MetaCLBench: Meta Continual Learning Benchmark on Resource-Constrained Edge Devices

Sijia Li[1*] , **Young D. Kwon**[2,3*] , **Lik-Hang Lee**[4] and **Pan Hui**[1]
[1]HKUST
[2]University of Cambridge
[3]Samsung AI Center, Cambridge
[4]The Hong Kong Polytechnic University
sli308@connect.hkust-gz.edu.cn, ydk21@cam.ac.uk,

Abstract

Meta-Continual Learning (Meta-CL) has emerged as a promising approach to minimize manual labeling efforts and system resource requirements by enabling Continual Learning (CL) with limited labeled samples. However, while existing methods have shown success in image-based tasks, their effectiveness remains unexplored for sequential time-series data from sensor systems, particularly audio inputs. To address this gap, we conduct a comprehensive benchmark study evaluating six representative Meta-CL approaches using three network architectures on five datasets from both image and audio modalities. We develop *MetaCLBench*, an end-to-end Meta-CL benchmark framework for edge devices to evaluate system overheads and investigate trade-offs among performance, computational costs, and memory requirements across various Meta-CL methods. Our results reveal that while many Meta-CL methods enable to learn new classes for both image and audio modalities, they impose significant computational and memory costs on edge devices. Also, we find that pre-training and meta-training procedures based on source data before deployment improve Meta-CL performance. Finally, to facilitate further research, we provide practical guidelines for researchers and machine learning practitioners implementing Meta-CL on resource-constrained environments and make our benchmark framework and tools publicly available, enabling fair evaluation across both accuracy and system-level metrics.

1 Introduction

With the popularization of artificial intelligence, the demand for smart home technologies and voiceprint recognition is increasing. Consequently, vision and audio-based applications such as image classification [Mai *et al.*, 2022], keyword spotting (KWS) [Michieli *et al.*, 2023], and environmental sound classification (ESC) [Chen *et al.*, 2022] have garnered significant attention. However, these applications face a critical challenge: deployed models must continually learn and update to accommodate new data and vocabulary as users' linguistic patterns evolve with their environments [Marco *et al.*, 2024]. On resource-constrained edge devices, this adaptation process often leads to the Catastrophic Forgetting (CF) problem [McCloskey and Cohen, 1989], where learning new tasks (*e.g.,* words) compromises the model's ability to recognize previously learned data.

Many studies have proposed addressing CF through Continual Learning (CL) methods, which enables learning of new tasks with less forgetting [Shin *et al.*, 2017; Chaudhry *et al.*, 2019]. A multitude of CL methods have been proposed to mitigate CF, including rehearsal-based approaches [Chauhan *et al.*, 2020; Pellegrini *et al.*, 2020; Rebuffi *et al.*, 2017], regularization-based methods [Kirkpatrick *et al.*, 2017; Zenke *et al.*, 2017], and parameter isolation-based methods [Hung *et al.*, 2019; Rusu *et al.*, 2016; Yoon *et al.*, 2017]. While rehearsal-based approaches have demonstrated superior accuracy in preventing forgetting [Kwon *et al.*, 2021a], they demand substantial labeled data [Parisi *et al.*, 2019] and pose significant memory and computational overheads for resource-constrained devices.

To address these limitations, Meta-Continual Learning (Meta-CL) has been introduced. It integrates few-shot learning into CL [Javed and White, 2019; Jerfel *et al.*, 2019] to minimize the burden of manual labeling of conventional CL methods and optimizes resource utilization through algorithm-system co-design [Kwon *et al.*, 2024a]. Despite the benefits of Meta-CL, there are still notable limitations that warrant further investigation. First, the effectiveness of the existing methods, which are typically used for tasks involving images, has not been investigated fully with sequential time series data provided by auditory sensor systems where the modality of the data is significantly different from images [Purwins *et al.*, 2019]. Second, the datasets tested have been relatively simple and limited in its variety, consisting of a few image/audio or specially crafted datasets that lack generalizability. Third, many existing studies [Javed and White, 2019; Beaulieu *et al.*, 2020; Lee *et al.*, 2021] primarily focus on accuracy, neglecting crucial system requirements for edge devices, potentially limiting their practical applicability.

