BrAIn Lab.
Brain and Artificial Intelligence Lab.

SEOULTECH
SEOUL NATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY

On Tiny Episodic Memories in Continual Learning (ER)

Arslan Chaudhry

2026.02.04

BrainLAB Journal Club
국방인공지능응용학과 이정인

Overview

01 Author & Journal

02 Introduction

03 Background

04 Method

05 Result

Author & Journal

01 Author & Journal

Arslan Chaudhry

팔로우

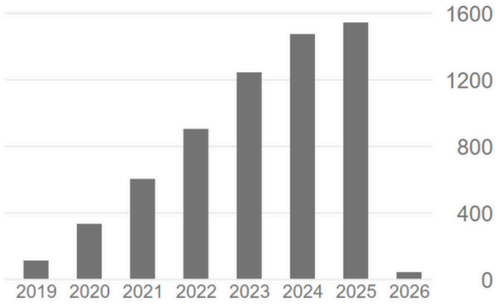
DeepMind
deepmind.com의 이메일 확인됨 - 홈페이지
Machine Learning Artificial Intelligence

내 프로필 만들기

제목	인용	연도
Efficient Lifelong Learning with A-GEM A Chaudhry, MA Ranzato, M Rohrbach, M Elhoseiny ICLR (2019)	2092	2019
Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence A Chaudhry, PK Dokania, T Ajanthan, PHS Torr ECCV (2018)	1743	2018
On Tiny Episodic Memories in Continual Learning A Chaudhry, M Rohrbach, M Elhoseiny, T Ajanthan, PK Dokania, ... ICML, MTLRL Workshop (2019)	1382 *	2019
Using hindsight to anchor past knowledge in continual learning A Chaudhry, A Gordo, PK Dokania, P Torr, D Lopez-Paz AAAI (2021)	323	2020

인용

	전체	2021년 이후
서지정보	6315	5816
h-index	14	14
i10-index	15	15



- Arslan Chaudhry et al.
- arXiv preprint (2019)

Introduction

BrAIn Lab.
Brain and Artificial Intelligence Lab.

