Adversarial Continual Learning

Sayna Ebrahimi[1,2], Franziska Meier[1], Roberto Calandra[1], Trevor Darrell[2], and Marcus Rohrbach[1]

[1] Facebook AI Research
[2] UC Berkeley

Organized by: Xi Fang
Date:06/10/2020

Continual learning

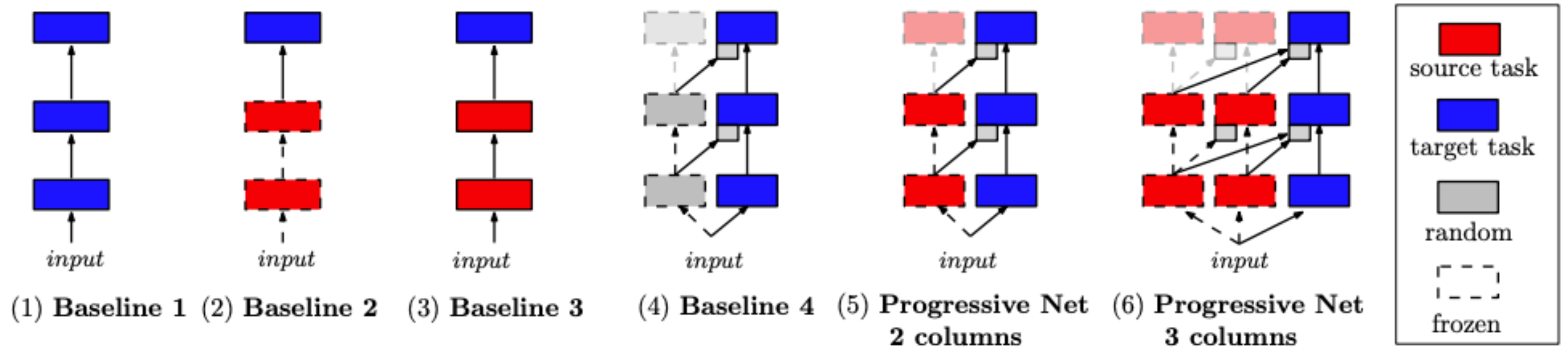
- Lifelong learning
  - learn new tasks without forgetting previously learned ones
  - solve each task in a sequence have a shared structure while containing some task-specific properties
  - novel hybrid continual learning framework that learns a disjoint representation for task-invariant and task-specific features

Memory based approach

- Generative model
  - VAE, GAN
  - synthesizes them to perform pseudo-rehearsal.
  - Allow for simultaneous multi-task learning

Architecture-based approaches

- Exploit modularity and attempt to localize inference to a subset of the network such as columns
    - Progressive neural network



Regularization methods

- The change in parameters is controlled and reduced or prevented if it causes performance downgrade on prior tasks.
- Preserve important parameters
  - Elastic weight consolidation
  - HAT