In this paper, we address these gaps through a comprehensive benchmark study of six representative Meta-CL methods across three network architectures and five diverse datasets spanning image and audio modalities. The employed methods

[*]Co-first authors

include regularization-based approaches ((1) Online aware Meta-Learning (OML) [Javed and White, 2019], (2) A Neuromodulated Meta-Learning Algorithm (ANML) [Beaulieu *et al.*, 2020], (3) OML with Attentive Independent Mechanisms (OML+AIM), and (4) ANML with AIM (ANML+AIM) [Lee *et al.*, 2021]) and rehearsal-based approaches ((5) Latent OML and (6) LifeLearner [Kwon *et al.*, 2024a]).

While these methods have been largely evaluated in computer vision applications utilizing Convolutional Neural Network (CNN) architectures, their efficacy in processing audio data and other architectures remains under-explored. Hence, we adopted diverse audio datasets such as UrbanSound8K [Salamon *et al.*, 2014] and ESC50 [Piczak, 2015] datasets for Environmental Sound Classification and the GSCv2 [Warden, 2018] dataset for keyword spotting (KWS). Furthermore, we incorporated two commonly used image datasets such as MiniImageNet [Vinyals *et al.*, 2016] and CIFAR-100 [Krizhevsky *et al.*, 2009] to compare the effectiveness of Meta-CL on image datasets versus audio datasets.

Previous studies [Beaulieu *et al.*, 2020; Lee *et al.*, 2021; Kwon *et al.*, 2024a] have exclusively evaluated simple three-layer CNN architectures for Meta-CL. To bridge this research gap, we incorporated MobileNet-based architecture, YAMNet [Howard *et al.*, 2017], capable of recognizing audio events. We also employed Vision Transformers (ViT) architecture [Dosovitskiy *et al.*, 2020] for images to conduct a comparative study.

Overall, we make the following key contributions.

- **Comprehensive Evaluation Framework** (Section 3): We developed *MetaCLBench*, an end-to-end benchmark framework that incorporates diverse audio (GSCV2, UrbanSound8K, ESC50) and visual datasets (CIFAR100, MiniImageNet), enabling thorough evaluation across different data modalities. The framework assesses both accuracy and system-level metrics, including computational efficiency, memory and energy consumption. Also, our evaluation covers multiple architectures of varying complexity, offering a robust understanding of Meta-CL performance across different computational scenarios.

- **Empirical Insights** (Section 4): We present comprehensive findings demonstrating the effectiveness of Meta-CL methods across both image and audio domains. Our analysis quantifies the computational and memory demands on edge devices (Section 4.1), while revealing that pre-training and meta-training procedures significantly enhance Meta-CL performance during deployment (Section 4.2). Notably, we find that well-tuned basic 3-layer CNN architecture with proper pre-training can outperform more sophisticated models such as ViT and YAMNet (Table 1) under difficult Meta-CL setups, challenging conventional assumptions about model complexity.

- **Open Resources and Practical Guidelines** (Section 5): By implementing *MetaCLBench* on edge devices, we provide realistic insights and practical guidelines specifically tailored for resource-constrained devices. Furthermore, we plan to release our benchmark framework and evaluation tools[1] to enable fair and comprehensive assessment of both accuracy and system-level performance metrics. This will facilitate reproducible research and accelerate the development of practical Meta-CL solutions for real-world applications.

2 Related Work

Continual Learning: CL aims to assimilate new knowledge in dynamically evolving input domains through progressive acquisition [Mai *et al.*, 2022; Wang *et al.*, 2024]. When algorithms are trained sequentially, the new learning will interfere with previous established weights, leading to CF of the previous learning [McCloskey and Cohen, 1989]. Current CL approaches to the CF challenge can be divided into two categories based on task information management. The first approach incorporates regularization terms into the loss function to facilitate knowledge of old and new models, thereby preserving previously learned knowledge [Rannen *et al.*, 2017; Li and Hoiem, 2018; Rodríguez *et al.*, 2021; Ahn *et al.*, 2019]. The second approach involves either original or pseudo-samples from previous tasks during new task training [Shin *et al.*, 2017; Rolnick *et al.*, 2019; Chaudhry *et al.*, 2019].