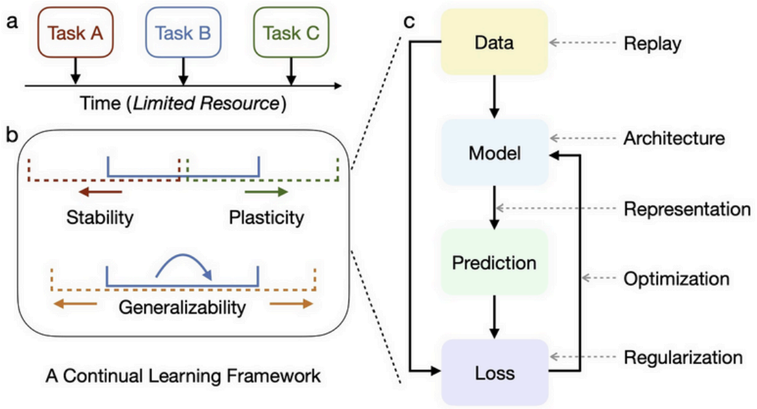
1) Continual Learning

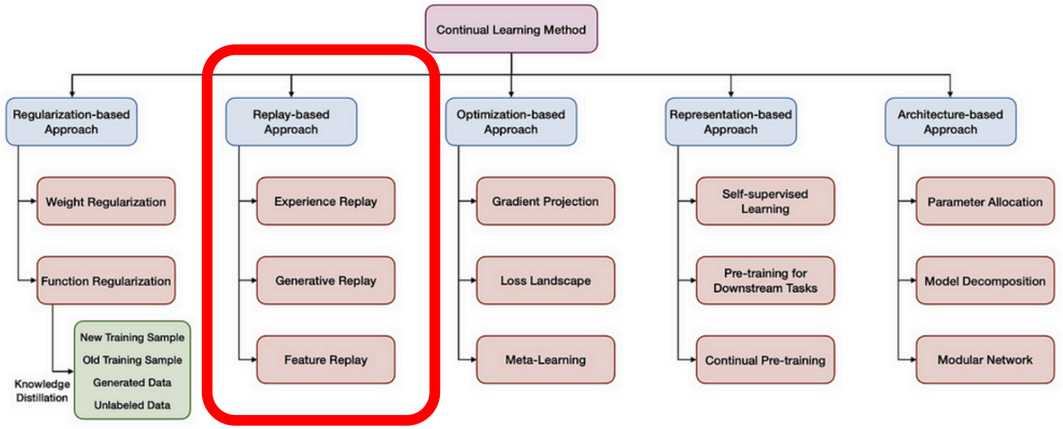
- 여러 작업(Task)을 순차적으로 학습하는 AI 모델이 이전 작업에서 배운 지식을 보존하고, 이를 바탕으로 새로운 작업에 전이(transfer)시키는 학습 방법
- 하지만 현재 대부분의 딥러닝 모델은 "한 번에 하나의 작업만 학습"하도록 설계되어 있어서 **Catastrophic Forgetting** 발생
- Catastrophic Forgetting: 새 작업을 학습하면서 이전 작업의 성능이 급격히 감소하는 현상

2) Episodic Memory

- 과거 작업에서 본 일부 데이터를 저장해두고 새로운 작업을 학습할 때 데이터를 다시 사용하는 방식
- 기존 연구는 주로 메모리 크기가 크거나, 복잡한 제약(A-GEM, MER 등)을 걸어 직접 학습에는 사용하지 않는 방식

→ 복잡한 제약 없이, 현재 작업 데이터 + 메모리 데이터를 같이 학습하기만 해도 성능이 좋을 수 있음을 실험





BrAIn Lab.
Brain and Artificial Intelligence Lab.

1) 정규화 기반 방법 (Regularization-based CL)

1. **일반적인 딥러닝 학습 방법**
- 모든 데이터가 한 번에 존재
- 여러 번 반복 학습 가능
- 과거 데이터를 저장할 수 없는 환경이 다수

2. **Continual Learning(CL)**
- 작업(Task)이 순차적으로 도착
- 과거 데이터는 사라짐
- 새 작업을 배우면 기존 파라미터가 바뀌며 망각 발생

"과거 작업에 중요했던 파라미터를 덜 바꾸면 Catastrophic Forgetting이 줄지 않을까?"

- 과거 작업에서 중요했던 파라미터는 고정
- 덜 중요했던 파라미터만 새 작업에 사용
- 과거 데이터 저장이나 메모리 사용을 안하고, 중요한 파라미터일 수록 움직이면 큰 벌점 부여

정규화 기반 CL:
과거 작업 보존 항을 추가

과거 vs 현재 균형 조절

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{current}}(\theta) + \lambda \sum_i \text{importance}_i \cdot (\theta_i - \theta_i^*)^2$$

현재 파라미터 과거 작업에서 최적이었던 파라미터

BrAIn Lab.
Brain and Artificial Intelligence Lab.

