

Citation for published version:
Banerjee, S, Verma, VK, Mukherjee, A, Gupta, D, Namboodiri, VP & Rai, P 2024, VERSE: Virtual-Gradient Aware Streaming Lifelong Learning with Anytime Inference. in *2024 IEEE International Conference on Robotics and Automation, ICRA 2024.* Proceedings - IEEE International Conference on Robotics and Automation, IEEE, U. S. A., pp. 493-500, 2024 IEEE International Conference on Robotics and Automation, ICRA 2024, Yokohama, Japan, 13/05/24. https://doi.org/10.1109/ICRA57147.2024.10610702

DOI:
10.1109/ICRA57147.2024.10610702

Publication date:
2024

Document Version
Peer reviewed version

Link to publication

Publisher Rights
CC BY-ND

University of Bath

Alternative formats
If you require this document in an alternative format, please contact:
openaccess@bath.ac.uk

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 06. Mar. 2026

VERSE: Virtual-Gradient Aware Streaming Lifelong Learning with Anytime Inference

Soumya Banerjee[1], Vinay K. Verma[2], Avideep Mukherjee[1], Deepak Gupta[2], Vinay P. Namboodiri[3], Piyush Rai[1]

***Abstract*— Lifelong learning or continual learning is the problem of training an AI agent continuously while also preventing it from forgetting its previously acquired knowledge. Streaming lifelong learning is a challenging setting of lifelong learning with the goal of continuous learning in a dynamic non-stationary environment without forgetting. We introduce a novel approach to lifelong learning, which is streaming (observes each training example only once), requires a single pass over the data, can learn in a class-incremental manner, and can be evaluated on-the-fly (anytime inference). To accomplish these, we propose a novel *virtual gradients* based approach for continual representation learning which adapts to each new example while also generalizing well on past data to prevent catastrophic forgetting. Our approach also leverages an exponential-moving-average-based semantic memory to further enhance performance. Experiments on diverse datasets with temporally correlated observations demonstrate our method's efficacy and superior performance over existing methods.**

I. INTRODUCTION

Continuous machine perception is crucial for AI agents to learn while interacting with the environment, preventing catastrophic forgetting [1]. Lifelong Learning (LL) or Continual Learning (CL) [2], [3] methods are designed with the goal to accomplish this. Recent CL research focuses mainly on static environments [4], [5], [6], [7], [2], assuming large batch data, ignoring changing data distribution, and requires multiple passes over the data to facilitate CL. However, these approaches are not suitable for rapidly changing dynamic environments. While there have been efforts to enable CL in online settings [8], these methods have various limitations, such as batch data requirements, inability to perform anytime inference (i.e., asking the model to make predictions while it is still training), and the need for large replay buffers, limiting their applicability in Streaming Lifelong Learning (SLL) [9], [10], [11]. In SLL, the goal is to learn by observing each training example only once without forgetting.

Below, we outline the key properties of SLL [9], [11]:

- The AI agent observes each training example only once without storing it in memory.
- The agent is required to adapt to new sample(s) in a single pass.
- The input data stream may exhibit temporal correlations, deviating from the typical i.i.d pattern.

[1] IIT Kanpur, India, {soumyab, avideep, piyush}@cse.iitk.ac.in
[2] Amazon, India, {vinayugc, deepakgupta.cbs}@gmail.com
[3] University Of Bath, UK, vpn22@bath.ac.uk



Fig. 1: SLL involves continuous learning from non-i.i.d. labeled streams with multiple views without forgetting. This fig. shows temporally ordered cup frames from CoRe50 [26].

- The agent is required to be evaluated at any time (anytime inference) [1] without fine-tuning its parameters.
- The agent needs to perform class-incremental streaming lifelong learning (CISLL), i.e., predict a class label from all the previously observed classes.
- To make it practical, especially in resource-constrained environments, the agent should minimize its memory requirements.

Existing CL approaches often make strong assumptions that violate one or more key constraints required for SLL. Despite being desirable because of also being closer to biological learning [10], SLL hasn't received much attention. The SLL setting is natural in real-world scenarios like home robots, smart appliances, and drones, where AI agents must adapt quickly and continuously without forgetting. Table I categorizes existing CL approaches based on their underlying assumptions, revealing that only a few non-SLL methods can be adapted to the SLL setting without violating the constraints. Notably, ExStream [9], being an SLL method, violates subset replay in SLL by using all past samples for CL. Non-SLL methods like TinyER [17] and DER/DER++ [18] perform poorly when applied in the SLL setting.

We introduce **V**irtual Gradi**E**nt Awa**R**e **S**treaming L**E**arning (VERSE), a rehearsal-based CL model, facilitating CISLL in deep neural networks (DNNs). VERSE *implicitly* regularizes the network parameters by employing virtual gradient updates, fostering robust representations that require minimal changes to adapt to the new task/sample(s), preventing catastrophic forgetting. We also utilize a small episodic buffer to store past samples, which are used for both local/virtual and global parameter updates to perform a single-step virtual gradient regularization, enabling CL. VERSE first adapts to a new example with a *virtual* (local) parameter update and generalizes to past samples with a

[1]Note that this is different from the *anytime inference* considered in [20], [21], [22], [23], [24] where the model has multiple decision/exit points, one of which is chosen at inference time depending on the desired inference latency; our setting is similar to that considered in [25], [10]

TABLE I: Baseline approaches are categorized based on simplifying assumptions. $\zeta(n)$ denotes the number of times the network visits the data for continual learning. It follows that $\zeta(n) \gg \zeta(2) > \zeta(1)$ with '-' indicating we are unable to find the exact value. Categories: B (Batch), O (Online), S (Streaming).