Meta Continual Learning: Meta-CL updates the model in the outer loop using learned random samples and optimizes it in the inner loop with a few samples before passing it to the outer loop [Jerfel *et al.*, 2019]. Previous research on Meta-CL has primarily concentrated on image recognition applications. Online-aware Meta-Learning (OML) [Javed and White, 2019] and A Neuromodulated Meta-Learning (ANML) [Beaulieu *et al.*, 2020] both demonstrated high CL performance on the Omniglot dataset [Lake *et al.*, 2015]. Based on these approaches, state-of-the-art (SOTA) Meta-CL incorporated the Attentive Independent Mechanisms (AIM) module, which can independently acquire new knowledge to enhance model performance [Lee *et al.*, 2021]. However, due to its high processing cost, using Meta-CL is challenging on low-end edge devices [Kwon *et al.*, 2024a]. LifeLearner addresses these limitations by integrating meta-learning, rehearsal mechanisms, and resource-efficient compression, making it suitable for deploying Meta-CL applications on resource-constrained devices [Kwon *et al.*, 2024a].

Continual Learning for Image and Audio Applications: CL has remarkable capabilities in sensing applications, including action recognition, audio sensing, and speech recognition [Jha *et al.*, 2021; Kwon *et al.*, 2021a]. Its ability to update models based on continuous data streams makes it particularly valuable for mobile devices processing speech and image data. Researchers have successfully applied CL to speech and emotion recognition [Thuseethan *et al.*, 2021]. CL systems integrated into edge devices with audio and microphones continually learn from ambient human voices and environmental sounds to improve recognition accuracy [Kwon *et al.*, 2021b]. In image recognition, CL facilitates applications such as food recognition [He and Zhu, 2021], where systems must continuously learn from new instances to enhance recognition accuracy.

[1]https://github.com/theyoungkwon/MetaCLBench

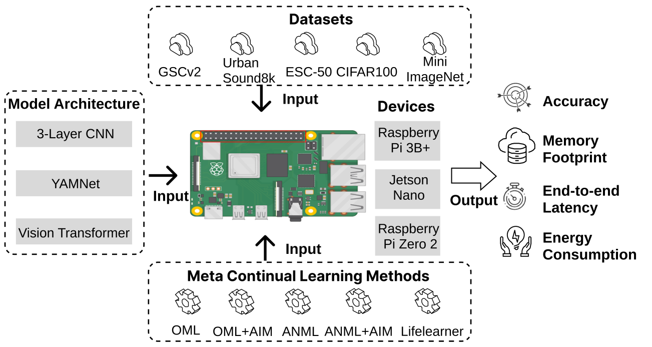


Figure 1: The framework overview. Testing trade-offs between performance and system resources across three devices with five datasets and six Meta-CL methods using three model architectures

In contrast to these real-world applications, our study explores CL's capability to optimize the trade-off between performance and system resources on resource-constrained edge devices. We comprehensively evaluate five Meta-CL methods across image and audio domains using various architectures ranging from 3-layer CNN, YAMNet, to ViT, accounting for the unique characteristics of each modality and architecture.

3 Benchmark Framework

3.1 Continual Learning Task

Problem Formulation: CL addresses the challenge of learning new tasks without forgetting previously learned ones. Given a sequence of tasks S, the objective is to maintain robust performance across all tasks $S_{1....n}$. Our evaluation framework encompasses $(T(Task), S(Sequence), D(Datasets))$, where each task represents a distinct sound class. Each subsequent task Sn is derived from the historical training records of the most recent sets $S_{n-1}, S_{n-2}...S_{n-m}$ of the employed audio and image datasets.

