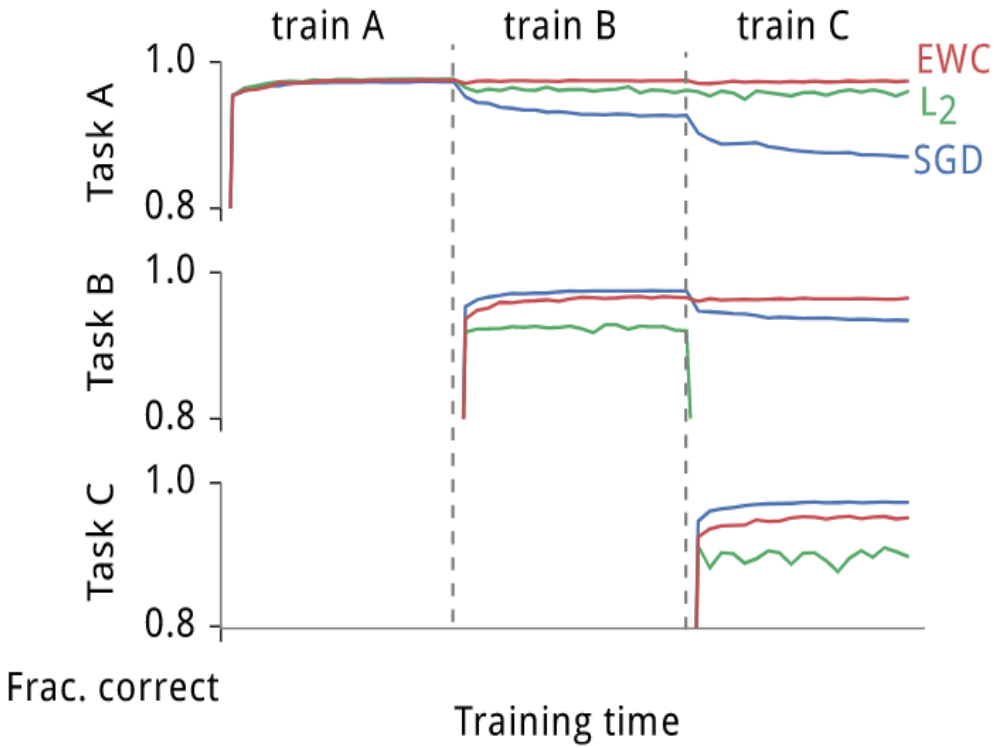


Algorithm 1 Purity and Diversity aware Episode Replay

Input: $\mathcal{S}_t$: stream data at task t, $\mathcal{M}$: exemplars stored in a episodic memory, T: the number of tasks, θ_0: initial model.
for $t = 1$ **to** T **do**
  for each mini-batch $\mathcal{B} \in \mathcal{S}_t$ **do**
    $\theta \leftarrow \theta - \alpha\nabla \sum_{i\in\mathcal{B}} \ell(\theta(x_i), \tilde{y}_i)$ ▷ online training
    for $x_i, \tilde{y}_i$ in $\mathcal{B}$ **do**
      Update $\mathcal{M}$ from $\mathcal{M} \cup (x_i, \tilde{y}_i)$ ▷ Sec. 4.1
  for $e = 1$ **to** MaxEpoch **do**
    Split $\mathcal{C}, \mathcal{R}, \mathcal{U}$ from $\mathcal{M}$ via Eqs. (5) and (7)
    for each mini-batch $\mathcal{B}_\mathcal{C} \in \mathcal{C}, \mathcal{B}_\mathcal{R} \in \mathcal{R}, \mathcal{B}_\mathcal{U} \in \mathcal{U}$ **do**
      $\theta \leftarrow \theta - \alpha\nabla[\ell_{cls} + \eta\ell_{reg}]$ ▷ Sec. 4.2

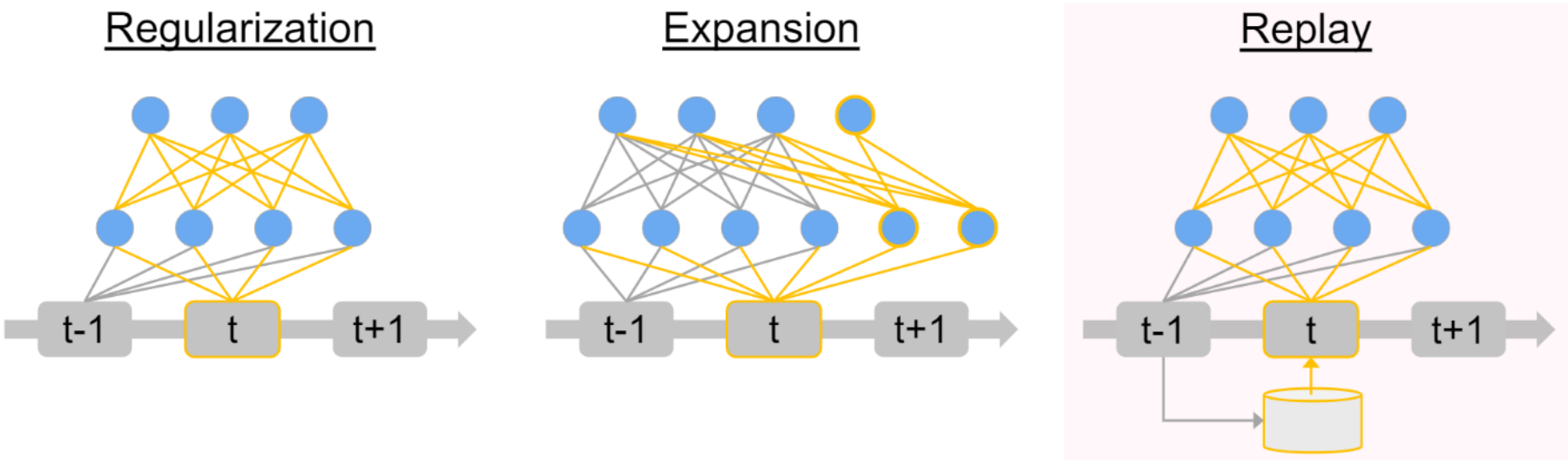
Main Problem of Continual Learning: Catastrophic Forgetting

Phenomenon that ***forgets*** the knowledge of **previous classes** when the model trains with new incoming classes.