Methods	LWF [3]	EWC++ [2], [12]	MAS [4]	SI [13]	VCL [14]	CVCL [14]	GEM [15]	AGEM [16]	GDumb [8]	TinyER [17]	DER [18]	DER++ [18]	ExStream [9]	REMIND [10]	CLS-ER [19]	**VERSE (Ours)**	**CISLL Constraints**
Type	B	B	B	B	B	B	O	O	O	O	O	O	S	S	B	S	
Batch-Size (N_t)	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t \gg 1$	$N_t = 1$	$N_t = 1$	$N_t \gg 1$	$N_t = 1$	
Fine-tuning	✗	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗
Single Pass	✗	✗	✗	✗	✗	✗	✓	✓	✗	✓	✓	✓	✓	✓	✗	✓	✓
Follows CIL	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓
Subset Replay	n/a	n/a	n/a	n/a	n/a	✗	✓	✓	✗	✓	✓	✓	✗	✓	✓	✓	✓
Training Time	$\zeta(n)$	$\zeta(n)$	$\zeta(n)$	$\zeta(n)$	$\zeta(n)$	$\zeta(n)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(n)$	$\zeta(1)$	$\zeta(1)$
Inference Time	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(n)$	$\zeta(1)$	$\zeta(1)$	$\zeta(n)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$	$\zeta(1)$
Buffer Size	n/a	n/a	n/a	n/a	n/a	-	-	-	-	$\leq 5\%$	$\leq 5\%$	$\leq 5\%$	$\leq 5\%$	$\gg 10\%$	$\leq 5\%$	$\leq 5\%$	
Doesn't violate CISLL	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗	✓	✗	✓	

global parameter update, promoting convergence between the two. This process facilitates SLL allowing the AI agent to be evaluated on-the-fly (anytime inference) without parameter fine-tuning with the stored samples.

Moreover, VERSE utilizes an exponential-moving-average [27], [28], [29], [30] based semantic memory akin to long-term memory in mammalian brains [31], [32], [33]. Semantic memory is updated intermittently, consolidating new knowledge within the agent's parameters. It interacts with episodic memory, interleaving the past predictions on stored buffer samples, minimizing self-distillation loss [34], [35] to prevent forgetting and enhance the agent's performance.

Experimental results on three temporally contiguous datasets show that VERSE is effective in challenging SLL scenarios. It outperforms recent SOTA methods, with ablations confirming the significance of its components.

In summary, our contributions are as follows:

- We present a novel approach VERSE, a rehearsal-based virtual gradient regularization framework, that incorporates both virtual and global parameter updates to mitigate catastrophic forgetting in CISLL, enabling 'any-time-inference' without fine-tuning.
- We propose a semantic memory based on an exponential-moving-average approach, which enhances the agent's overall performance.
- Through empirical evaluations and ablation studies conducted on three benchmark datasets with temporal correlations, we affirm the superiority of VERSE over the existing SOTA methods.

II. RELATED WORK

This section briefly summarizes different CL paradigms.

Task Incremental Learning (TIL). In TIL, the AI agent learns from task-batches, observing samples related to specific tasks [36], [4], [12], [3], [2], [37], [13], [14], [38], each involving learning a few distinct classes. These methods rely on knowing the task-identifier during inference; otherwise, it leads to severe catastrophic forgetting [12].

Incremental Class Batch Learning (IBL). IBL, also referred to as class incremental learning (CIL), assumes that the dataset is divided into batches, each containing samples from different classes [12], [17], [39], [5], [40], [18], [19], [41], [42], [43], [44]. The AI agent observes and can loop over these batches in each incremental session. During inference, the agent isn't provided with task labels and is evaluated over all the observed classes.

Online Continual Learning (OCL). Unlike TIL and IBL, OCL involves an AI agent sequentially observing and adapting to samples in a *single pass* over the entire dataset, avoiding catastrophic forgetting [8], [17], [45], [46], [16], [15]. While these methods enable continuous learning in dynamic environments, they have various limitations: (i) they require data in batches, assuming $\forall t, |B_t| \gg 1$, where B_t is a batch of samples at time t, (ii) they need fine-tuning before inference, lacking any-time-inference ability (e.g., GDumb [8], which is an OCL method, requires fine-tuning model parameters with replay-buffer samples before each inference), (iii) they require large replay buffers [10].

Streaming Lifelong Learning (SLL). SLL, a challenging variant of LL, enables CL in a rapidly changing environment without forgetting [11], [9], [10]. It shares similarities with OCL but has additional constraints: (i) SLL limits the batch size to one datum per incremental step, while OCL requires $|B_t| \gg 1$, (ii) it doesn't allow AI agent to fine-tune its parameters during training or inference. Additionally, in SLL, the input data stream can be temporally correlated in terms of class instance and instance ordering. Detailed essential and desirable properties of SLL are discussed in Section I.

To our knowledge, ExStream [9], REMIND [10], and BaSiL [11] are the three methods tackling the challenging SLL setting. However, it's important to note some of the key differences: ExStream [9] uses full-buffer replay, violating the subset-replay constraint; REMIND [10] stores a much larger number of past samples compared to other baselines (e.g., iCaRL [47] stores 10K past ImageNet [48], [49] samples, whereas REMIND stores 1M samples); BaSiL [11] focuses on Bayesian methods for SLL and relies entirely on pretrained weights for visual features in SLL. It does not adapt convolutional layers to sequential data and only trains linear layers, potentially posing severe challenges with non-

i.i.d. data. In contrast, our approach (VERSE) adheres to the SLL constraints, stores a limited number of past samples, replays only a subset of buffer samples, and trains both convolutional and fully connected (FC) layers for SLL.

In this paper, we introduce VERSE, adhering to CISLL constraints, enabling CL in challenging SLL settings. We compare VERSE with SLL frameworks, REMIND [10] and ExStream [9], as well as various OCL and IBL methods.