Approach

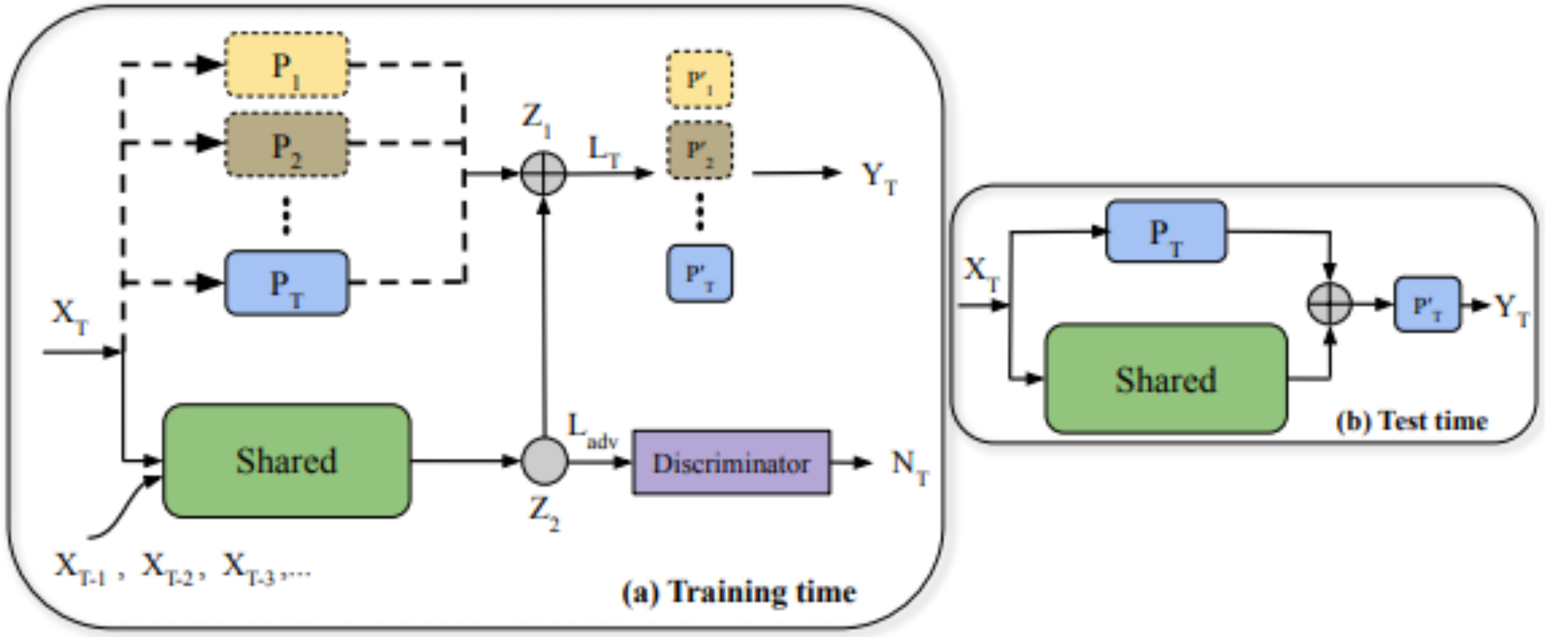
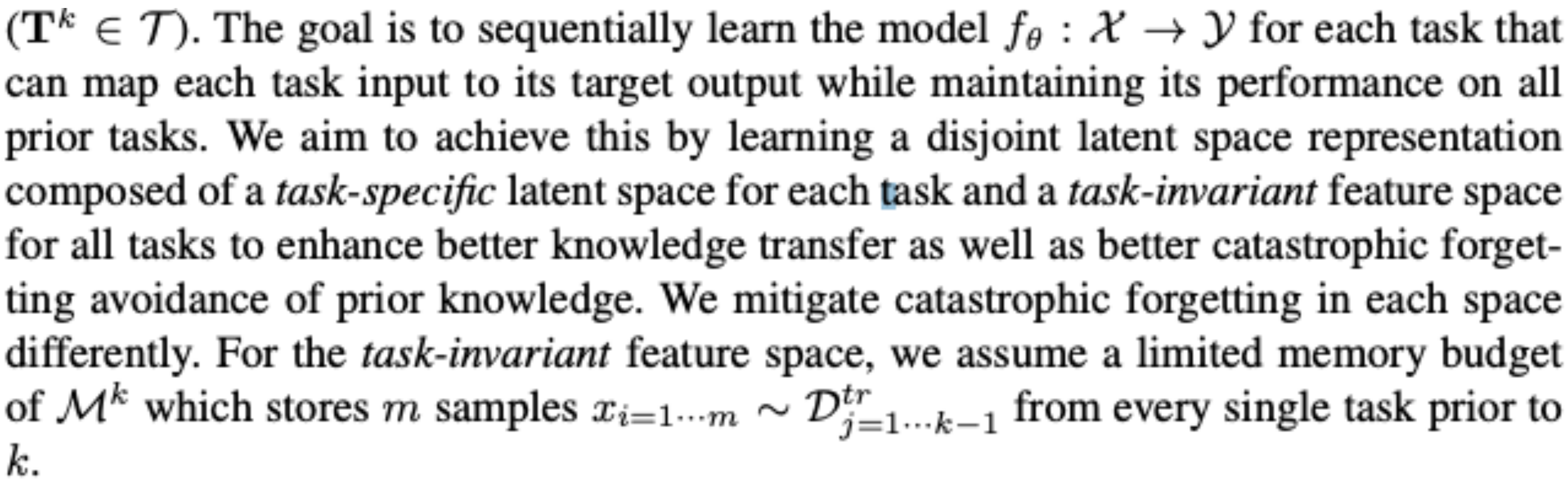


Fig. 1: Factorizing task-specific and task-invariant features in our method (ACL). *Left:* Shows ACL at training time where the Shared module is adversarially trained with the discriminator to generate *task-invariant* features ($\mathbf{z}_S$) while the discriminator attempts to predict task labels. Architecture growth occurs at the arrival of each task by adding a *task-specific* module optimized to generate orthogonal representation ($\mathbf{z}_P$) to $\mathbf{z}_S$. To prevent forgetting, 1) Private modules are stored for each task and 2) A shared module which is less prone to forgetting, yet is also retrained with experience reply with a limited number of exemplars *Right:* At test time, discriminator is removed and ACL uses the P module for the specific task it is evaluated on.

Latent Space Factorization

- Multi-view learning
  - Aim at either maximizing the mutual agreement on distinct views of the data, or focus on obtaining a latent subspace shared by multiple views
  - Factorizing the latent space into shared and private parts has been extensively explored for different data modalities

Adversarial continual learning (ACL)



($\mathbf{T}^k \in \mathcal{T}$). The goal is to sequentially learn the model $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ for each task that can map each task input to its target output while maintaining its performance on all prior tasks. We aim to achieve this by learning a disjoint latent space representation composed of a *task-specific* latent space for each task and a *task-invariant* feature space for all tasks to enhance better knowledge transfer as well as better catastrophic forgetting avoidance of prior knowledge. We mitigate catastrophic forgetting in each space differently. For the *task-invariant* feature space, we assume a limited memory budget of $\mathcal{M}^k$ which stores m samples $x_{i=1\cdots m} \sim \mathcal{D}^{tr}_{j=1\cdots k-1}$ from every single task prior to k.