Previous Works for Continual Learning Methods

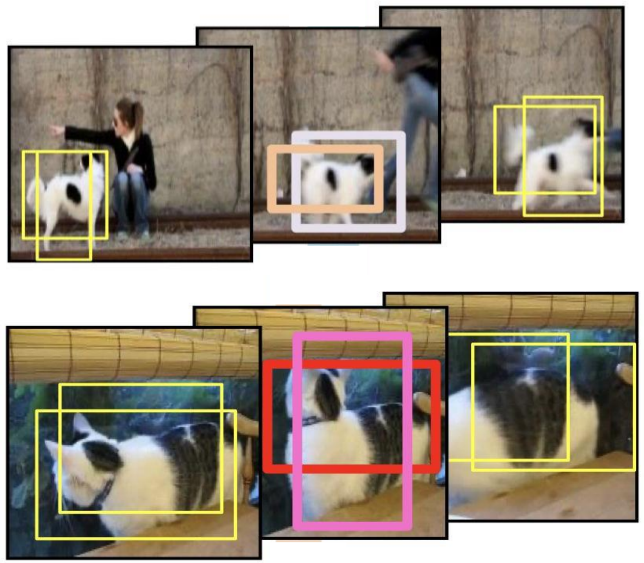
- Regularization based CL
  - **Trade-off:** keeping weights for maintaining previous knowledges vs. changing weights for learning new knowledges.
- Parameter isolation based CL
  - To prevent forgetting previous tasks, an intuition is to construct sufficiently large models, and for each task construct a subset of the larger model. This approach can be achieved by fixing Backbone and adding new branches for new tasks
- Rehearsal/Replay based CL
  - Assume that we can use small amount of data from previous tasks
  - **Key Point:** How to get informative data from the previous tasks?

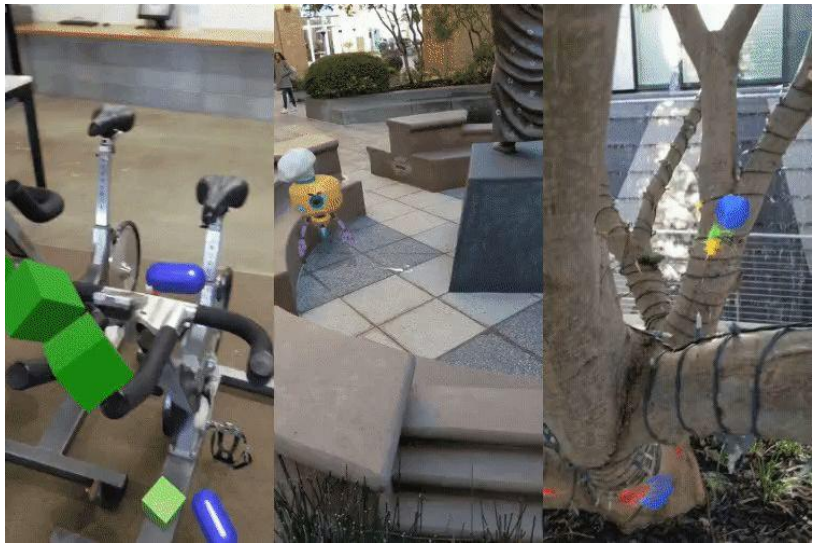


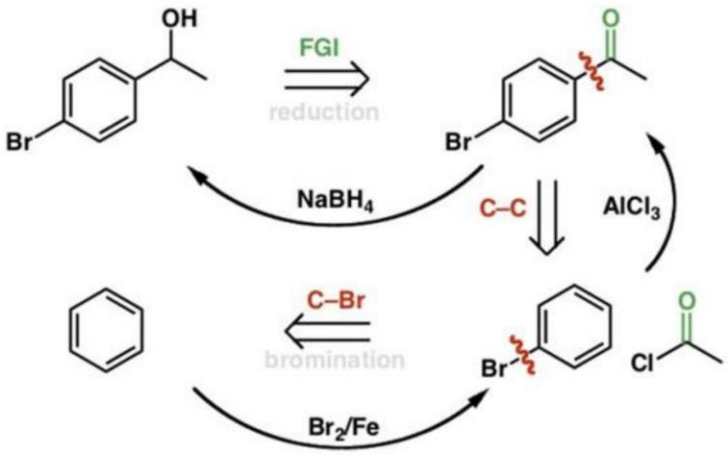
Motivation

Annotation Challenges

Annotation process is ***time-consuming*** (eg. bounding boxes in videos) and ***expensive*** as it requires expert knowledge in various applications.



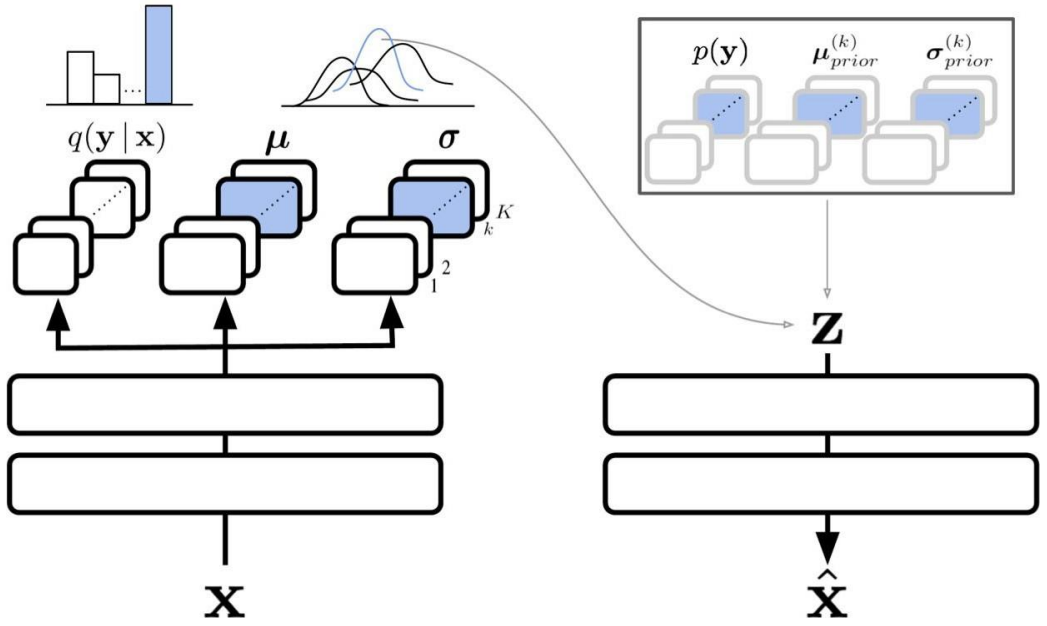




Therefore, it is essential to learn ***continual representations*** on ***unlabelled data streams***.