III. Proposed Approach

In this section, we introduce the proposed approach VERSE (Fig. 2), which trains the CNN architecture in CISLL setup. The model, consisting of the parameters $\Theta = \{\xi, \theta\}$, comprises of two components: (i) a non-plastic feature extractor (G_ξ) with parameter ξ, and (ii) a plastic neural network (F_θ) with parameter θ. G_ξ includes the initial CNN layers, and F_θ encompasses the final few layers, including the fully connected (FC) layers. The class label output for a given input $\boldsymbol{x}$, is predicted as: $y = F_\theta(G_\xi(\boldsymbol{x}))$.

We focus on adapting the plastic network $F_\theta(\cdot)$ while keeping non-plastic parameters ξ frozen throughout. In each streaming incremental step of CISLL, data arrives sequentially, $\mathscr{D}_t = (\boldsymbol{x}_t, y_t)$, one datum at a time, and the learner adapts without catastrophic forgetting [50], [51] by observing this datum only once. The following section provides brief details of the proposed model.

A. Virtual Gradient as Regularizer

We denote by $\mathscr{D}_1, \mathscr{D}_2, \ldots, \mathscr{D}_T$ the set of tasks, with $(\boldsymbol{z}_t, y_t) = (G_\xi(\boldsymbol{x}_t), y_t) \sim \mathscr{D}_t$ with $|\mathscr{D}t| = 1$, arriving one-by-one in each incremental step. During the training of the t^{th} task only the data $\mathscr{D}_t$ is available, the previous tasks' data $\mathscr{D}_1, \mathscr{D}_2 \ldots \mathscr{D}_{t-1}$ are discarded, and we only keep a few samples into a small memory buffer $\mathscr{M}$. Optimizing network parameters with a single example is challenging; therefore, we select a subset of samples with size C from memory $\mathscr{M}$, that is, $\mathscr{D}_v^t \sim \mathscr{M}$ with $|\mathscr{D}_v^t| = C$. We combine the subset $\mathscr{D}_v^t$ with the new example $\mathscr{D}_t$ to form a joint batch: $\mathscr{D}_v^t \leftarrow \mathscr{D}_t \cup \mathscr{D}_v^t$ and compute the cross-entropy loss, as defined below:

$$\mathscr{L}_v^t = \mathbb{E}_{(\boldsymbol{z},y)\sim\mathscr{D}_v^t}\left[\mathscr{L}_{CE}(y, F_\theta(\boldsymbol{z}))\right] \tag{1}$$

Let $\nabla_\theta \mathscr{L}_v^t(F_\theta)$ be the gradient of the loss (Eq. 1) w.r.t. the plastic network parameters θ. With this gradient, we compute the updated *local parameter* as follows:

$$\theta^v \leftarrow \theta - \alpha \nabla_\theta \mathscr{L}_v^t(F_\theta) \tag{2}$$

The optimization above is a virtual/local gradient update, as it doesn't alter the model parameters θ. However, θ^v is optimized focusing on the novel sample, which may not generalize well to the observed past samples due to changes in the previously optimal network parameters, leading to forgetting. To address this, we perform a global optimization with rehearsal. For this, we choose two more sample subsets from memory: $\mathscr{D}_l^t, \mathscr{D}_\mathscr{M}^t \sim \mathscr{M}$, each with size C. Let $H_l^t \leftarrow F_\Phi(\mathscr{D}_l^t)$ represent the logits obtained over replay samples $\mathscr{D}_l^t$, with $F_\Phi(\cdot)$ denoting the semantic memory (see Sec. III-C). Then, we compute the loss over virtual parameters θ^v using

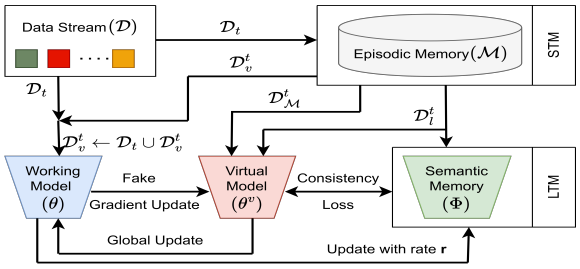


Fig. 2: In VERSE, Virtual-gradient-regularization (VGR) enables CL by adapting to new sample(s) with a virtual model (θ^v), which computes the final model (θ) through rehearsal. Episodic memory (TEM) stores a few observed samples, while Semantic memory (SEM) enforces consistency with self-distillation loss, improving overall performance.

both subsets from the replay buffer as the sum of cross-entropy and knowledge distillation loss [34], defined as:

$$\mathscr{L}^t = \mathbb{E}_{(\boldsymbol{z},y)\sim\mathscr{D}_\mathscr{M}^t}\left[\mathscr{L}_{CE}(y, F_{\theta^v}(\boldsymbol{z}))\right] + \lambda \mathscr{L}_{MSE}(H_l^t, F_{\theta^v}(\mathscr{D}_l^t)) \tag{3}$$

Eq. 3 assesses the generalization loss over the rehearsal subsets using virtual parameters optimized for the new streaming example. If the model exhibits forgetting, then the virtual parameters will incur high loss due to poor generalization. Otherwise, the loss will be small.

Suppose $\nabla_{\theta^v}\mathscr{L}^t(F_{\theta^v})$ be the gradient of the loss in Eq. 3 w.r.t. the virtual model parameters (θ^v). Then, we can compute the *global parameters*, for the plastic network (θ), as follows:

$$\theta \leftarrow \theta - \beta \nabla_{\theta^v}\mathscr{L}^t(F_{\theta^v}) \tag{4}$$

Eq. 4 updates θ using the gradient of the virtual parameter, which might appear counter-intuitive. However, the alternating competitive training between Eq. 2 and Eq. 4 is crucial. Below, we briefly discuss this training behavior.