- **중요도(importance) 계산 방식**

EWC (Elastic Weight Consolidation)	SI (Synaptic Intelligence)	MAS (Memory Aware Synapses)
$F_i = \mathbb{E}\left[\left(\frac{\partial \log p(y\|x,\theta)}{\partial \theta_i}\right)^2\right]$	$\Omega_i = \sum_t \frac{\Delta\theta_i^t \cdot g_i^t}{(\Delta\theta_i^t)^2 + \epsilon}$	$\Omega_i = \mathbb{E}\left[\left\|\frac{\partial f(x)}{\partial \theta_i}\right\|\right]$
• Fisher Information Matrix (FIM) 사용 • 파라미터가 조금만 바뀌어도 예측이 크게 달라지면 중요한 파라미터	• 실제 학습 과정에서의 기여도 • 학습 과정 중 누적 변화량으로 중요도 계산	• 레이블 없이도 계산 가능 • 출력의 변화에 민감한 파라미터 보호하여 과거 작업의 성능을 유지
“이 파라미터가 출력에 얼마나 민감한가?”	“이 파라미터가 얼마나 많이, 유용하게 쓰였는가”	“이 파라미터가 출력에 얼마나 영향을 미치고 있는가?”
$\mathcal{L}_{\text{EWC}}(\theta) = \mathcal{L}_{\text{new}}(\theta) + \sum_i \frac{\lambda}{2} F_i(\theta_i - \theta_i^*)^2$	$\mathcal{L}_{\text{SI}}(\theta) = \mathcal{L}_{\text{new}}(\theta) + c\sum_i \Omega_i(\theta_i - \theta_i^*)^2$	$\mathcal{L}_{\text{MAS}}(\theta) = \mathcal{L}_{\text{new}}(\theta) + \lambda \sum_i \Omega_i(\theta_i - \theta_i^*)^2$

BrAIn Lab.
Brain and Artificial Intelligence Lab.

2) 메모리 기반 방법 (Memory-based CL)

과거 작업 데이터를 일부 저장해두고, 이를 활용해 망각을 방지하는 방식

- 장점: 메모리를 이용하면 모델이 실제 데이터를 재학습할 수 있어, 일반화 성능이 좋음
- 단점: 메모리 크기 제한, 샘플 선택 전략 등이 성능에 큰 영향을 미침

1. Replay 방식: 메모리 샘플을 현재 학습과 함께 사용 (e.g. ER, MER)

현재의 작업 손실 + 과거 샘플 손실

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{current}} + \mathcal{L}_{\text{memory}}$$

ER (Experience Replay): 기존 supervised learning 구조에 메모리 샘플을 추가하는 방식
MER (Meta Experience Replay): R + 메타러닝: 메모리 샘플과 현재 샘플의 파라미터 업데이트 방향을 정렬

2. Gradient 방식: 메모리 샘플을 직접 사용하지 않고 제약으로 활용 (e.g. GEM, A-GEM)

GEM (Gradient Episodic Memory): 과거 task의 손실을 증가시키는 방향으로 파라미터를 바꾸지 않도록 제약 $g^\top g_k \geq 0 \quad \forall k$
A-GEM (Average GEM): GEM보다 계산을 줄이기 위해 평균 그래디언트 하나만 사용

$$\text{if } g^\top g_{\text{ref}} < 0: \quad g \leftarrow g - \frac{g^\top g_{\text{ref}}}{g_{\text{ref}}^\top g_{\text{ref}}} g_{\text{ref}}$$

Background

BrAIn Lab.
Brain and Artificial Intelligence Lab.

1) 기존의 지속학습 연구 한계

- **Catastrophic Forgetting:** 모델이 새로운 작업을 학습할 때 이전에 학습했던 작업의 성능이 급격히 저하되는 현상
- **데이터 요구 사항**: 기존 방법은 과거 데이터를 저장하거나, 중요도 정보를 저장하여 망각 방지, 계산 비용이 많이 듦
- **메모리 사용의 과적합 우려**: 에피소드 메모리를 사용하더라도, 과거 데이터를 그대로 다시 학습시키면 해당 데이터에 과소적합되어 오히려 일반화 성능이 떨어질 수 있음

2) Experience Replay (ER) 방법론

- 경험 재현(Experience Replay, ER)을 지속 학습(Continual Learning, CL)에 적용
- tiny episodic memory (1~10) 사용

ER: 강화 학습(Reinforcement Learning, RL) 분야에서 많이 사용되는 기법
에이전트가 경험한 과거 상태-행동-보상 튜플(tuple)을 작은 메모리에 저장해두고, 이를 무작위로 뽑아 현재 학습에 함께 사용하는 방식

Method

04 Method: Experience Replay, ER

BrAIn Lab.
Brain and Artificial Intelligence Lab.