Previous Works

CURL learned ***task-specific representations*** on top of ***shared parameters*** using MLP encoders/decoders and a simple MoG generative replay.



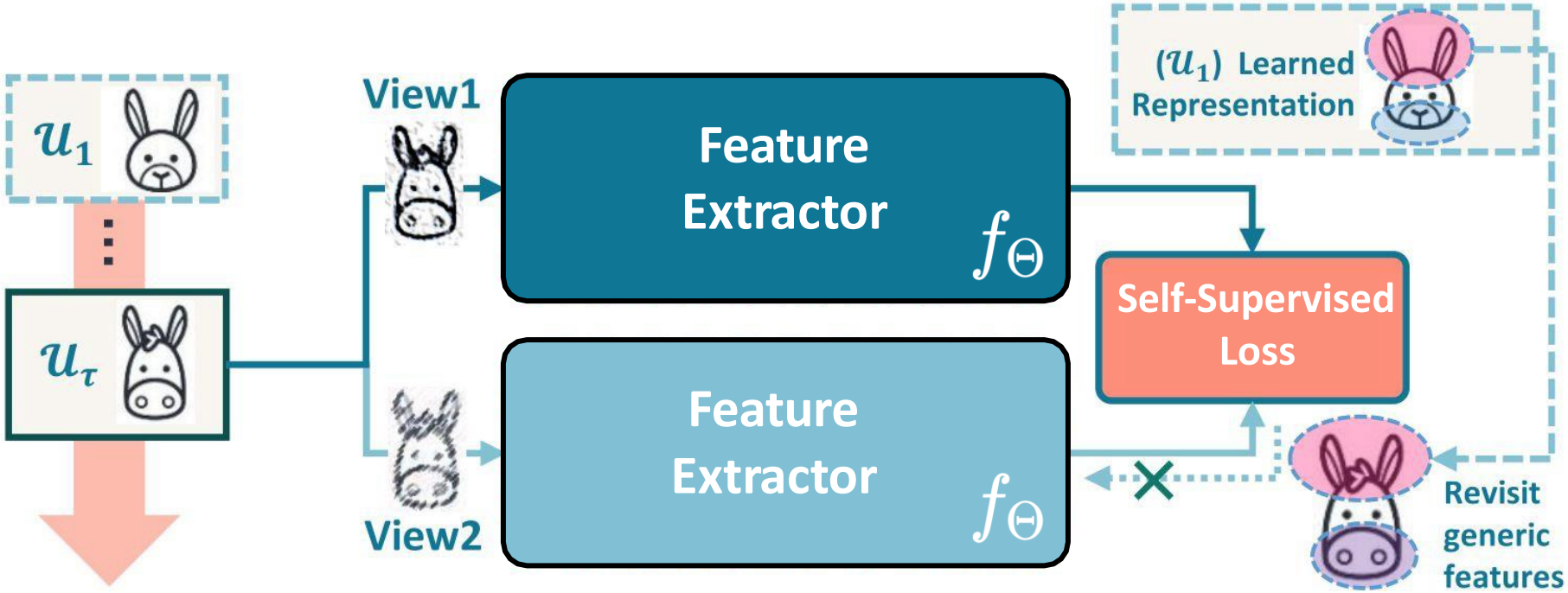
However, the method was restricted to ***simple low-resolution tasks*** and ***not scalable*** to standard CL benchmark datasets.

[Rao et al. 2019] Continual Unsupervised Representational Learning. NeurIPS 2019

Unsupervised Continual Learning (Offline)

learn the feature representations on an ***unlabelled sequence of tasks***.

use ***self-supervised learning*** to learn the feature representations.

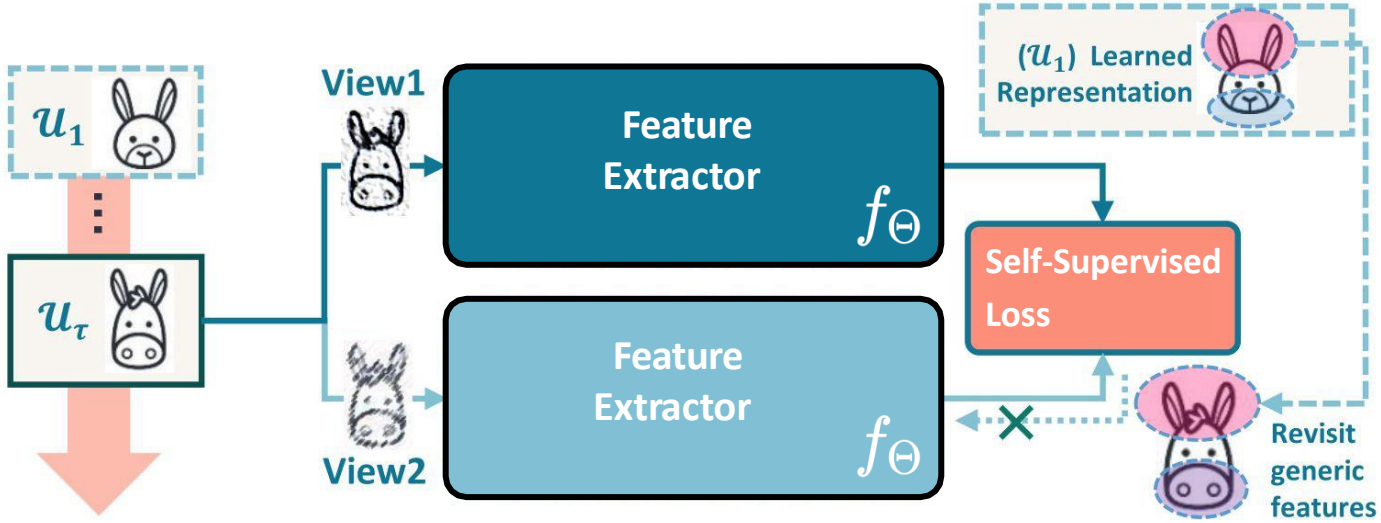


However, a ***trivial combination*** of self-supervised learning with a sequence of tasks can result in ***catastrophic forgetting***.