Adversarial continual learning (ACL)

$$\mathcal{L}_{\text{task}}(f_\theta^k, \mathbf{X}^k, \mathbf{Y}^k, \mathcal{M}^k) = -\mathbb{E}_{(x^k, y^k) \sim (\mathbf{X}^k, \mathbf{Y}^k) \cup \mathcal{M}^k} \sum_{c=1}^{C} \mathbb{1}_{[c=y^k]} \log(\sigma(f_\theta^k(x^k))) \quad (1)$$

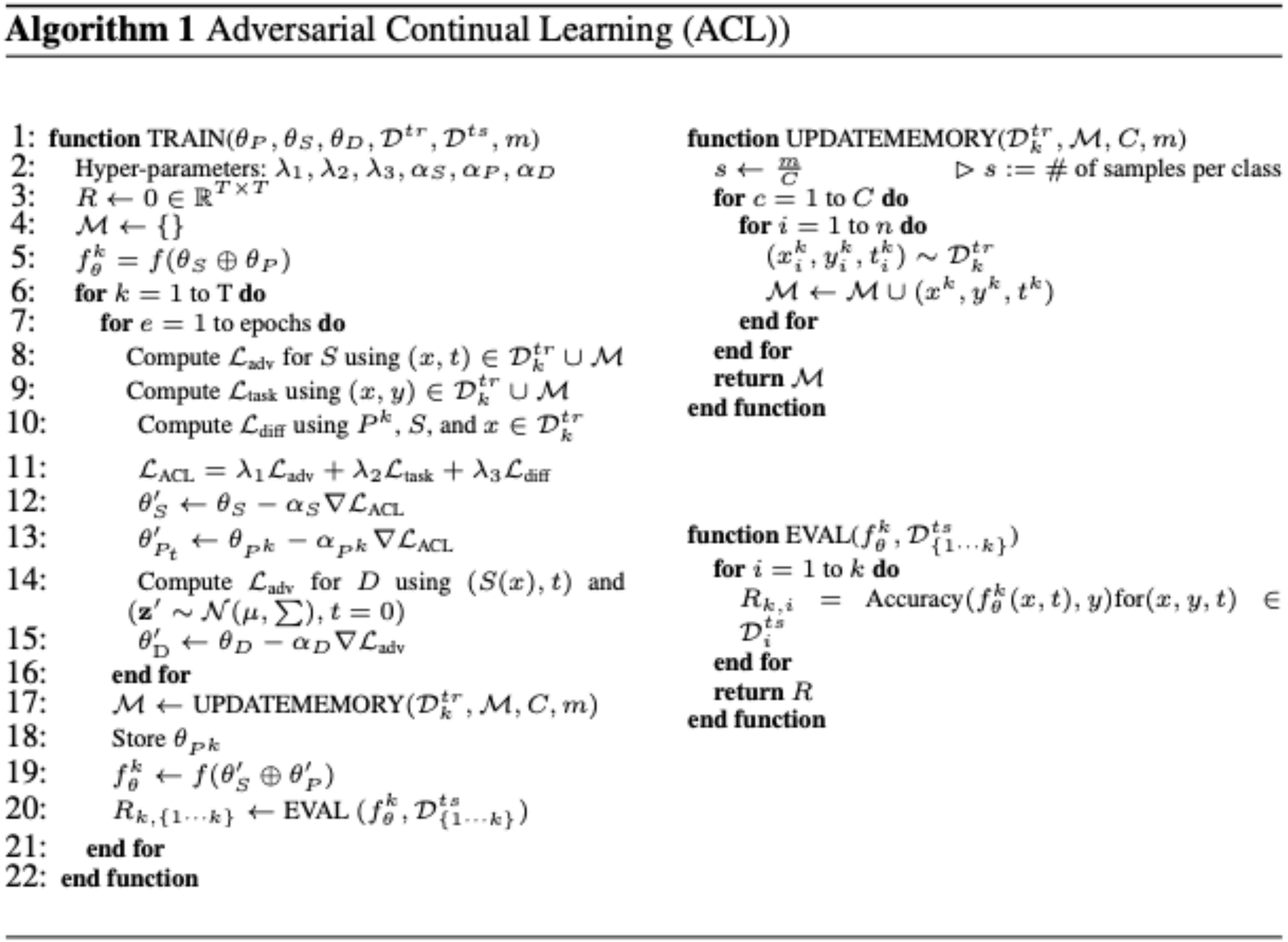
corresponds to the following T-way classification cross-entropy adversarial loss for this minimax game

$$\mathcal{L}_{\text{adv}}(D, S, \mathbf{X}^k, \mathbf{T}^k, \mathcal{M}^k) = \min_S \max_D \sum_{k=0}^{T} \mathbb{1}_{[k=t^k]} \log\left(D\left(S\left(x^k\right)\right)\right) . \quad (2)$$

$$\mathcal{L}_{\text{diff}}(S, P, \mathbf{X}^k, \mathcal{M}^k) = \sum^{T} ||(S(x^k))^{\mathrm{T}} P^k(x^k)||_F^2 , \quad (3)$$

Taken together, these loss form the complete objective for ACL as

$$\mathcal{L}_{\text{ACL}} = \lambda_1 \mathcal{L}_{\text{adv}} + \lambda_2 \mathcal{L}_{\text{task}} + \lambda_3 \mathcal{L}_{\text{diff}} , \quad (4)$$