Meta-Learning is conducted in both the inner and outer loops, enabling the training of each inner loop round through the outer loop. Each inner loop iteration is performing a complete learning process, evaluating both the acquisition of new tasks and the retention of previously learned ones. The meta-loss gradient (comprising newly learned errors and errors from random sampling of the dataset) propagates back to update the initial parameters. This process continues iteratively as new tasks are incorporated. By constraining each inner loop to a single new task while continuously undergoes meta-training, it mitigates CF and optimizing computational efficiency.

Our Framework Overview: Figure 1 shows our experimental framework, *MetaCLBench*, designed to evaluate the performance and resource trade-offs across multiple dimensions: three devices, five datasets, six Meta-CL methods, and three architectures. *MetaCLBench* extends Meta-CL by implementing a comprehensive evaluation system that spans multiple modalities and practical considerations. By combining both visual and audio datasets, we provide thorough insights into Meta-CL performance across diverse data types. We conduct a detailed system-level analysis of computational efficiency, memory usage, and energy consumption, particularly on resource-constrained edge devices for real-world applications. Overall, our framework evaluates six established Meta-CL methods across three network architectures of varying complexity, creating a robust benchmark framework that assesses both theoretical capabilities and practical constraints. We now explain each of the components of our framework in the following subsections.

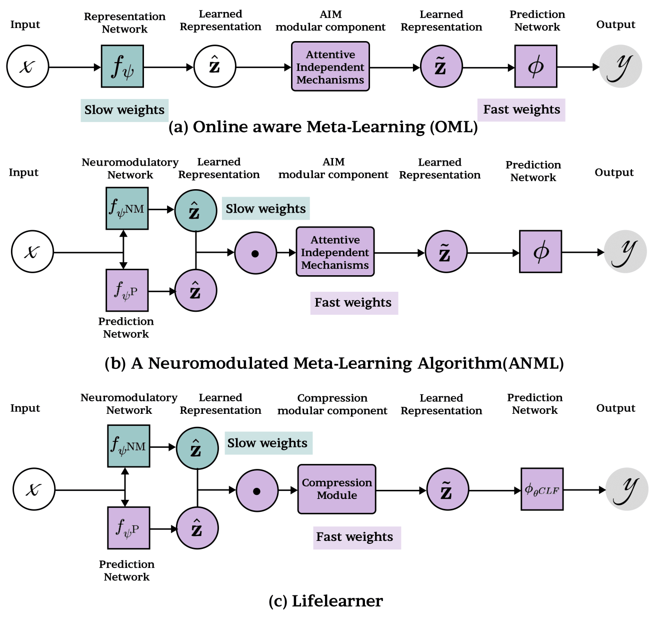


Figure 2: The illustration of the Meta-CL methods evaluated in our benchmark framework.

3.2 Meta-CL Methods

Figure 2 presents six distinct Meta-CL methods: Online aware Meta-Learning (OML), OML with Attentive Independent Mechanisms (OML+AIM), A Neuromodulated Meta-Learning Algorithm (ANML), ANML+AIM, and LifeLearner. The following sections detail each method.

OML [Javed and White, 2019] continuously optimizes representations during online updates to quickly acquire predictions for new tasks, facilitating seamless and efficient CL. This approach uses CF as signals to justify the use of storage space for accommodating new tasks. The OML can accomplish CL in more than 200 consecutive classes. It is an online updating method that constantly learns representations and often outperforms rehearsal-based continuous learning methods.

ANML [Beaulieu *et al.*, 2020]: This is an approach inspired by the neuromodulation process in the brain. This method accomplishes selective activation (forward pass) and selective plasticity (backward pass) in DNNs by meta-learning through a CL process within the prediction network ($f_{\varphi P}$). Optimizing activation timing by controlling the activation of a predictive model based on input conditions enhances continual learning. It is often reported to be superior to OML, as it can be optimized to reduce both forward and backward interference.