Unsupervised Continual Learning

To mitigate catastrophic forgetting, we ***revisit representations*** learnt on previous tasks.

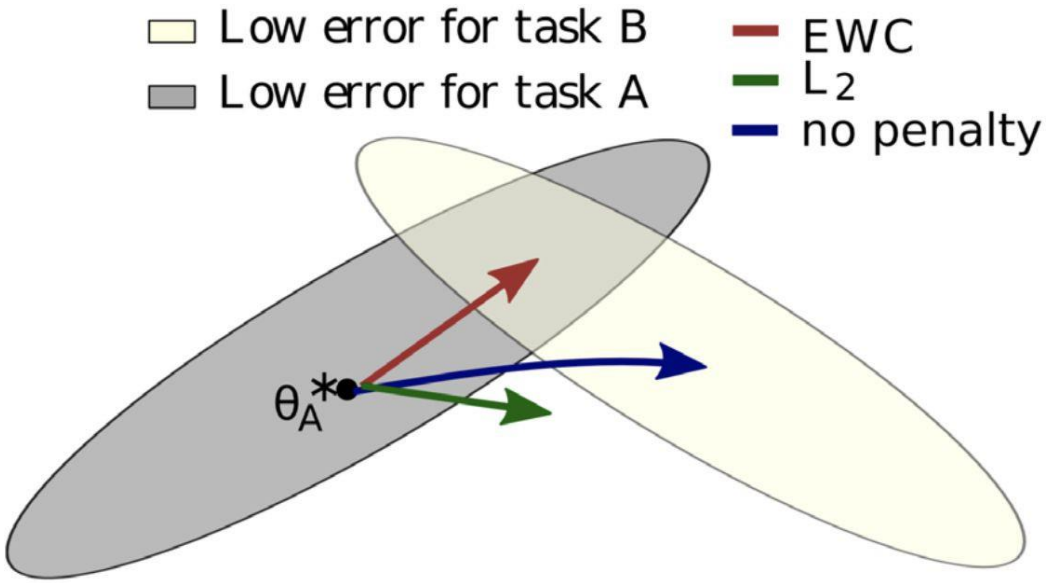
- **Structural Regularization**
- **Unsupervised Replay**
- **Architectural Expansion**
- **Lifelong Unsupervised Mixup**



Our quantitative and qualitative empirical analysis shows that ***reliance on annotated data*** is ***not necessary*** for continual learning.

Unsupervised Structural Regularization

Encourage the current task parameters to ***stay close*** to the parameters of the old task.



$$\mathcal{L}_{\text{UCL}}(\theta) = \mathcal{L}^{B}_{\text{UCL}}(\theta) + c\sum_i \Omega^{B}_i F_i(\theta_i - \theta_{A,i})^2$$

Task B Loss

Surrogate Loss

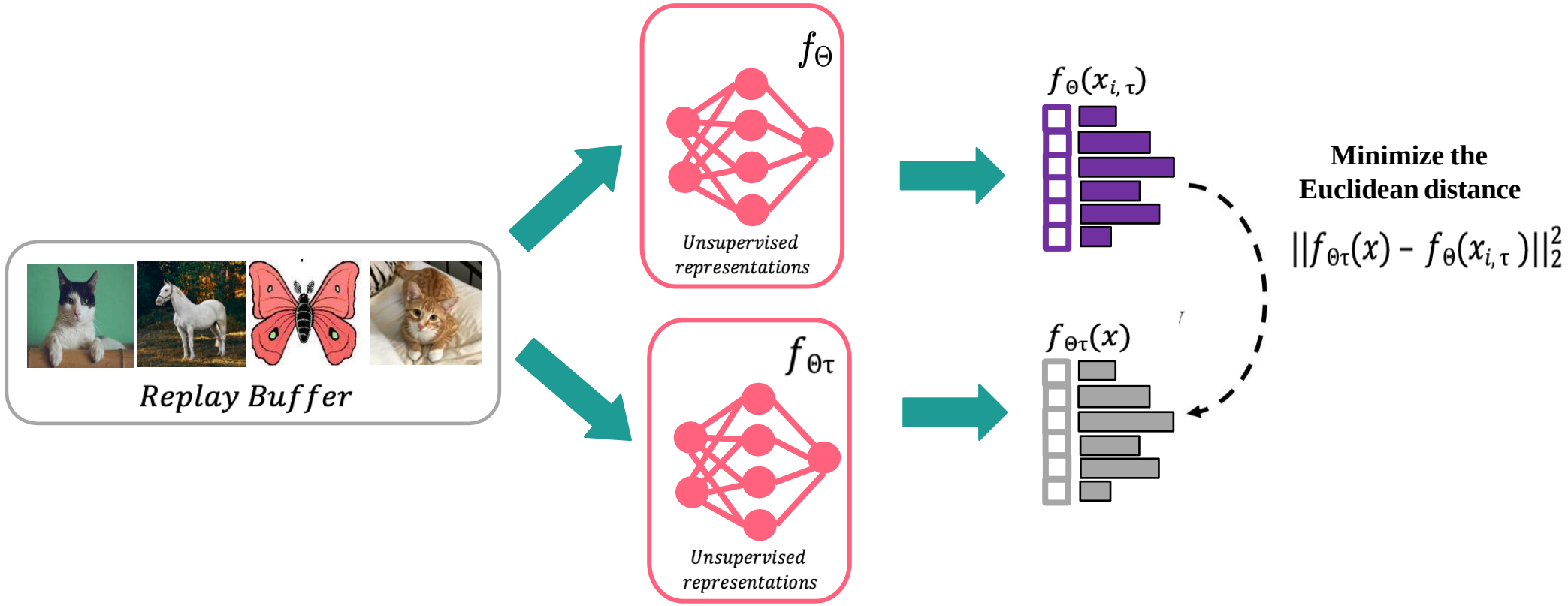
Synaptic Intelligence [Zenke et al. 2017]

$$\mathcal{L}^{B}_{\text{UCL}} = \frac{1}{2}D(p^1_{i,B}, \text{stopgrad}(z^2_{i,B})) + \frac{1}{2}D(p^2_{i,B}, \text{stopgrad}(z^1_{i,B}))$$

UCL loss with Simsiam for current task

Unsupervised Replay

We minimize the ***Euclidean distance*** between the ***projected outputs*** to evaluate DER (Buzzega et al. 2020) for Unsupervised Replay.