1) 방법론

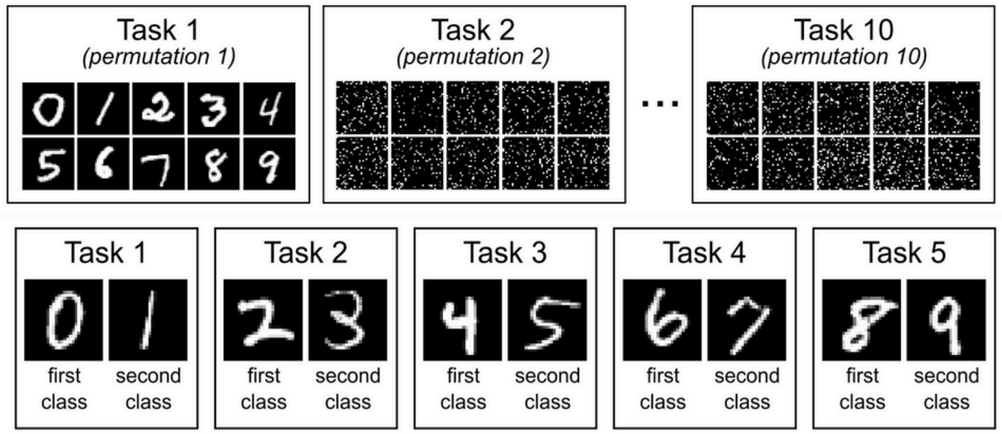
- 현재 태스크의 미니배치와 에피소딕 메모리에서 샘플링한 미니배치를 함께 사용하여 모델 파라미터를 업데이트
- 이전 태스크의 파라미터 벡터에서 새로운 태스크를 fine-tune하는 가장 단순한 baseline에 비해, 에피소딕 메모리를 활용하여 망각 완화

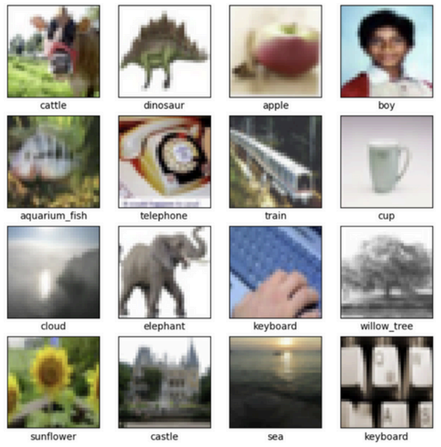
Algorithm 1 Experience Replay for Continual Learning.

procedure ER($\mathcal{D}$, mem_sz, batch_sz, lr)
  $\mathcal{M} \leftarrow \{\} * \text{mem\_sz}$ ▷ Allocate memory buffer of size mem_sz
  $n \leftarrow 0$ ▷ Number of training examples seen in the continuum
  for $t \in \{1, \cdots, T\}$ **do**
    for $B_n \overset{K}{\sim} \mathcal{D}_t$ **do** ▷ Sample without replacement a mini-batch of size K from task t
      $B_{\mathcal{M}} \overset{K}{\sim} \mathcal{M}$ ▷ Sample a mini-batch from $\mathcal{M}$
      $\theta \leftarrow SGD(B_n \cup B_{\mathcal{M}}, \theta, \text{lr})$ ▷ Single gradient step to update the parameters by stacking current minibatch with minibatch from memory
      $\mathcal{M} \leftarrow \texttt{UpdateMemory}(\text{mem\_sz}, t, n, B_n)$ ▷ Memory update, see §4
      $n \leftarrow n + \text{batch\_sz}$ ▷ Counter update
  return $\theta, \mathcal{M}$