The updates in Eq. 2 and Eq. 4 occur alternately. Eq. 2 assesses how well θ generalizes to new data, while Eq. 4 focuses on θ^v's generalization to past data. For minimum loss (in Eq. 1), θ must adapt to new samples, which is more challenging as compared to θ^v, which is optimized for new samples. Conversely, θ^v may struggle to generalize to past samples due to its emphasis on new data. Hence, both θ and θ^v need to generalize effectively to new and replayed samples, minimizing losses in both Eq. 1 and Eq. 3. In this alternating learning, convergence occurs when these losses approach zero, implying mutual generalization and reduced forgetting.

B. Tiny Episodic Memory (TEM)

The model uses a fixed-sized tiny episodic memory (TEM) to act as short-term memory. In each incremental step, it (i) replays subsets of (C) samples uniformly selected from memory for continual learning, and (ii) stores the new sample in memory. It employs Reservoir Sampling [52] and Class Balancing Random Sampling to maintain a fixed-sized replay buffer. Reservoir Sampling selects a random buffer sample to replace with the new example, while Class Balancing Random Sampling picks a sample from the most populated class in the buffer to replace the new one.

TABLE II: Ω_{all} results. Best-performing CISLL method is highlighted in **Bold**. The reported results are an average over 10 runs with different permutations of data. Offline model is trained once. $\widehat{\text{Offline}} = \frac{1}{T}\sum_{t=1}^{T} \alpha_{\text{Offline},t}$, with '-' denoting experiments we are unable to run due to compatibility issues.

Method	iid			Class-iid			instance			Class-instance		
	iCub1.0	iCub28	CoRe50	iCub1.0	iCub28	CoRe50	iCub1.0	iCub28	CoRe50	iCub1.0	iCub28	CoRe50
Fine-tune	0.9550	0.8432	0.9681	0.3902	0.4265	0.4360	0.1981	0.2483	0.2468	0.3508	0.4810	0.3400
EWC++ [2], [12]	-	-	-	0.3747	0.4218	0.4307	-	-	-	0.3507	0.4805	0.3401
MAS [4]	-	-	-	0.3758	0.4334	0.4333	-	-	-	0.3509	0.4807	0.3401
AGEM [16]	-	-	-	0.4626	0.7507	0.5633	-	-	-	0.3510	0.4811	0.3399
FIFO	0.9269	0.9774	0.9943	0.4971	0.6550	0.4763	0.3257	0.2807	0.1481	0.3609	0.4811	0.3399
GDumb [8]	0.9269	0.7076	0.9502	0.9683	0.8293	0.9767	0.6240	0.4704	0.6521	0.7734	0.6481	0.6628
TinyER [17]	0.9852	0.9752	1.0064	0.9766	0.8584	0.9723	0.9324	0.7995	0.9315	0.8825	0.7074	0.8525
DER [18]	0.5976	0.8625	0.9807	0.8727	0.8402	0.9734	0.7972	0.8397	0.9870	0.8293	0.8286	0.9630
DER++ [18]	0.9004	0.9020	0.9985	0.9398	0.8746	0.9786	0.8785	0.8547	0.9933	0.9125	0.8484	0.9696
CLS-ER [19]	0.9573	0.1837	0.1107	0.5010	0.6664	0.3182	0.6854	0.1837	0.1107	0.5007	0.5858	0.2580
ExStream [9]	0.9114	0.8053	0.9286	0.9035	0.8375	0.8884	0.8713	0.7389	0.8530	0.8806	0.8339	0.9091
REMIND [10]	0.9666	0.9483	0.9988	0.9544	0.8197	0.9507	0.9102	0.7764	0.8993	0.8453	0.6784	0.8259
Ours	**1.0087**	**1.0045**	**1.0202**	**1.0069**	**0.8874**	**0.9918**	**0.9613**	**0.8555**	**0.9945**	**0.9985**	**0.8840**	**0.9851**
Offline	1.000	1.0000	1.0000	1.0000	1.0000	1.0000	1.000	1.0000	1.0000	1.0000	1.0000	1.0000
$\widehat{\text{Offline}}$	0.8046	0.8726	0.9038	0.8785	0.9266	0.9268	0.8046	0.8726	0.9038	0.8785	0.9266	0.9268

C. Semantic Memory (SEM)

Semantic memory (SEM) retains long-term knowledge and combats forgetting using self-distillation-loss [34], [53], [54], [35], [55] minimization (Eq. 3), aligning the current model's decision boundary with past memories. SEM, based on a DNN and initialized with working model parameters, absorbs knowledge from the working network θ in incremental steps. Inspired by mean-teacher [27], [28], [29], [30], SEM is updated stochastically via exponential moving average (EMA) rather than at every iteration. Given a randomly sampled value u from a uniform distribution ($u \sim \mathscr{U}(0,1)$) and an acceptance probability r, the update process for SEM denoted with Φ and the working model parameter θ, is defined as follows

$$\Phi = \begin{cases} \gamma\,\Phi + (1-\gamma)\,\theta & ,u < r \\ \Phi & ,Otherwise \end{cases} \tag{5}$$

The acceptance probability is a hyper-parameter that regulates the frequency of SEM updates. A lower (higher) acceptance probability means less (more) frequent updates, retaining more (less) information from the remote model. This update resembles the mammalian brain, with information initially stored in short-term memory before transitioning to long-term memory. Algorithm 1 illustrates the different stages of the proposed model.

IV. Experiments

A. Datasets, Data Orderings and Metrics

Datasets. We evaluate our method (VERSE) through extensive experiments on three temporally coherent datasets: iCub1.0 [56], iCub28 [57], and CoRe50 [26]. iCub1.0 involves object recognition from video frame sequences, with each frame containing a single object instance, while iCub28 is similar but spans across four days. CoRe50, like iCub1.0/28, includes temporally ordered images divided into 11 sessions with varying backgrounds and lighting.

Data Orderings. To test VERSE's robustness in a challenging SLL setup, we assess its streaming learning capability using four data-ordering schemes, as in [10], [9], [11]. These schemes include (i) streaming i.i.d., (ii) streaming class i.i.d., (iii) streaming instance, and (iv) streaming class instance ordering.