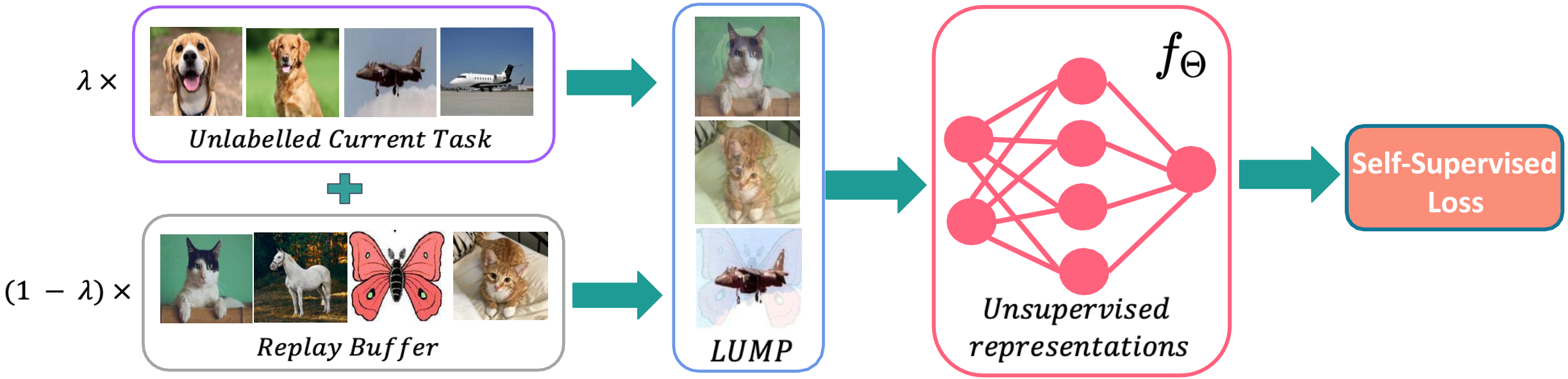


$$\mathcal{L}_{\text{UCL}}(\theta) = \mathcal{L}^{B}_{\text{UCL}}(\theta) + \alpha \cdot \mathbb{E}_{(x)\sim\mathcal{M}}\left[\|f_{\Theta_\tau}(x) - f_\Theta(x_{i,\tau})\|_2^2\right]$$

[Buzzega et al. 2020] Dark Experience for General Continual Learning: a Strong, Simple Baseline. NeurIPS 2020

Lifelong Unsupervised Mixup (LUMP)

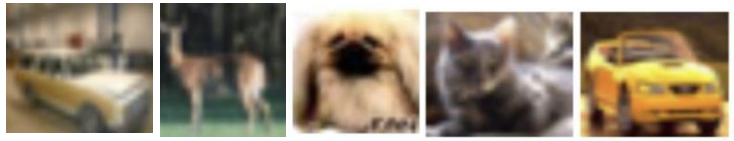
LUMP interpolates between the ***examples of the current task*** and random examples selected using uniform sampling ***from the replay buffer***.



Experiment Setup

Datasets

1) **Split CIFAR-10 [Krizhevsky 2012]** A dataset with 60,000 images composed of ***five tasks*** from ***ten animal and vehicle classes***.



2) **Split CIFAR-100 [Krizhevsky 2012]** A dataset with 60,000 images composed of ***20 tasks*** from ***100 generic object classes***.

3) **Split Tiny-ImageNet [Russakovsky 2015]** A ***subset of ImageNet*** dataset. We construct ***20 tasks*** using ***100 classes***.

Metrics

1) **Accuracy** is the average test accuracy of all the tasks completed until the continual learning of task $\mathcal{T}$

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

2) **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 $\mathcal{T}_\tau$ using a KNN on frozen pre-trained representations on task $\mathcal{T}_\tau$

[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

Continual Learning Evaluation

Unsupervised CL methods ***improve accuracy*** with LUMP ***outperforming*** all methods.

Table 1: **Accuracy and forgetting** of the learnt representations on Split CIFAR-10, Split CIFAR-100 and Split Tiny-ImageNet on Resnet-18 architecture with KNN classifier (Wu et al., 2018). All the values are measured by computing mean and standard deviation across three trials. The best and second-best results are highlighted in **bold** and underline respectively.

	METHOD	SPLIT CIFAR-10		SPLIT CIFAR-100		SPLIT TINY-IMAGENET	
		ACCURACY	FORGETTING	ACCURACY	FORGETTING	ACCURACY	FORGETTING
	SUPERVISED CONTINUAL LEARNING						
	FINETUNE	82.87 (± 0.47)	14.26 (± 0.52)	61.08 (± 0.04)	31.23 (± 0.41)	53.10 (± 1.37)	33.15 (± 1.22)
	PNN (Rusu et al., 2016)	82.74 (± 2.12)	–	66.05 (± 0.86)	–	64.38 (± 0.92)	–
	SI (Zenke et al., 2017)	85.18 (± 0.65)	11.39 (± 0.77)	63.58 (± 0.37)	27.98 (± 0.34)	44.96 (± 2.41)	26.29 (± 1.40)
	A-GEM (Chaudhry et al., 2019a)	82.41 (± 1.24)	13.82 (± 1.27)	59.81 (± 1.07)	30.08 (± 0.91)	60.45 (± 0.24)	24.94 (± 1.24)
	GSS (Aljundi et al., 2019)	89.49 (± 1.75)	7.50 (± 1.52)	70.78 (± 1.67)	21.28 (± 1.52)	70.96 (± 0.72)	14.76 (± 1.22)
	DER (Buzzega et al., 2020)	91.35 (± 0.46)	5.65 (± 0.35)	79.52 (± 1.88)	12.80 (± 1.47)	68.03 (± 0.85)	17.74 (± 0.65)
	MULTITASK	97.77 (± 0.15)	–	93.89 (± 0.78)	–	91.79 (± 0.46)	–
	UNSUPERVISED CONTINUAL LEARNING						
SIMSIAM	FINETUNE	90.11 (± 0.12)	5.42 (± 0.08)	75.42 (± 0.78)	10.19 (± 0.37)	71.07 (± 0.20)	9.48 (± 0.56)
SIMSIAM	PNN (Rusu et al., 2016)	90.93 (± 0.22)	–	66.58 (± 1.00)	–	62.15 (± 1.35)	–
SIMSIAM	SI (Zenke et al., 2017)	**92.75** (± **0.06**)	1.81 (± 0.21)	80.08 (± 1.30)	5.54 (± 1.30)	72.34 (± 0.42)	8.26 (± 0.64)
SIMSIAM	DER (Buzzega et al., 2020)	91.22 (± 0.30)	4.63 (± 0.26)	77.27 (± 0.30)	9.31 (± 0.09)	71.90 (± 1.44)	8.36 (± 2.06)
SIMSIAM	LUMP	91.00 (± 0.40)	2.92 (± 0.53)	**82.30** (± **1.35**)	4.71 (± 1.52)	**76.66** (± **2.39**)	3.54 (± 1.04)
	MULTITASK	95.76 (± 0.08)	–	86.31 (± 0.38)	–	82.89 (± 0.49)	–
BARLOWTWINS	FINETUNE	87.72 (± 0.32)	4.08 (± 0.56)	71.97 (± 0.54)	9.45 (± 1.01)	66.28 (± 1.23)	8.89 (± 0.66)
BARLOWTWINS	PNN (Rusu et al., 2016)	87.52 (± 0.33)	–	57.93 (± 2.98)	–	48.70 (± 2.59)	–
BARLOWTWINS	SI (Zenke et al., 2017)	90.21 (± 0.08)	2.03 (± 0.22)	75.04 (± 0.63)	7.43 (± 0.67)	56.96 (± 1.48)	17.04 (± 0.89)
BARLOWTWINS	DER (Buzzega et al., 2020)	88.67 (± 0.24)	2.41 (± 0.26)	73.48 (± 0.53)	7.98 (± 0.29)	68.56 (± 1.47)	7.87 (± 0.44)
BARLOWTWINS	LUMP	90.31 (± 0.30)	**1.13** (± **0.18**)	80.24 (± 1.04)	**3.53** (± **0.83**)	72.17 (± 0.89)	**2.43** (± **1.00**)
	MULTITASK	95.48 (± 0.14)	–	87.16 (± 0.52)	–	82.42 (± 0.74)	–