2) 데이터셋

1. **Permuted MNIST:** 각 task마다 입력 픽셀에 랜덤한 순열(permutation) 적용
2. **Split CIFAR-100:** 100개의 클래스와 각 클래스당 600개의 이미지 (100개의 클래스를 20개의 disjoint한 작업)
3. **Split miniImageNet:** ImageNet의 일부인 miniImageNet 데이터셋 (100개의 클래스를 20개의 disjoint한 작업)
4. **Split CUB (Caltech-UCSD Birds-200-2011):** 미세한 이미지 분류(fine-grained image classification) 데이터셋 (100개의 클래스를 20개의 disjoint한 작업)





3) 학습 프레임워크 (Protocol for Single-Pass Through the Data)

- 연속된 Task들: T1,T2, ..., Tt
- 각 Task는 독립된 클래스 묶음으로 구성됨
- 모델은 한 번에 하나의 Task만 학습 가능 (Single-pass)

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \mathcal{L}_{\text{memory}} = \frac{1}{|\mathcal{B}|} \sum_{(x,y)\in\mathcal{B}} \ell(f_\theta(x), y) + \frac{1}{|\mathcal{B}_m|} \sum_{(x',y')\in\mathcal{B}_m} \ell(f_\theta(x'), y')$$

- $\mathcal{B}$: 현재 task의 mini-batch
- $\mathcal{B}_m$: 메모리에서 샘플링된 mini-batch
- ℓ: cross-entropy loss
- θ: 모델 파라미터

4) 학습 평가 방식

- **Average Accuracy:** 학습 후 모든 Task에 대한 정확도 평균

$$A_T = \frac{1}{T}\sum_{j=1}^{T} a_{T,j}$$

- **Forgetting Measure:** f_j^i 는 모델이 Task i까지 학습한 후, Task j에서 발생한 망각
  F_T 는 모든 Task를 학습 완료한 시점 (T)에서, 이전 Task들까지에 대한 평균 망각 정도

$$f_j^i = \max_{l\in\{1,\cdots,i-1\}} a_{l,j} - a_{i,j}$$

$$F_T = \frac{1}{T-1}\sum_{j=1}^{T-1} f_j^T$$

5) 메모리 업데이트 전략 (Memory Update Strategies)

✓ 전체 메모리 크기 제한: M (클래스당 1~10 샘플)
✓ 클래스 균형 유지 (Class-balanced memory)

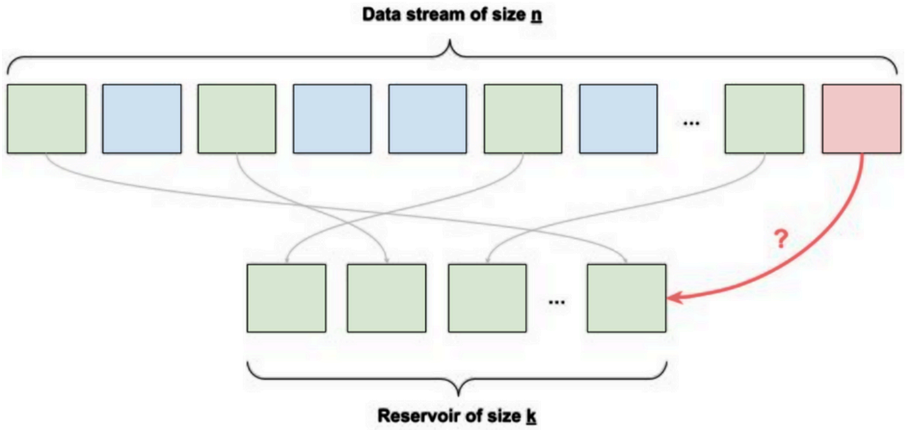
4가지 샘플 선택 전략 실험:

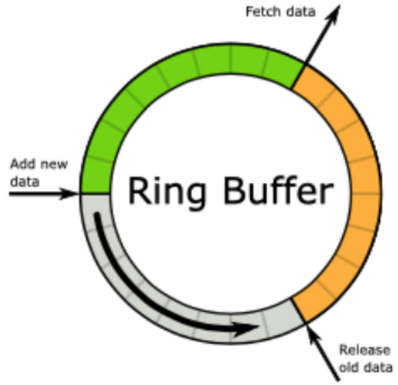
1. **Reservoir Sampling:**

- 길이가 알려지지 않은 스트림에서 무작위로 항목의 부분 집합을 추출하는 알고리즘
- 스트림에서 n번째 데이터 포인터가 들어오면 데이터 포인트가 메모리에 포함될 확률은 mem_sz / n
- 이미 메모리에 데이터가 가득 차 있다면, mem_sz / n (n: 현재까지 본 데이터 총 개수)의 확률로 메모리 내의 기존 샘플 대체

2. **Ring Buffer:**

- 각 클래스마다 동일한 크기의 FIFO 버퍼를 두어, 각 클래스의 가장 최근 mem_sz / C (C: 총 클래스 수)개의 예시를 저장
- 새로운 예시가 들어오면, 해당 클래스의 버퍼에 추가하고, 버퍼가 가득 찼으면 가장 오래된 예시 제거





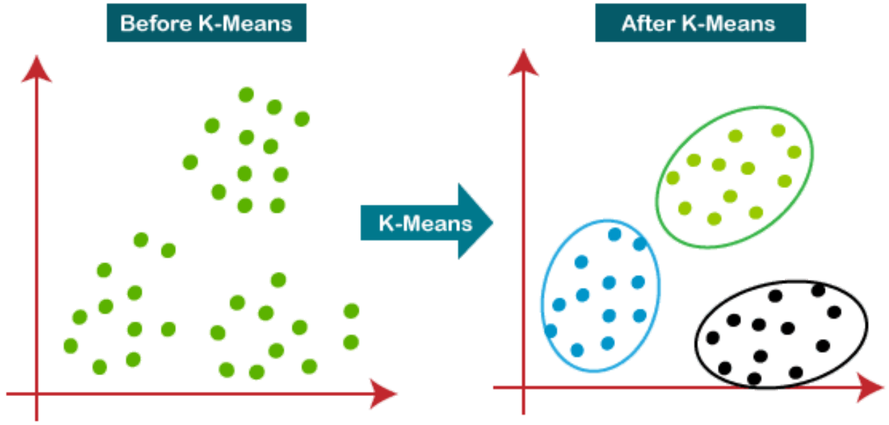
5) 메모리 업데이트 전략 (Memory Update Strategies)

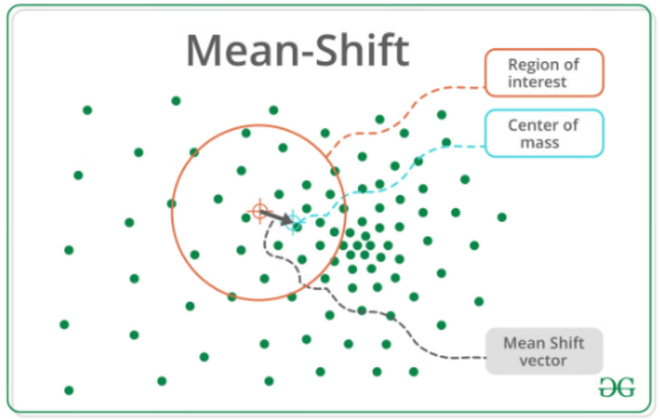
✓ 전체 메모리 크기 제한: M (클래스당 1~10 샘플)
✓ 클래스 균형 유지 (Class-balanced memory)

4가지 샘플 선택 전략 실험:

3. **k-Means:**
- 각 클래스에 대해 특징 공간(feature space)에서의 K-평균 클러스터링을 통해 중심점을 찾고,
- 해당 중심점에 가장 가까운 입력 예시들을 메모리에 저장