TABLE III: Buffer capacity used for various datasets.

Dataset	iCub1.0	iCub28	CoRe50	ImageNet100
Buffer Capacity	230	230	1000	1000
Training-Set Size	6002	20363	119894	127778

Algorithm 1 VERSE

Require: Initialize: $\Phi = \theta$
Require: Hyperparameters: $\lambda, r, \alpha, \beta, \gamma$, Memory: $\mathscr{M}$
for $t \in 1,\ldots,T,\ldots$ **do**
  $\{(z_t, y_t)\} = \{(G_\xi(x_t), y_t)\} \sim \mathscr{D}_t$ ▷ $|\mathscr{D}_t| = 1$
  Select samples from $\mathscr{M}$: $\mathscr{D}_v^t \sim \mathscr{M}$ ▷ $|\mathscr{D}_v^t| = C$
  $\mathscr{D}_v^t \leftarrow \mathscr{D}_t \cup \mathscr{D}_v^t$
  Compute $\mathscr{L}_v^t$ using Eq. 1 and evaluate $\nabla_\theta \mathscr{L}_v^t(F_\theta)$
  $\theta^v = \theta - \alpha \nabla_\theta \mathscr{L}_v^t(F_\theta)$ ▷ Virtual gradient update
  Select samples from $\mathscr{M}$: $\mathscr{D}_l^t, \mathscr{D}_{\mathscr{M}}^t \sim \mathscr{M}$ ▷ $|\mathscr{D}_l^t| = |\mathscr{D}_{\mathscr{M}}^t| = C$
  $H_l^t \leftarrow F_\Phi(\mathscr{D}_l^t)$
  Compute $\mathscr{L}^t$ using Eq. 3 and evaluate $\nabla_{\theta^v} \mathscr{L}^t(F_{\theta^v})$
  $\theta = \theta - \beta \nabla_{\theta^v} \mathscr{L}^t(F_{\theta^v})$ ▷ Global update
  sample $u \sim \mathscr{U}(0,1)$
  if $u < r$ **then**
    $\Phi \leftarrow \gamma\,\Phi + (1-\gamma)\,\theta$
  UpdateMemory$(\mathscr{M}, \mathscr{D}_t, t)$ ▷ Add sample to $\mathscr{M}$
return θ, Φ

Metrics. To assess the learner's performance in a CISLL setup, we employ the Ω_{all} metric, following the approach in [7], [9], [10], [11]. This metric quantifies CL performance normalized against an *Offline* baseline:

$$\Omega_{\text{all}} = \frac{1}{T}\sum_{t=1}^{T} \frac{\alpha_t}{\alpha_{\text{Offline},t}} \tag{6}$$

where (i) T is the total number of testing events, (ii) α_t is the streaming learner's unnormalized performance at time t, and (iii) $\alpha_{\text{Offline},t}$ is the unnormalized performance of the Offline baseline at time t.

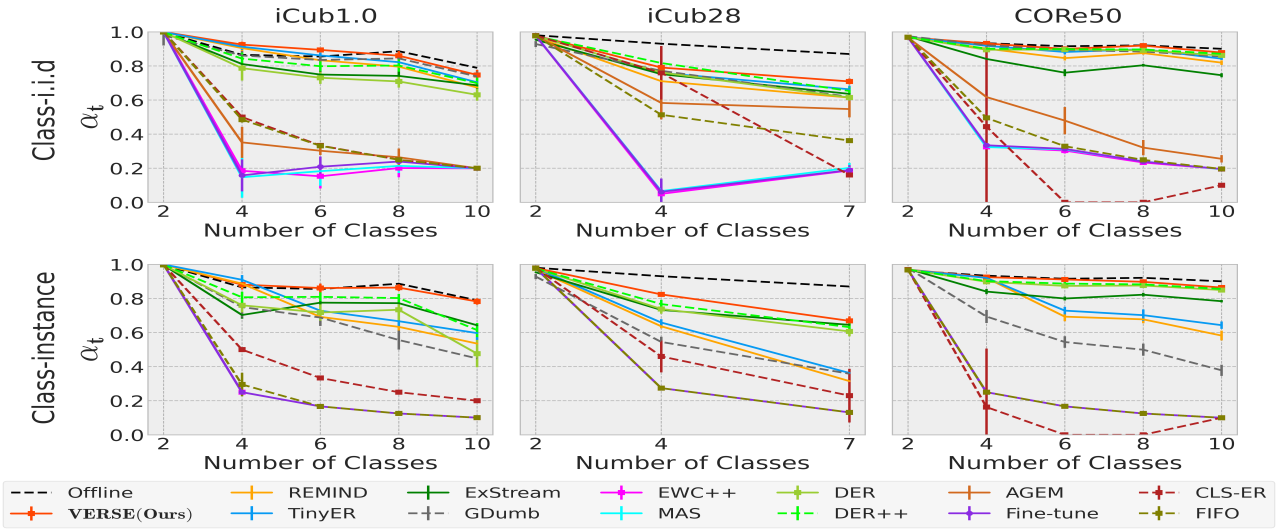


Fig. 3: Plots of α_t as a function of streaming learning model and data-orderings. VERSE outperforms other SLL models in both streaming class-iid (top-row) and streaming class-instance (bottom-row) orderings across datasets.

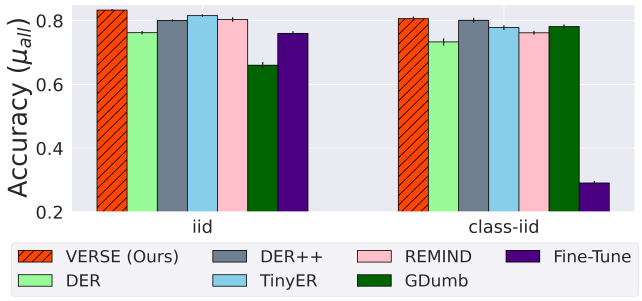


Fig. 4: Performance (μ_{all}) comparison between VERSE (Ours) and the other baselines on ImageNet100.