Out of Distribution Evaluation

Evaluation on various OOD datasets also show ***consistent improvements***.

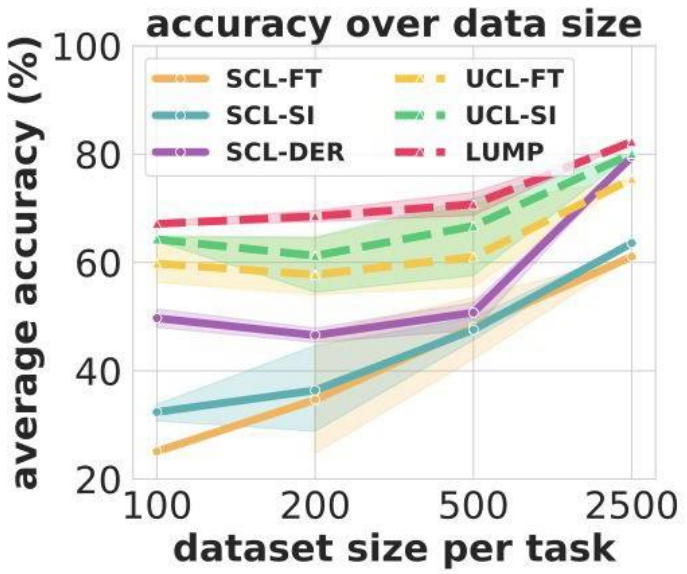
Table 2: **Comparison of accuracy** on out of distribution datasets using a KNN classifier (Wu et al., 2018) on pre-trained SCL and UCL representations. We consider MNIST (LeCun, 1998), Fashion-MNIST (FMNIST) (Xiao et al., 2017), SVHN (Netzer et al., 2011) as out of distribution for Split CIFAR-100 and Split CIFAR-10. All the values are measured by computing mean and standard deviation across three trials. The best and second-best results are highlighted in **bold** and underline respectively.