Algorithm 1 Adversarial Continual Learning (ACL))

```
1: function TRAIN($\theta_P, \theta_S, \theta_D, \mathcal{D}^{tr}, \mathcal{D}^{ts}, m$)
2:     Hyper-parameters: $\lambda_1, \lambda_2, \lambda_3, \alpha_S, \alpha_P, \alpha_D$
3:     $R \leftarrow 0 \in \mathbb{R}^{T \times T}$
4:     $\mathcal{M} \leftarrow \{\}$
5:     $f_\theta^k = f(\theta_S \oplus \theta_P)$
6:     for $k = 1$ to T do
7:         for $e = 1$ to epochs do
8:             Compute $\mathcal{L}_{\text{adv}}$ for $S$ using $(x, t) \in \mathcal{D}_k^{tr} \cup \mathcal{M}$
9:             Compute $\mathcal{L}_{\text{task}}$ using $(x, y) \in \mathcal{D}_k^{tr} \cup \mathcal{M}$
10:            Compute $\mathcal{L}_{\text{diff}}$ using $P^k$, $S$, and $x \in \mathcal{D}_k^{tr}$
11:            $\mathcal{L}_{\text{ACL}} = \lambda_1 \mathcal{L}_{\text{adv}} + \lambda_2 \mathcal{L}_{\text{task}} + \lambda_3 \mathcal{L}_{\text{diff}}$
12:            $\theta'_S \leftarrow \theta_S - \alpha_S \nabla \mathcal{L}_{\text{ACL}}$
13:            $\theta'_{P_t} \leftarrow \theta_{P^k} - \alpha_{P^k} \nabla \mathcal{L}_{\text{ACL}}$
14:            Compute $\mathcal{L}_{\text{adv}}$ for $D$ using $(S(x), t)$ and
               $(\mathbf{z}' \sim \mathcal{N}(\mu, \sum), t = 0)$
15:            $\theta'_D \leftarrow \theta_D - \alpha_D \nabla \mathcal{L}_{\text{adv}}$
16:        end for
17:        $\mathcal{M} \leftarrow$ UPDATEMEMORY($\mathcal{D}_k^{tr}, \mathcal{M}, C, m$)
18:        Store $\theta_{P^k}$
19:        $f_\theta^k \leftarrow f(\theta'_S \oplus \theta'_P)$
20:        $R_{k,\{1 \cdots k\}} \leftarrow$ EVAL ($f_\theta^k, \mathcal{D}^{ts}_{\{1 \cdots k\}}$)
21:    end for
22: end function

function UPDATEMEMORY($\mathcal{D}_k^{tr}, \mathcal{M}, C, m$)
    $s \leftarrow \frac{m}{C}$                ▷ $s$ := # of samples per class
    for $c = 1$ to $C$ do
        for $i = 1$ to $n$ do
            $(x_i^k, y_i^k, t_i^k) \sim \mathcal{D}_k^{tr}$
            $\mathcal{M} \leftarrow \mathcal{M} \cup (x^k, y^k, t^k)$
        end for
    end for
    return $\mathcal{M}$
end function

function EVAL($f_\theta^k, \mathcal{D}^{ts}_{\{1 \cdots k\}}$)
    for $i = 1$ to $k$ do
        $R_{k,i}$ = Accuracy($f_\theta^k(x, t), y$)for$(x, y, t) \in \mathcal{D}_i^{ts}$
    end for
    return $R$
end function
```

Experiments

- Datasets:
  - 5-Split MNIST and Permuted MNIST, 20-Split CIFAR100, 20-Split miniImageNet
  - a sequence of 5-Datasets including SVHN, CIFAR10, not-MNIST, Fashion-MNIST and, MNIST
- Baselines:
  - Elastic Weight Consolidation (EWC) [17], Progressive neural networks (PNNs) [28], and Hard Attention Mask (HAT)
- Criterion:

$$\mathrm{ACC} = \frac{1}{T}\sum_{i=1}^{T} R_{T,i}, \quad \mathrm{BWT} = \frac{1}{T-1}\sum_{i=1}^{T-1} R_{T,i} - R_{i,i}$$