OML+AIM [Lee *et al.*, 2021]: This is a Meta-CL approach that integrates Attentive Independent Mechanisms (AIM), which is a modular component into OML framework. AIM increases the speed of learning a new task by selecting out the mechanism that best explains the representation of the inner loop to activate.

ANML+AIM [Lee *et al.*, 2021]: This is a Meta-CL approach that incorporates AIM into ANML framework. The accuracy of SOTA Meta-CL continues to improve beyond the performance of the previous three approaches by integrating AIM into the existing CL framework.

LifeLearner [Kwon *et al.*, 2024a]: This module is based on ANML and combines both lossy and lossless compression to achieve high compression rates and minimize footprint. The LifeLearner approach not only achieves optimal CL efficiency but also significantly reduces energy consumption and latency compared to the SOTA. And it achieves efficient learning even on resource-constrained edge devices.

Latent OML: This module is based on OML and incorporates the rehearsal strategy from LifeLearner to achieve high Meta-CL accuracy. It is used to evaluate advanced architectures (YAMNet and ViT).

3.3 Datasets

Google Speech Commands V2 (GSCV2) [Warden, 2018] is a massive database of one-second voice clips of 30 short words consisting of a solitary spoken English word or ambient noise divided into 35 classes with 105,829 clips. These clips are derived from a limited number of commands and are spoken by various speakers. Each class has 2,424 input data for training and 314 for testing, with 25 classes allocated for meta-training and the remaining 10 for meta-testing. Then, up to 30 samples are provided for each class during meta-testing.

Urbansound8k is a comprehensive collection for environment sound classification (ESC) applications comprising 8,732 labeled sounds that are no more than four seconds [Salamon *et al.*, 2014]. This single-labeled dataset is categorized into ten classes from the taxonomy in real-world settings, including children playing sound, street music, gunshots, etc. Referring to previous studies [Su *et al.*, 2019], indicate that four features in each audio clip were extracted and resampled to 22 kHz. Among the ten classes, half of them are allocated for meta-training, while the other half are specifically assigned for meta-testing. Up to 30 samples are given for each class during the meta-testing. Based on the first three seconds of each audio segment, the input to the audio is 128 *(number of frames)* $\times$ 85 *(size of the set of four features)*.

ESC50 is a smaller volume collection consisting of 2,000 environment sounds in 5-second segments [Piczak, 2015]. This single-labeled dataset is categorized into 50 classes and divided into five groups: animals, natural soundscapes, human sound without speech, domestic sounds and urban noises. Out of the 50 classes. 30 are designated for meta-training and the remaining 20 for meta-testing. Then, up to 30 samples are given for each class during meta-testing. As part of the preprocessing procedure, we resampled the ESC50 samples to 32kHz and generated an input with dimensions of 157 *(number of frames)* $\times$ 64 *(features)* [Chen *et al.*, 2022; Pons *et al.*, 2019].

CIFAR-100 consists of 60,000 images distributed across 100 classes and is widely used for training and assessing various machine learning algorithms in tasks related to image classification [Krizhevsky *et al.*, 2009]. Each class contains 500 photographs for training and 100 images for testing. Of the 100 classes, 70 are designated for meta-training and 30 for meta-testing. In both phases, 30 random training images per class are utilized. For the meta-testing phase, 900 samples are utilized during meta-testing to carry out CL.

MiniImageNet is excerpted from the Imagenet dataset and is utilized in few-shot learning research [Vinyals *et al.*, 2016]. Based on the previous research, the MiniImageNet dataset is composed of 64 classes designated for meta-training and 20 for meta-testing. Each class offers 540 images for training and 60 for testing. Moreover, a total of 600 samples are available during the meta-testing phase.

3.4 Model Architecture

We augmented the 3-layer CNN architecture previously employed in CL studies by incorporating the YAMNet network architecture with pre-training. In addition to assessing YAMNet as a model for the speech domain with pre-training, we also evaluated the ViT, a transformer model that integrates a pre-trained model used for image classification by segmenting the image to a fixed size for input to the transformer encoder.