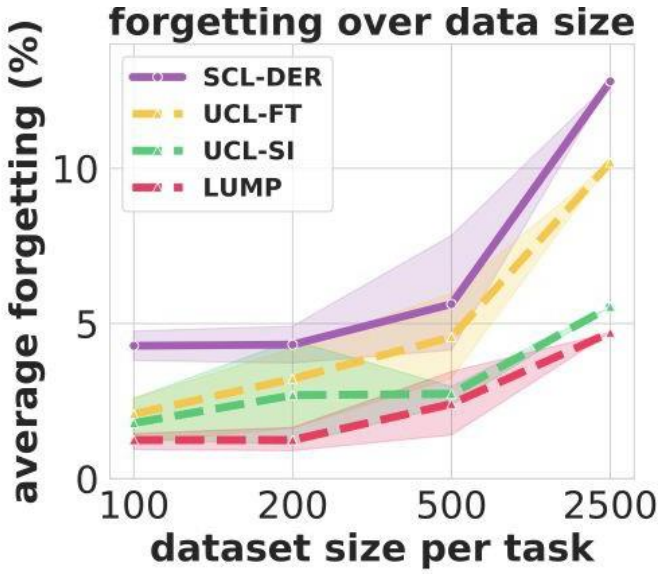
	In-class	Split CIFAR-10				Split CIFAR-100			
	Out-of-class	MNIST	FMNIST	SVHN	CIFAR-100	MNIST	FMNIST	SVHN	CIFAR-10
	Supervised Continual Learning								
	Finetune	86.42 (± 1.11)	74.47 (± 0.84)	41.00 (± 0.85)	17.42 (± 0.96)	75.02 (± 3.97)	62.37 (± 3.20)	38.05 (± 0.73)	39.18 (± 0.83)
	SI (Zenke et al., 2017)	87.08 (± 0.79)	76.41 (± 0.81)	42.62 (± 1.31)	19.14 (± 0.91)	79.96 (± 2.63)	63.71 (± 1.36)	40.92 (± 1.64)	40.41 (± 1.71)
	A-GEM (Chaudhry et al., 2019a)	86.07 (± 1.94)	74.74 (± 3.21)	37.77 (± 3.49)	16.11 (± 0.38)	77.56 (± 3.21)	64.16 (± 2.29)	37.48 (± 1.73)	37.91 (± 1.33)
	GSS (Aljundi et al., 2019)	70.36 (± 3.54)	69.20 (± 2.51)	33.11 (± 2.26)	18.21 (± 0.39)	76.54 (± 0.46)	65.31 (± 1.72)	35.72 (± 2.37)	49.41 (± 1.81)
	DER (Buzzega et al., 2020)	80.32 (± 1.91)	70.49 (± 1.54)	41.48 (± 2.76)	17.72 (± 0.25)	87.71 (± 2.23)	75.97 (± 1.29)	50.26 (± 0.95)	59.07 (± 1.06)
	MULTITASK	88.79 (± 1.13)	79.50 (± 0.52)	41.26 (± 1.95)	27.68 (± 0.66)	92.29 (± 3.37)	86.12 (± 1.87)	54.94 (± 1.77)	54.04 (± 3.68)
	Unsupervised Continual Learning								
SimSiam	Finetune	89.23 (± 0.99)	80.05 (± 0.34)	49.66 (± 0.81)	34.52 (± 0.12)	85.99 (± 0.86)	76.90 (± 0.11)	50.09 (± **1.41**)	57.15 (± 0.96)
SimSiam	SI (Zenke et al., 2017)	**93.72** (± **0.58**)	**82.50** (± **0.51**)	**57.88** (± **0.16**)	**36.21** (± **0.69**)	91.50 (± 1.26)	80.57 (± 0.93)	**54.07** (± **2.73**)	60.55 (± **2.54**)
SimSiam	DER (Buzzega et al., 2020)	88.35 (± 0.82)	79.33 (± 0.62)	48.83 (± 0.55))	30.68 (± 0.36)	87.96 (± 2.04)	76.21 (± 0.63)	47.70 (± 0.94)	56.26 (± 0.16)
SimSiam	LUMP	91.03 (± 0.22)	80.78 (± 0.88)	45.18 (± 1.57)	31.17 (± 1.83)	**91.76** (± **1.17**)	**81.61** (± **0.45**)	50.13 (± 0.71)	**63.00** (± **0.53**)
SimSiam	MULTITASK	90.69 (± 0.13)	80.65 (± 0.42)	47.67 (± 0.45)	39.55 (± 0.18)	90.35 (± 0.24)	81.11 (± 1.86)	52.20 (± 0.61)	70.19 (± 0.15)
BarlowTwins	Finetune	86.86 (± 1.62)	78.37 (± 0.74)	44.64 (± 2.39)	28.03 (± 0.52)	76.08 (± 2.86)	76.82 (± 0.83)	42.95 (± 0.90)	53.12 (± 0.13)
BarlowTwins	SI (Zenke et al., 2017)	90.31 (± 0.69)	80.58 (± 0.68)	49.18 (± 0.51)	31.80 (± 0.4)	85.24 (± 0.99)	78.82 (± 0.67)	45.18 (± 1.37)	53.99 (± 0.56)
BarlowTwins	DER (Buzzega et al., 2020)	85.15 (± 2.19)	77.96 (± 0.59)	45.68 (± 0.93)	27.83 (± 0.86)	78.08 (± 1.95)	76.67 (± 0.68)	44.58 (± 1.01)	53.24 (± 0.82)
BarlowTwins	LUMP	88.73 (± 0.54)	81.69 (± 0.45)	51.53 (± 0.41)	31.53 (± 0.36)	90.22 (± 1.39)	81.28 (± 0.91)	50.24 (± 0.95)	60.76 (± 0.87)
BarlowTwins	MULTITASK	88.63 (± 1.38)	79.49 (± 0.29)	49.24 (± 2.44)	36.33 (± 0.29)	86.98 (± 1.70)	79.40 (± 1.10)	50.19 (± 0.81)	49.50 (± 0.38)

Evaluation of representations ***trained*** with ***Sequential CIFAR-100*** on OOD datasets using KNN classifier.

Few-Shot Training Evaluation

Next, we evaluate on a ***limited number*** of training instances, where UCL ***improves accuracy and mitigate forgetting*** in comparison to SCL.



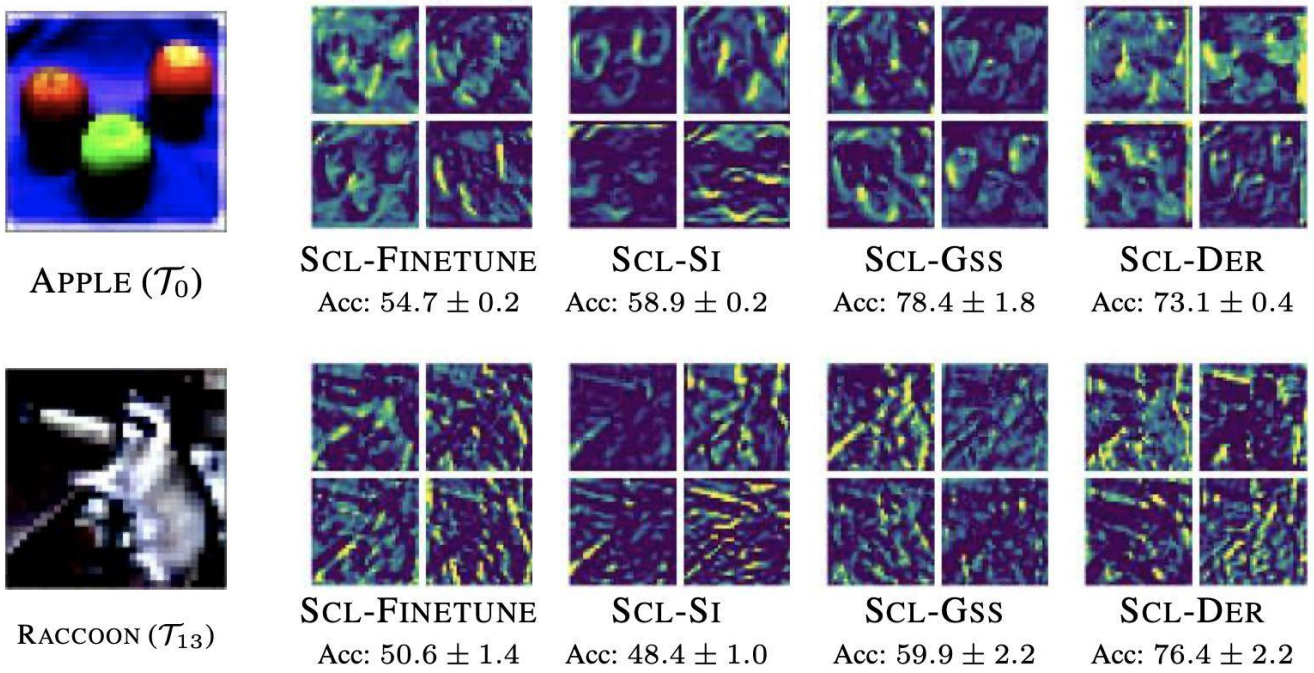


Few Shot Training with ***Split CIFAR-100***

This is an outcome of the ***discriminative feature embeddings*** learned by UCL.