Results

- ACL on 20-Split miniImageNet

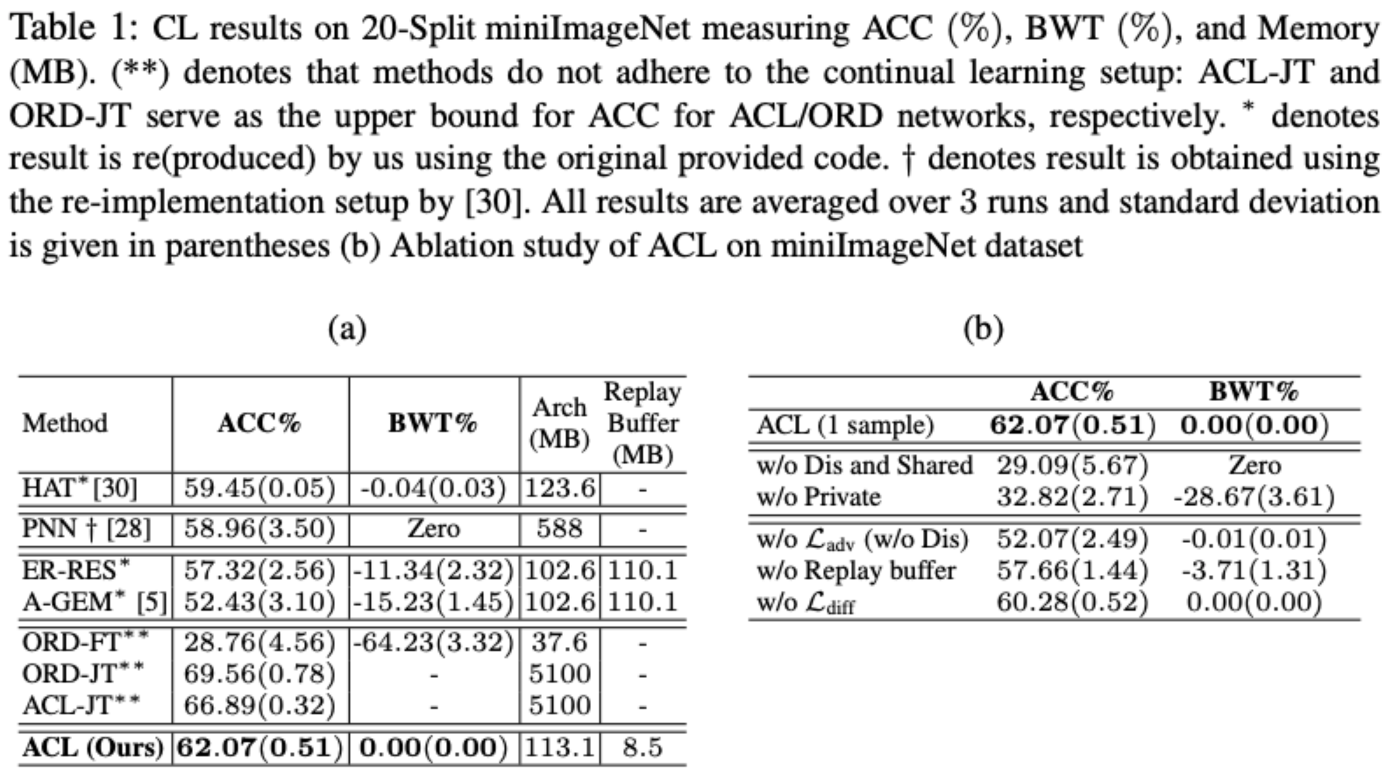


Table 1: CL results on 20-Split miniImageNet measuring ACC (%), BWT (%), and Memory (MB). (**) denotes that methods do not adhere to the continual learning setup: ACL-JT and ORD-JT serve as the upper bound for ACC for ACL/ORD networks, respectively. * denotes result is re(produced) by us using the original provided code. † denotes result is obtained using the re-implementation setup by [30]. All results are averaged over 3 runs and standard deviation is given in parentheses (b) Ablation study of ACL on miniImageNet dataset

(a)

Method	ACC%	BWT%	Arch (MB)	Replay Buffer (MB)
HAT* [30]	59.45(0.05)	-0.04(0.03)	123.6	-
PNN † [28]	58.96(3.50)	Zero	588	-
ER-RES*	57.32(2.56)	-11.34(2.32)	102.6	110.1
A-GEM* [5]	52.43(3.10)	-15.23(1.45)	102.6	110.1
ORD-FT**	28.76(4.56)	-64.23(3.32)	37.6	-
ORD-JT**	69.56(0.78)	-	5100	-
ACL-JT**	66.89(0.32)	-	5100	-
ACL (Ours)	**62.07(0.51)**	**0.00(0.00)**	113.1	8.5

(b)

	ACC%	BWT%
ACL (1 sample)	**62.07(0.51)**	**0.00(0.00)**
w/o Dis and Shared	29.09(5.67)	Zero
w/o Private	32.82(2.71)	-28.67(3.61)
w/o $\mathcal{L}_{\text{adv}}$ (w/o Dis)	52.07(2.49)	-0.01(0.01)
w/o Replay buffer	57.66(1.44)	-3.71(1.31)
w/o $\mathcal{L}_{\text{diff}}$	60.28(0.52)	0.00(0.00)

Results

- Ablation Studies on 20-Split miniImageNet

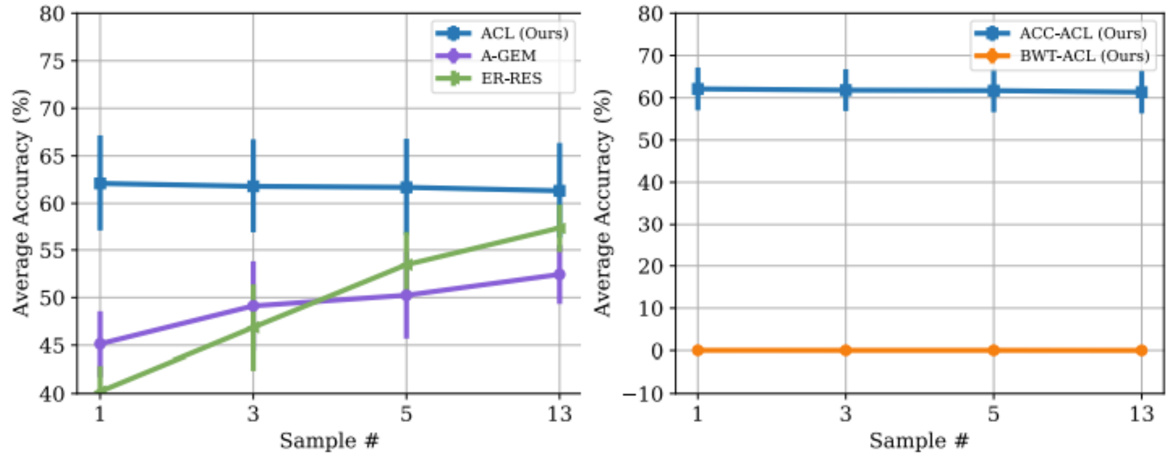


Fig. 2: *Left:* Comparing the replay buffer effect on ACC on 20-Split miniImageNet achieved by ACL against A-GEM [5] and ER-RES [6] when using 1, 3, 5, and 13 samples per classes within each task discussed in 5.2. *Right:* Insensitivity of ACC and BWT to replay buffer in ACL. Best viewed in color.