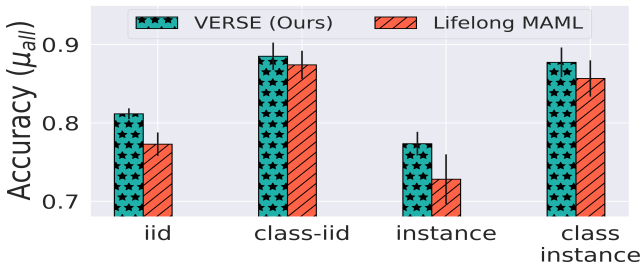


Fig. 5: Performance (μ_{all}) comparison between VERSE and Lifelong MAML [60] on iCub1.0.

B. Baselines and Compared Methods

VERSE adheres to the challenging CISLL approach. We compare it with ExStream [9] and REMIND [10]. We also evaluate various IBL and OCL approaches (EWC++, MAS, AGEM, GDumb, TinyER, DER/DER++, CLS-ER, FIFO), as well as two additional baselines: (i) offline training with full dataset access (Offline/Upper Bound), and (ii) fine-tuning with one example at a time and no CL strategy (Fine-tuning/Lower Bound). All comparisons are performed under the SLL setup, except for *GDumb*, which fine-tunes with replay buffer samples, giving it an unfair advantage.

C. Implementation Details

In all experiments, baselines are trained with one sample at a time using the same network architecture. We employ ResNet-18 [58] pretrained on ImageNet-1K [48], [49] available in PyTorch [59] TorchVision package, using its first 15 convolutional (conv) layers and 3 downsampling layers as the feature extractor (G). The remaining 2 conv layers and 1 fully connected (FC) layer constitute the plastic network (F). For ExStream [9], all 17 conv and 3 downsampling layers are utilized for feature extraction (G), and the final FC layer serves as the plastic network (F). Feature embeddings are stored in memory for all baselines, including VERSE. Replay buffer capacity is specified in Table III. We employ reservoir sampling for class-instance and instance ordering and class-balancing random sampling for class-iid and iid ordering. Experience-replay and self-distillation consistently use $C = 16$ samples across all baselines. Hyperparameters are set as follows: $\alpha = 0.005, \beta = 0.01, \lambda = 0.3$, and $\gamma = 0.9$. r values are set as: (i) $r = 0.4$ for iCub1.0, (ii) $r = 0.1$ for iCub28, and (iii) $r = 0.05$ for CORe50 dataset. Each experiment is repeated 10 times with different data permutations, and the average accuracy is reported.

D. Results

Table II presents VERSE's performance in various experimental setups with different data orderings and datasets. We conducted 10 repetitions of each experiment, reporting average accuracy. Notably, VERSE consistently surpasses the baselines by a significant margin. It demonstrates robustness to different data-ordering schemes, which are known to induce catastrophic forgetting. In contrast, IBL methods like EWC++ [2], [12] and MAS [4] experience severe forgetting. Even *GDumb* [8], which fine-tunes network parameters with buffer samples before each inference, fails to outperform VERSE.

iCub1.0/28 and CoRe50 are temporally coherent datasets, offering a more realistic and challenging evaluation scenario. Class-instance and instance ordering requires the agent to learn from temporally ordered video sequences one at a time. Table II shows that VERSE achieves notable improvements:

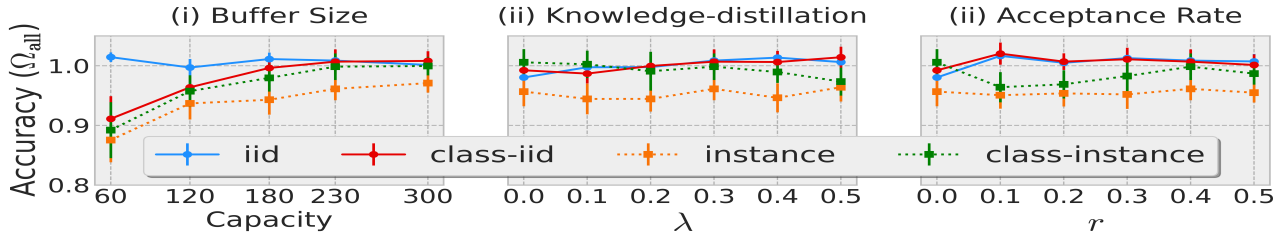


Fig. 6: Plots of $\boldsymbol{\Omega}_{\text{all}}$ as as function of (i) replay buffer capacity $(|\mathscr{M}|)$, (ii) knowledge-distillation hyper-parameter (λ), and (iii) acceptance-rate hyper-parameter (r) on iCub1.0.

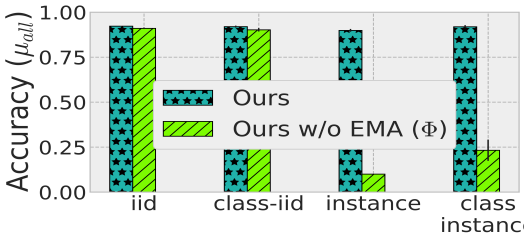


Fig. 7: Plots of $\boldsymbol{\mu}_{\text{all}}$ as a function of EMA (Φ) used to compute the semantic memory (Eq. 5) on CORe50.



Fig. 8: Plots of $\boldsymbol{\mu}_{\text{all}}$ as a function of buffer replacement policies on iCub1.0.

(i) up to 8.6% and 2.89% on iCub1.0 for class-instance and instance ordering, (ii) 3.56% on iCub28 for class-instance ordering, and (iii) 1.55% on CoRe50 for class-instance ordering. Fig. 3 plots accuracy (α_t) of VERSE (Ours) and other baselines for class-iid and class-instance ordering. Notably, VERSE better retain knowledge of old classes compared to other baselines, particularly excelling in class-instance ordering.