4. **Mean of Features (MoF):**
- 각 클래스에 대해 분류 계층 바로 이전의 평균 특징 벡터를 추적
- 이 평균 벡터에 가장 가까운 예시들을 메모리에 저장



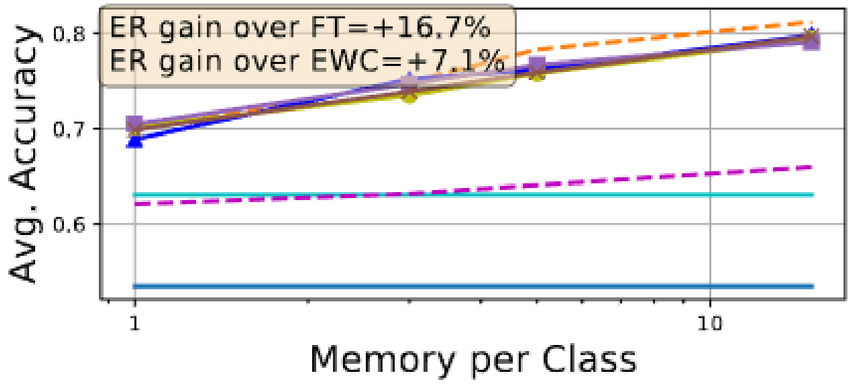


Result

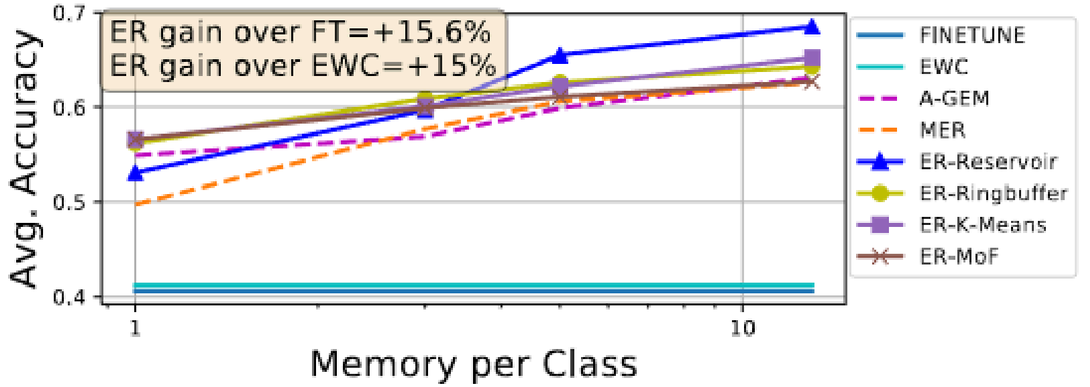
05 Result: Experience Replay, ER

BrAIn Lab.
Brain and Artificial Intelligence Lab.