Visualization of Feature Space

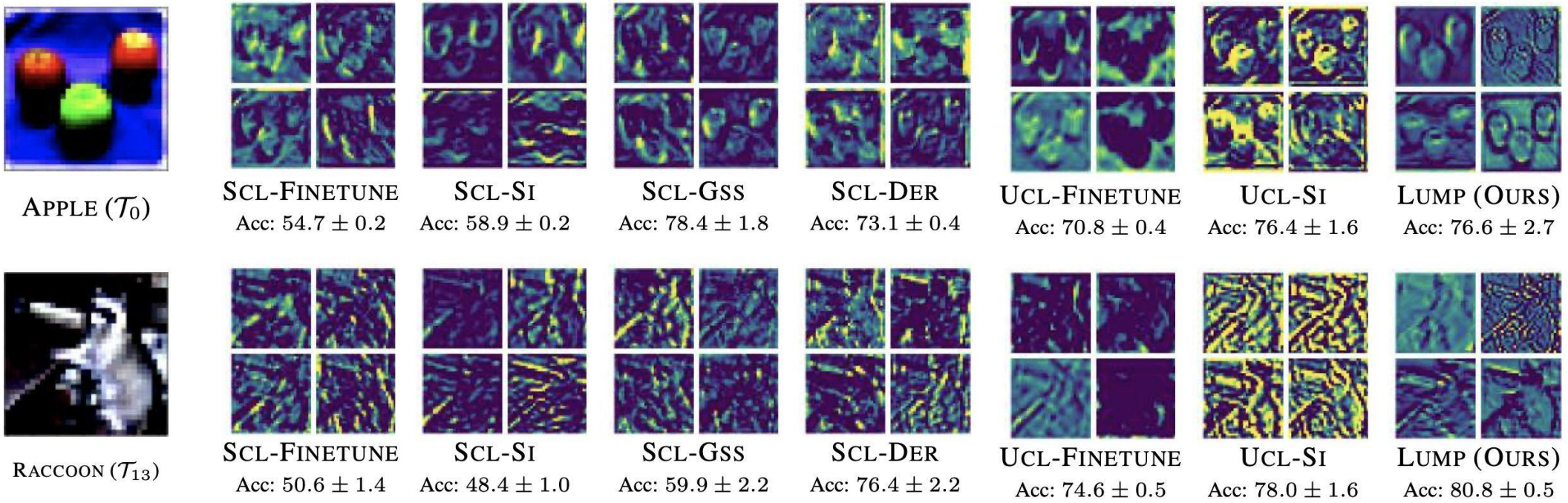
SCL is prone to catastrophic forgetting, as the features are ***noisy w/o coherent patterns***.



This is because SCL is ***prone to forgetting***, which hurts the past tasks representations.

Visualization of Feature Space

UCL features are ***more relevant***, with LUMP learning the most ***distinctive features***.



We believe this is because UCL captures ***more than*** class-specific features, and captures generic information ***independent*** of the class labels.

Visualization of Loss Landscape

UCL also obtaining a ***flatter and smoother loss landscape*** compared to SCL.

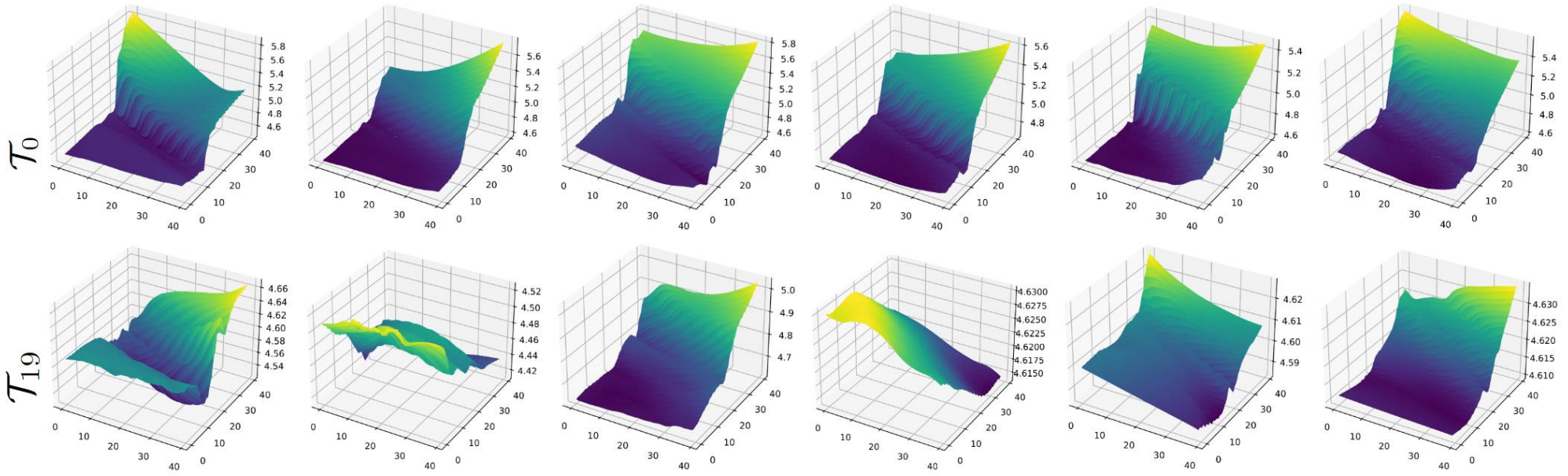


Figure 5: **Loss landscape visualization** of $\mathcal{T}_0$ after the completion of training on task $\mathcal{T}_0$ (**top**) and $\mathcal{T}_{19}$ (**bottom**) for Split CIFAR-100 dataset on ResNet-18 architecture. We use Simsiam for UCL methods.

It indicates that UCL is ***stable and robust*** to the forgetting.

Thanks