3-layer Architecture: The methods depicted in Fig. 2 consist of feature extractors and classifiers. The feature extractor transmits the learned features to the AIM and the compression modules, followed by classification. The $f_\varphi W$ added to the AIM module components can be used to minimize forgetting and learn new categories efficiently. In the 3-layer architecture, the feature extractor for OML and OML+AIM incorporates f_φ, which is a 6-layer CNN with 112 channels, and the classifier has two fully-connected layers. And the LifeLearner and ANML+AIM share the same natural structure. The feature extractor includes a neuromodulatory network ($f_{\varphi NM}$) and a prediction network ($f_{\varphi P}$), which is a 3-layer CNN with 112 channels. The classifier consists of a single fully-connected layer with 256 channels.

YAMNet [Howard *et al.*, 2017] employs the MobileNetV1, which features a depthwise separable convolution architecture designed to decrease model size and latency and a full convolution on the first layer. The depthwise separable convolution comprises a deep convolution responsible for filtering and a pointwise convolution used 1×1 convolution responsible for combining features, effectively mixing information from all output channels in the deep convolution step. MobileNetV1 consists of 28 layers, with each layer being downsampled through strided convolution. Additionally, MobileNets introduces two hyperparameters, a width multiplier and a resolution multiplier, which facilitate the balance latency and accuracy.

ViT [Dosovitskiy *et al.*, 2020] embodies a hybrid architecture. Since ViT only has a Multilayer Perceptron (MLP) layer with localization and translation equivalence, the image-specific generalization bias of ViT is much smaller than that of CNN. The same five methodologies deployed in the 3-layer architecture have been adapted for use with ViT. ViT begins by partitioning an image into patches; these patches, extracted from the CNN feature maps, serve as alternative input sequences for the model.

3.5 System Implementation

The implementation of our benchmark framework, *MetaCLBench*, consists of two stages. We developed the first stage, pre-training and meta-training, of Meta CL methods on a Linux server to initialize and optimize neural weights in order to enable fast adaptation during deployment scenarios with a few samples. After that, we implemented the second stage, meta-testing (i.e., actual deployment setup), on our target devices: (1) embedded and mobile systems such as Jetson Nano and Raspberry Pi 3B+ and (2) a severely resource-constrained IoT device like Raspberry Pi Zero 2. In addition, for methods that utilize hardware-friendly optimization (e.g., LifeLearner), we also adopt their optimization techniques in our framework. Specifically, *MetaCLBench* incorporates hardware-friendly 8-bit integer arithmetic [Jacob *et al.*, 2018] which reduces the precision of weights/activations of the model from 32-bit floats to 8-bit integers, increasing the computation throughput and minimizing latency and energy. *MetaCLBench* uses the scalar quantization scheme [Jacob *et al.*, 2018] to minimize the information loss in quantization. Also, the QNNPACK backend engine is used to execute the quantized model on the embedded devices.

Embedded Devices: *MetaCLBench* employs three diverse edge devices: (1) Jetson Nano, (2) Raspberry Pi 3B+, and (3) Raspberry Pi Zero 2 which are equipped with a quad-core ARM processor and 4 GB, 1 GB, 0.5 GB of RAM, respectively. In addition, it should be noted that the available memory on the Jetson Nano, Raspberry Pi 3B+, and Pi Zero 2 during idle time is about 1.7 GB, 600 MB, and 250 MB. This is because some of the memory is already being used by background processes, concurrent apps, and the operating system. Hence, we allocate an additional 1GB of swap space, enabling the execution of memory-intensive Meta-CL methods such as ANML+AIM and OML+AIM. Our software stack includes Faiss [Johnson *et al.*, 2019] and PyTorch 1.13 for meta-training and meta-testing stages.