We also assess VERSE and other baselines on ImageNet100, a subset of ImageNet-1K (ILSVRC-2012) [48], [49], comprising randomly selected 100 classes, each with 700-1300 training samples and 50 validation samples. As ImageNet-1K lacks labels for test samples, we used the validation set for testing, following [10]. Fig. 4 illustrates the performance $(\boldsymbol{\mu}_{\text{all}})$ of various baselines, including VERSE, showing VERSE consistently outperforming all other CL methods. $\boldsymbol{\mu}_{\text{all}}$ represents the mean-absolute accuracy with $\boldsymbol{\mu}_{\text{all}} = \frac{1}{T}\sum_{t=1}^{T}\boldsymbol{\alpha}_t$, where (i) T is the total number of testing events and (ii) $\boldsymbol{\alpha}_t$ is the accuracy of the streaming learner at time t.

Finally, we compare VERSE and Lifelong MAML [60], a continual learning variant of MAML [61], on the iCub1.0 dataset. Fig. 5 illustrates the comparison using $\boldsymbol{\mu}_{\text{all}}$ metric with VERSE consistently outperforming LifeLong MAML across all data-orderings.

V. Ablations

We perform extensive ablations to validate the importance of the various components of VERSE.

Choice Of Buffer Capacity. Fig. 6 (left) shows the impact of different buffer capacities on iCub1.0. Increased buffer capacity leads to improved model performance.

Choice Of Hyper-parameter (λ). Fig. 6 (middle) shows the impact of changing the self-distillation hyper-parameter (λ) on iCub1.0. The best performance is consistently achieved across all data orderings with $\lambda = 0.3$.

Significance Of Self-Distillation Loss. Fig. 6 (middle) depicts the model's performance with $\lambda = 0.0$, indicating no self-distillation. While the best performance is achieved with $\lambda = 0.3$, self-distillation alone does not significantly improve performance.

Significance Of Acceptance-Rate (r). Fig. 6 (right) illustrates the impact of changing the acceptance-rate (r) on iCub1.0. The best performance is achieved with $r = 0.4$. However, increasing r to 0.50 leads to performance degradation. For instance ordering, the model tends to perform best with $r = 0.0$.

Significance of Exponential Moving Average (EMA). Fig. 7 highlights the importance of SEM (Φ) in the model's performance. Without using SEM and relying solely on the working model (θ) for computing logits in self-distillation loss (Eq. 3), the model's performance degrades. Additionally, for temporally coherent orderings (instance and class instance orderings), not using EMA to update SEM severely degrades VERSE's performance.

Significance of Buffer Replacement Policies. Fig. 8 shows the model's performance with different buffer replacement policies used for TEM. For temporally ordered data (instance and class instance ordering), reservoir sampling yields the best performance. However, for i.i.d and class i.i.d ordering, class balancing random sampling or balanced sampling achieves the best results.

VI. Conclusion

We address the challenging problem of streaming lifelong learning, where the learner is given only one sample at a time during training, the learned model is required to have anytime inference capability. Our replay-based virtual-gradient-regularization with global and virtual/local parameters generalization to both previous and novel task samples. Tiny episodic memory for rehearsal and semantic memory help align the decision boundary with past memories through self-distillation-loss. Extensive experiments and ablations on various datasets and data orderings demonstrate our approach's efficacy.

REFERENCES

[1] S. Thrun, "Lifelong learning algorithms," in *Learning to learn*. Springer, 1998, pp. 181–209.

[2] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[3] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.

[4] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 139–154.

[5] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu, "Large scale incremental learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 374–382.

[6] C. Wu, L. Herranz, X. Liu, J. van de Weijer, B. Raducanu, *et al.*, "Memory replay gans: Learning to generate new categories without forgetting," in *Advances In Neural Information Processing Systems*, 2018, pp. 5962–5972.

[7] R. Kemker and C. Kanan, "Fearnet: Brain-inspired model for incremental learning," *arXiv preprint arXiv:1711.10563*, 2017.

[8] A. Prabhu, P. H. Torr, and P. K. Dokania, "Gdumb: A simple approach that questions our progress in continual learning," in *European Conference on Computer Vision*. Springer, 2020, pp. 524–540.

[9] T. L. Hayes, N. D. Cahill, and C. Kanan, "Memory efficient experience replay for streaming learning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 9769–9776.

[10] T. L. Hayes, K. Kafle, R. Shrestha, M. Acharya, and C. Kanan, "Remind your neural network to prevent catastrophic forgetting," *arXiv preprint arXiv:1910.02509*, 2019.

[11] S. Banerjee, V. K. Verma, and V. P. Namboodiri, "Streaming lifelong learning with any-time inference," *arXiv preprint arXiv:2301.11892*, 2023.

[12] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 532–547.

[13] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3987–3995.

[14] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner, "Variational continual learning," *arXiv preprint arXiv:1710.10628*, 2017.

[15] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 6467–6476.

[16] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with a-gem," *arXiv preprint arXiv:1812.00420*, 2018.

[17] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. K. Dokania, P. H. Torr, and M. Ranzato, "On tiny episodic memories in continual learning," *arXiv preprint arXiv:1902.10486*, 2019.

[18] P. Buzzega, M. Boschini, A. Porrello, D. Abati, and S. Calderara, "Dark experience for general continual learning: a strong, simple baseline," *arXiv preprint arXiv:2004.07211*, 2020.

[19] E. Arani, F. Sarfraz, and B. Zonooz, "Learning fast, learning slow: A general continual learning method based on complementary learning system," *arXiv preprint arXiv:2201.12604*, 2022.

[20] A. Ruiz and J. Verbeek, "Anytime inference with distilled hierarchical neural ensembles," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, 2021, pp. 9463–9471.