Visualizing the effect of adversarial learning in ACL

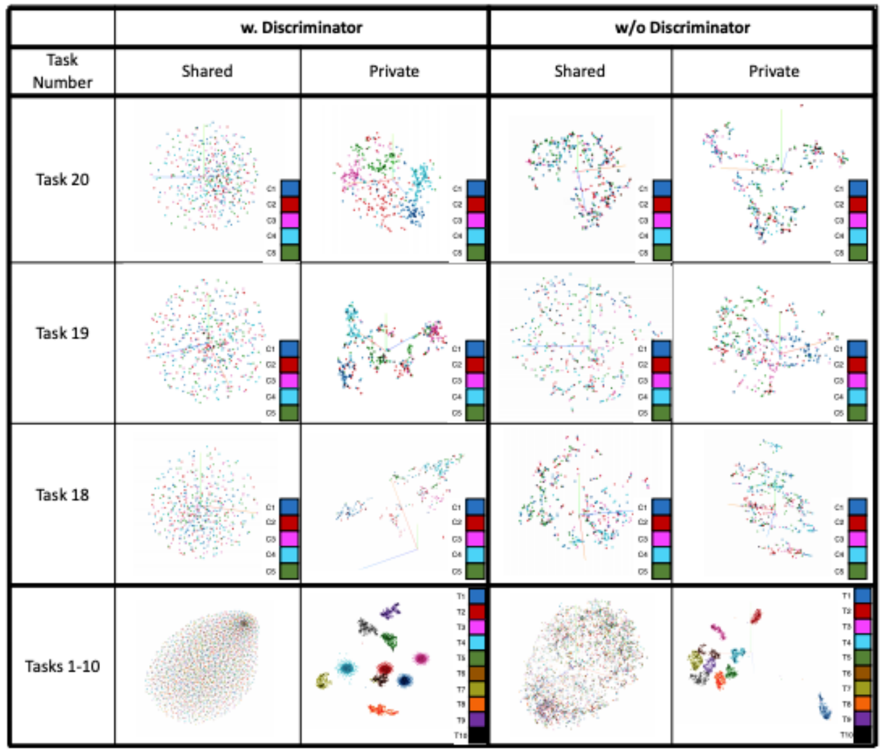


Fig. 3: Visualizing the effect of adversarial learning in ACL where the latent spaces of both private and shared modules are compared against the generated features by corresponding modules trained without a discriminator. This plot shows that the shared module has been successfully trained to generate task-invariant features using adversarial learning whereas in the fourth column from left, we observe that without the discriminator, the shared module was only able to generate a non-uniform embedding

Table 3: CL results on 20-Split CIFAR100 measuring ACC (%), BWT (%), and Memory (MB). (*) denotes that methods do not adhere to the continual learning setup: ACL-JT and ORD-JT serve as the upper bound for ACC for ACL/ORD networks, respectively. † denotes result reported from original work. ‡ denotes result reported from [20]. ** denotes result is reported by [6]. †† denotes result is obtained using the re-implementation setup by [30]. ^{o} denotes result is obtained by using the original provided code. All results are averaged over 3 runs and standard deviation is given in parentheses

(a) 20-Split CIFAR100

Method	**ACC%**	**BWT%**	Arch (MB)	Replay Buffer (MB)
HAT ^{o} [30]	76.96(1.23)	0.01(0.02)	27.2	-
PNN†† [28]	75.25(0.04)	Zero	93.51	-
A-GEM** [5]	54.38(3.84)	-21.99(4.05)	25.4	16
ER-RES**	66.78(0.48)	-15.01(1.11)	25.4	16
ORD-FT*	34.71(3.36)	-48.56(3.17)	27.2	-
ORD-JT*	78.67(0.34)	-	764.5	-
ACL-JT*	79.91(0.05)	-	762.6	-
ACL (Ours)	**78.08(1.25)**	**0.00(0.01)**	25.1	-

(b) Sequence of 5 Datasets

Method	**ACC%**	**BWT%**	Arch (MB)	Replay Buffer (MB)
UCB ^{o} [9]	76.34(0.12)	$-1.34(0.04)$	32.8	-
ORD-FT*	27.32(2.41)	-42.12(2.57)	16.5	-
ACL (Ours)	**78.55(0.29)**	**$-0.01(0.15)$**	16.5	-