3.6 Experimental Setup

Training Detail: We adopt a pre-training procedure, similar to prior works [Hu *et al.*, 2022; Yosinski *et al.*, 2014]. For the advanced model architectures such as ViT and YAMNet, there exist pre-trained model weights, thus we utilize them directly similar to [Hu *et al.*, 2022]. Then, for the CNN architecture, we pre-train the CNN model with a sufficient number of epochs (i.e., until the validation loss converges), which is consistent with conventional transfer learning for DNNs [Yosinski *et al.*, 2014]. Consistent with prior meta-training research [Lee *et al.*, 2021; Kwon *et al.*, 2024a], we employed a batch size of 1 for the inner-loop updates and 64 for the outer-loop updates across 20,000 steps to ensure the accuracy of our results. To obtain a meta-trained model with the highest possible validation set accuracy, we tested with various learning rates required for both the inner and outer loops. Consequently, we established the learning rate for the inner loop (α) was set to 0.001, and the learning rate for the outer loop (β) was also set to 0.001 for all datasets. In the meta-testing phase, we assessed ten distinct learning rates for each method and reported the best accuracy. To evaluate the precision of rehearsal-based methods, we experimented with batch sizes of 8 and 16 and observed a minimal performance difference depending on the batch sizes. We selected a batch size of 8 as it requires less memory than larger batch sizes.

We chose to use ANML with CNN architectures and OML with more advanced models like ViT and YAMNet. Our preliminary tests revealed minimal differences between ANML and OML within the same architecture, with ANML slightly outperforming OML in terms of accuracy. Due to this marginal difference, we focused on ANML for CNNs to streamline the analysis. For more complex architectures like ViT and YAMNet, we opted for OML, as ANML introduces additional computational overhead due to an extra layer. Our study concentrated on the performance of methods within specific architectures rather than comparing ANML and OML across all architectures. This streamlined approach allowed us to present a clearer analysis without redundant testing across combinations.

Evaluation Metrics: Following [Beaulieu *et al.*, 2020], this study focuses on the accuracy of unseen samples across new categories in CL. We measure the end-to-end latency, energy consumption, and peak memory by deploying various Meta-CL methods using *MetaCLBench* on edge devices: Jetson Nano, Raspberry Pi 3B+, Raspberry Pi Zero 2. Specifically, we measure system performance metrics by continually learning all given classes for deployed DNNs on embedded devices. Peak memory includes: (1) model memory for updated weights, (2) optimizer memory for gradients and momentum, (3) activations memory for intermediate outputs during weight updates, and (4) rehearsal samples. To measure end-to-end latency and energy consumption, we account for the time and energy required to: (1) load the model and (2) perform Meta-CL using all available samples (*e.g.,* 30) across all classes (*e.g.,* 30 for CIFAR-100 and 10 for GSCv2). Energy consumption is measured by monitoring the power consumption of the edge device using a YOTINO USB power meter. We derive the energy consumption using the equation (Energy = Power x Time).

4 Results & Findings

This section presents comprehensive evaluation results and findings from MetaCLBench. Due to page constraints, we present results from two datasets in the main text, with complete results available in the Appendix.

4.1 Main Results

Previous research [Kwon *et al.*, 2024a; Holla *et al.*, 2020], has shown that OML algorithms achieve lower accuracy than ANML algorithms when using 3-layer CNN architectures. Therefore, we focused our experimental investigation exclusively on ANML for CNN-based evaluations. For advanced models like YAMNet and ViT, we prioritized OML evaluation due to their inherent pre-training components, which eliminate the need for additional pre-training through ANML or the AIM module—additions that neither significantly improve accuracy nor justify their computational cost.

Accuracy. We evaluated the test accuracy of six Meta-CL approaches across five datasets against a baseline model. Oracle models (Oracle ANML for CNN, Oracle OML for YAMNet and ViT) establish the upper accuracy bounds and serve as

Table 1: Performance and computational costs, memory footprint of six representative Meta-CL methods using three network architectures on five datasets of image and audio domains. OOM indicates an out-of-memory issue.