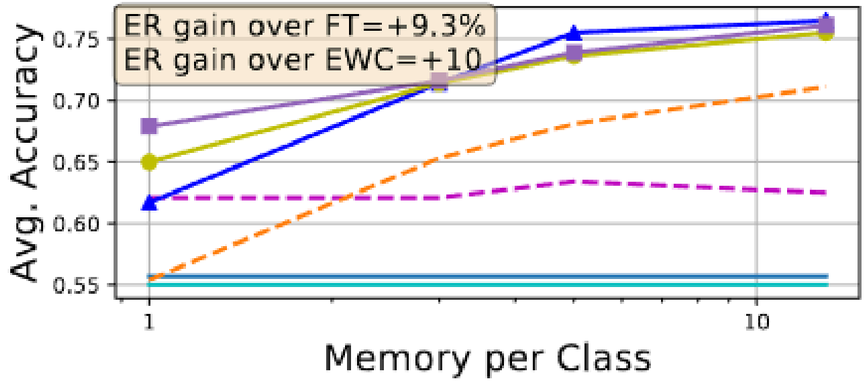
1) 결과



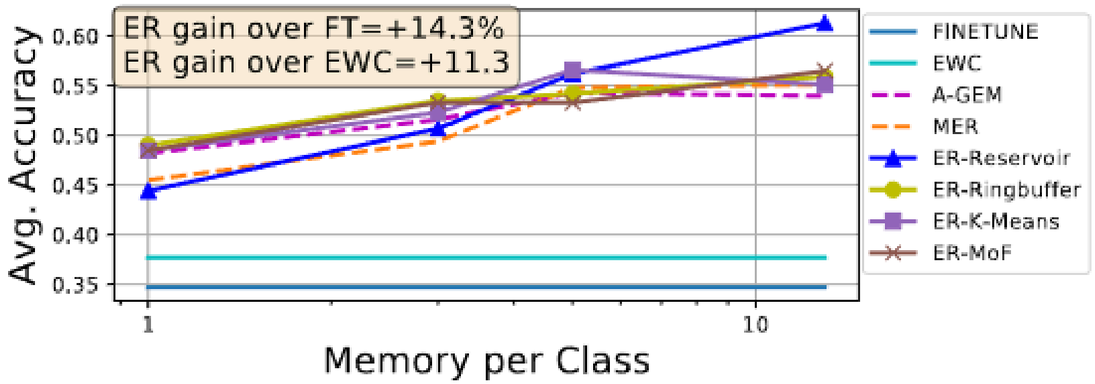
(a) MNIST



(b) CIFAR



(c) CUB



(d) miniImageNet

2) 결과

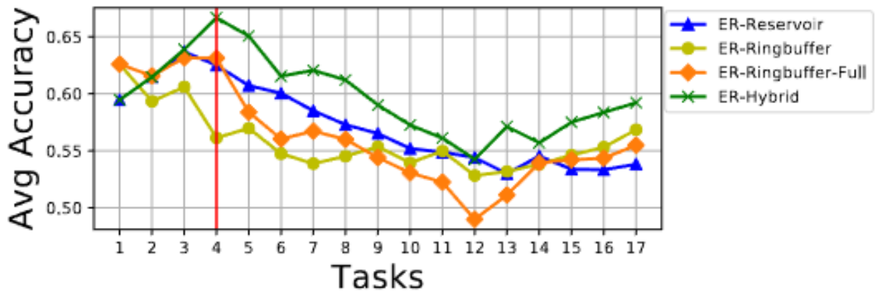


Figure 2: *Evolution of average accuracy (A_k) as new tasks are learned in Split CIFAR. The memory has only 85 slots (in average 1 slot per class). The vertical bar marks where the hybrid approach switches from reservoir to ring buffer strategy. The hybrid approach works better than both reservoir (once more tasks arrive) and ring buffer (initially, when the memory is otherwise not well utilized). The orange curve is a variant of ring buffer that utilizes the full memory at all times, by reducing the ring buffer size of observed classes as new classes arrive. Overall, the proposed hybrid approach works at least as good as the other approaches throughout the whole learning experience. (Averaged over 3 runs).*

Methods	Forgetting			
	MNIST	CIFAR	CUB	miniImageNet
FINETUNE	0.29	0.27	0.13	0.26
EWC	0.18	0.27	0.14	0.21
A-GEM	0.21	0.14	0.09	0.13
MER	0.14	0.19	0.10	0.15
ER-RINGBUFFER (ours)	**0.12**	**0.13**	**0.03**	**0.12**

Table 1: ***Forgetting when using a tiny episodic memory of single example per class.***

Methods	Training Time [s]	
	CIFAR	CUB
FINETUNE	87	194
EWC	159	235
A-GEM	230	510
MER	755	277
ER-RINGBUFFER (ours)	116	255

Table 2: ***Learning Time on $\mathcal{D}^{EV}$ [s]***

Thank You

2026.02.04

BrainLAB Journal Club
국방인공지능응용학과 이정인