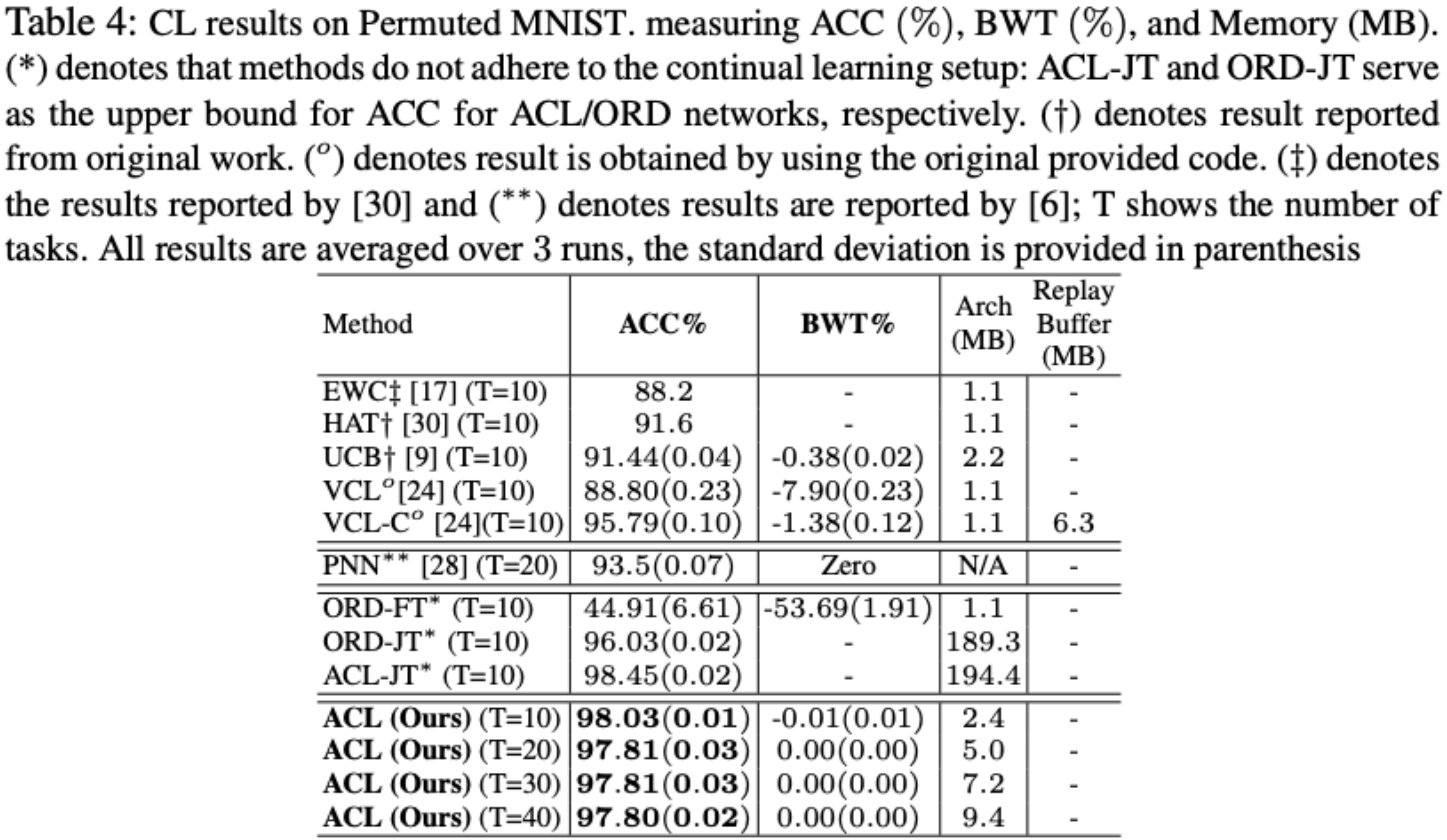
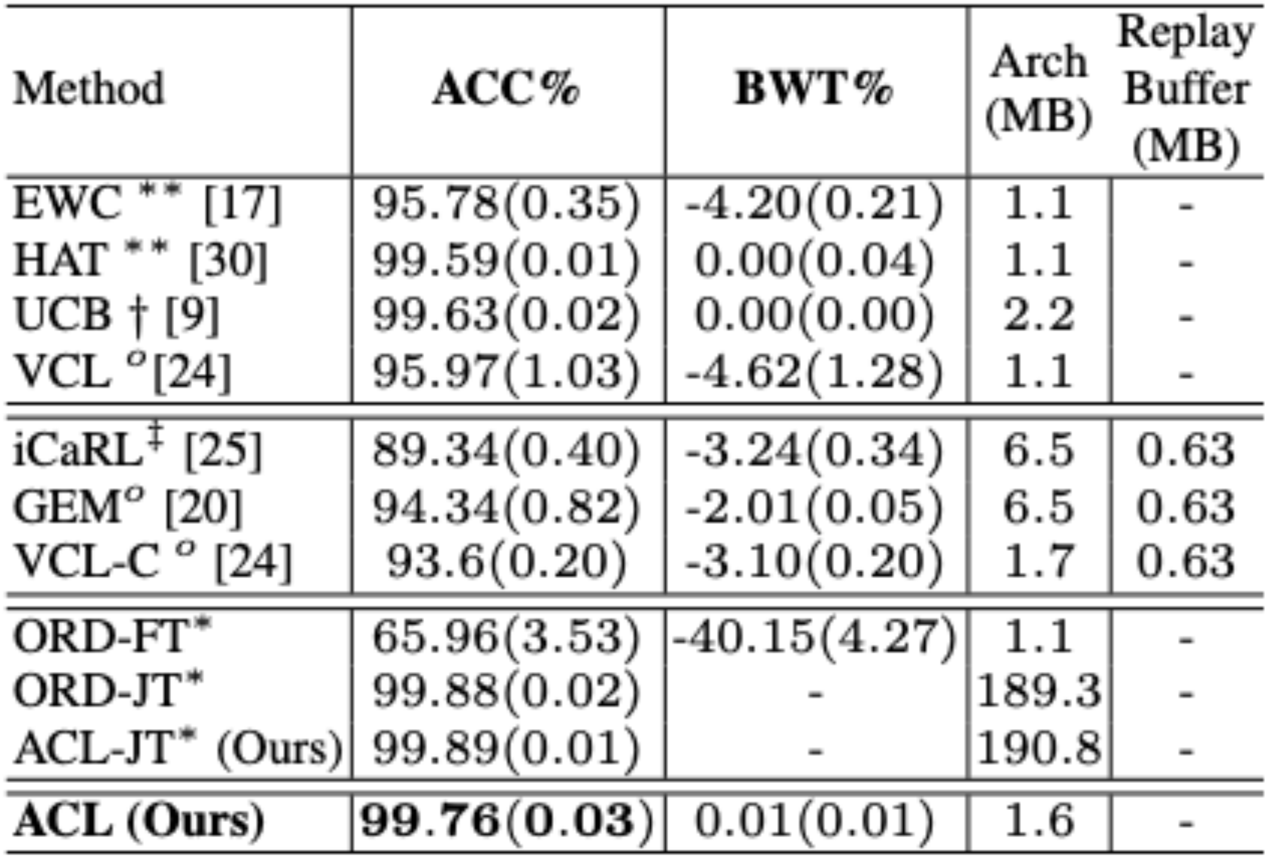