[21] G. Huang, D. Chen, T. Li, F. Wu, L. Van Der Maaten, and K. Q. Weinberger, "Multi-scale dense networks for resource efficient image classification," *arXiv preprint arXiv:1703.09844*, 2017.

[22] H. Li, H. Zhang, X. Qi, R. Yang, and G. Huang, "Improved techniques for training adaptive deep networks," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1891–1900.

[23] L. Yang, Y. Han, X. Chen, S. Song, J. Dai, and G. Huang, "Resolution adaptive networks for efficient inference," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2369–2378.

[24] C. Zhang, M. Ren, and R. Urtasun, "Graph hypernetworks for neural architecture search," *arXiv preprint arXiv:1810.05749*, 2018.

[25] H. Koh, D. Kim, J.-W. Ha, and J. Choi, "Online continual learning on class incremental blurry task configuration with anytime inference," *arXiv preprint arXiv:2110.10031*, 2021.

[26] V. Lomonaco and D. Maltoni, "Core50: a new dataset and benchmark for continuous object recognition," in *Conference on Robot Learning*. PMLR, 2017, pp. 17–26.

[27] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in neural information processing systems*, vol. 30, 2017.

[28] Q. Cai, Y. Pan, C.-W. Ngo, X. Tian, L. Duan, and T. Yao, "Exploring object relation in mean teacher for cross-domain detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 457–11 466.

[29] J. Deng, W. Li, Y. Chen, and L. Duan, "Unbiased mean teacher for cross-domain object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4091–4101.

[30] Z. Chen, L. Zhu, L. Wan, S. Wang, W. Feng, and P.-A. Heng, "A multi-task mean teacher for semi-supervised shadow detection," in *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, 2020, pp. 5611–5620.

[31] R. F. Thompson and D. J. Krupa, "Organization of memory traces in the mammalian brain," *Annual review of neuroscience*, vol. 17, no. 1, pp. 519–549, 1994.

[32] D. J. Linden, "Long-term synaptic depression in the mammalian brain," *Neuron*, vol. 12, no. 3, pp. 457–472, 1994.

[33] A. Holtmaat and K. Svoboda, "Experience-dependent structural synaptic plasticity in the mammalian brain," *Nature Reviews Neuroscience*, vol. 10, no. 9, pp. 647–658, 2009.

[34] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[35] L. Zhang, J. Song, A. Gao, J. Chen, C. Bao, and K. Ma, "Be your own teacher: Improve the performance of convolutional neural networks via self distillation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3713–3722.

[36] R. Aljundi, P. Chakravarty, and T. Tuytelaars, "Expert gate: Lifelong learning with a network of experts," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3366–3375.

[37] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," in *Advances in Neural Information Processing Systems*, 2017, pp. 2990–2999.

[38] R. Aljundi, M. Rohrbach, and T. Tuytelaars, "Selfless sequential learning," *arXiv preprint arXiv:1806.05421*, 2018.

[39] A. Rios and L. Itti, "Closed-loop memory gan for continual learning," *arXiv preprint arXiv:1811.01146*, 2018.

[40] E. Belouadah and A. Popescu, "Il2m: Class incremental learning with dual memory," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 583–592.

[41] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 831–839.

[42] X. Tao, X. Chang, X. Hong, X. Wei, and Y. Gong, "Topology-preserving class-incremental learning," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIX 16*. Springer, 2020, pp. 254–270.

[43] J. Yoon, E. Yang, J. Lee, and S. J. Hwang, "Lifelong learning with dynamically expandable networks," *arXiv preprint arXiv:1708.01547*, 2017.

[44] V. K. Verma, K. J. Liang, N. Mehta, P. Rai, and L. Carin, "Efficient feature transformations for discriminative and generative continual learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 865–13 875.

[45] R. Aljundi, M. Lin, B. Goujaud, and Y. Bengio, "Gradient based sample selection for online continual learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 11 816–11 825.

[46] R. Aljundi, L. Caccia, E. Belilovsky, M. Caccia, M. Lin, L. Charlin, and T. Tuytelaars, "Online continual learning with maximally interfered retrieval," *arXiv preprint arXiv:1908.04742*, 2019.

[47] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[48] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[49] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[50] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgetting in gradient-based neural networks," *arXiv preprint arXiv:1312.6211*, 2013.

[51] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of learning and motivation*. Elsevier, 1989, vol. 24, pp. 109–165.

[52] J. S. Vitter, "Random sampling with a reservoir," *ACM Transactions on Mathematical Software (TOMS)*, vol. 11, no. 1, pp. 37–57, 1985.

[53] Z. Allen-Zhu and Y. Li, "Towards understanding ensemble, knowledge distillation and self-distillation in deep learning," *arXiv preprint arXiv:2012.09816*, 2020.

[54] H. Mobahi, M. Farajtabar, and P. Bartlett, "Self-distillation amplifies regularization in hilbert space," *Advances in Neural Information Processing Systems*, vol. 33, pp. 3351–3361, 2020.

[55] L. Zhang, C. Bao, and K. Ma, "Self-distillation: Towards efficient and compact neural networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 8, pp. 4388–4403, 2021.

[56] S. Fanello, C. Ciliberto, M. Santoro, L. Natale, G. Metta, L. Rosasco, and F. Odone, "icub world: Friendly robots help building good vision data-sets," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2013, pp. 700–705.

[57] G. Pasquale, C. Ciliberto, F. Odone, L. Rosasco, and L. Natale, "Teaching icub to recognize objects using deep convolutional neural networks," in *Machine Learning for Interactive Systems*. PMLR, 2015, pp. 21–25.

[58] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[59] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in neural information processing systems*, 2019, pp. 8026–8037.

[60] G. Gupta, K. Yadav, and L. Paull, "Look-ahead meta learning for continual learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 11 588–11 598, 2020.

[61] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 1126–1135.