Table 4: CL results on Permuted MNIST. measuring ACC (%), BWT (%), and Memory (MB). (*) denotes that methods do not adhere to the continual learning setup: ACL-JT and ORD-JT serve as the upper bound for ACC for ACL/ORD networks, respectively. (†) denotes result reported from original work. (^{o}) denotes result is obtained by using the original provided code. (‡) denotes the results reported by [30] and (**) denotes results are reported by [6]; T shows the number of tasks. All results are averaged over 3 runs, the standard deviation is provided in parenthesis

Method	**ACC%**	**BWT%**	Arch (MB)	Replay Buffer (MB)
EWC‡ [17] (T=10)	88.2	-	1.1	-
HAT† [30] (T=10)	91.6	-	1.1	-
UCB† [9] (T=10)	91.44(0.04)	-0.38(0.02)	2.2	-
VCL^o[24] (T=10)	88.80(0.23)	-7.90(0.23)	1.1	-
VCL-C^o [24](T=10)	95.79(0.10)	-1.38(0.12)	1.1	6.3
PNN** [28] (T=20)	93.5(0.07)	Zero	N/A	-
ORD-FT* (T=10)	44.91(6.61)	-53.69(1.91)	1.1	-
ORD-JT* (T=10)	96.03(0.02)	-	189.3	-
ACL-JT* (T=10)	98.45(0.02)	-	194.4	-
ACL (Ours) (T=10)	**98.03(0.01)**	-0.01(0.01)	2.4	-
ACL (Ours) (T=20)	**97.81(0.03)**	0.00(0.00)	5.0	-
ACL (Ours) (T=30)	**97.81(0.03)**	0.00(0.00)	7.2	-
ACL (Ours) (T=40)	**97.80(0.02)**	0.00(0.00)	9.4	-

Table 5: Class Incremental Learning on 5-Split MNIST. measuring ACC (%), BWT (%), and Memory (MB). (*) denotes that methods do not adhere to the continual learning setup: ACL-JT and ORD-JT serve as the upper bound for ACC for ACL/ORD networks, respectively. † denotes result reported from original work. ^{o} denotes result is obtained by using the original provided code. All results are averaged over 3 runs, the standard deviation is provided in parenthesis



Method	ACC%	BWT%	Arch (MB)	Replay Buffer (MB)
EWC ** [17]	95.78(0.35)	-4.20(0.21)	1.1	-
HAT ** [30]	99.59(0.01)	0.00(0.04)	1.1	-
UCB † [9]	99.63(0.02)	0.00(0.00)	2.2	-
VCL ^{o}[24]	95.97(1.03)	-4.62(1.28)	1.1	-
iCaRL‡ [25]	89.34(0.40)	-3.24(0.34)	6.5	0.63
GEM^{o} [20]	94.34(0.82)	-2.01(0.05)	6.5	0.63
VCL-C ^{o} [24]	93.6(0.20)	-3.10(0.20)	1.7	0.63
ORD-FT*	65.96(3.53)	-40.15(4.27)	1.1	-
ORD-JT*	99.88(0.02)	-	189.3	-
ACL-JT* (Ours)	99.89(0.01)	-	190.8	-
ACL (Ours)	**99.76(0.03)**	0.01(0.01)	1.6	-

Conclusion

- The novelty of the work is that adversarial learning is used along with orthogonality constraints to disentangle the shared and private latent representations which results in compact private modules that can be stored into memory and hence, efficiently preventing forgetting