		CNN			
Dataset	**Method**	**Accuracy**	**Memory**	**Latency**	**Energy**
Mini ImageNet	Pre-trained	0.258	474.5MB	1,198s	5.5KJ
	ANML	0.327	474.5MB	1,152s	5.3KJ
	ANML+AIM	0.331	1,562 MB	OOM	OOM
	OML+AIM	0.187	1,051 MB	1,434s	6.5KJ
	Raw ANML	0.429	897.1 MB	185,610s	810KJ
	Oracle ANML	0.438	475.0 MB	3.414s	15.7KJ
	LifeLearner	0.433	136.7 MB	1,236s	6.17kJ
Urban Sound8K	Pre-trained	0.182	1,382 MB	302s	1.6KJ
	ANML	0.596	1,382 MB	305s	1.6KJ
	ANML+AIM	0.439	2,593 MB	OOM	OOM
	OML+AIM	0.385	2,648 MB	OOM	OOM
	Raw ANML	0.667	1,456 MB	60,120s	279KJ
	Oracle ANML	0.710	1,384 MB	916s	4.5KJ
	LifeLearner	0.650	496 MB	320s	1.7KJ

		ViT			
Dataset	**Method**	**Accuracy**	**Memory**	**Latency**	**Energy**
Mini ImageNet	Pre-trained	0.234	336.6 MB	2,056s	11.2KJ
	OML	0.255	336.6 MB	2,203s	11.4KJ
	Raw OML	0.454	1,334 MB	301,456s	1150KJ
	Oracle OML	0.512	337.7 MB	6,301s	31KJ
	Latent OML	0.376	336.6 MB	2,250s	13KJ
		YAMNet			
Urban Sound8K	Pre-trained	0.186	39.2 MB	912s	5.4KJ
	OML	0.262	39.2 MB	910s	5.4KJ
	Raw OML	0.595	139.3 MB	120,965s	550KJ
	Oracle OML	0.729	42.4 MB	2,501s	21.1KJ
	Latent OML	0.442	39.2 MB	910s	1.6KJ

benchmarks, as they are trained on all classes simultaneously in an i.i.d. manner, following traditional offline DNN training. By comparing Meta-CL methods against Oracle models, we demonstrate the trade-offs between Meta-CL approaches and ideal scenarios where all data is available initially, highlighting our methods' efficiency under practical constraints. The results in Table 1 show that models using only pre-trained weights ("Pre-trained") achieve lower accuracy, averaging 22.18% and 13.8% across the datasets for Convolutional Neural Networks (CNNs) and advanced architectures(YAMNet and ViT). This result indicates that traditional transfer learning fails to address few-shot learning challenges. In CNN architecture, baseline ANML improved upon the pre-trained-only approach with a 16.7% average accuracy gain. The AIM enhancement further increased accuracy by 2.54%. However, these improvements remained below Oracle and LifeLearner methods by 15.92%. The constrained sample size during the meta-testing phase poses a challenge to attaining high accuracy, even considering the theoretical upper limit.

Peak Memory Footprint. We conducted measurements to ascertain the maximum RAM requirements forbackpropagation execution and rehearsal samples storage. The backpropagation memory is divided into three parts: (1) model memory, which is responsible for storing model parameters; (2) optimizer memory, which is responsible for storing gradient and momentum vectors; (3) activation memory, which is responsible for storing intermediate activations that are reused during backpropagation. Table 1 presents the peak memory usage across various methodologies. First, we found that although AIM enhances accuracy, it increases peak usage due to additional parameters and activation storage requirements during training. The memory demands 1,562 MB for MiniImageNet and 2,648 MB for UrbanSound8K, significantly exceeding the RAM capacity of embedded devices such as the Pi 3B+ and resulting out-of-memory (OOM) problems. Consequently, it becomes challenging to utilize it in edge devices. While OML, ANML, Oracle, Pre-trained, and LifeLearner demonstrate lower memory requirements, Pre-trained and ANML are less accurate. LifeLearner emerges as the optimal solution, requiring only 136-496 MB while maintaining accuracy comparable to Oracle through high compression rates. This efficiency is achieved through latent replay, which reduces activation memory and overall memory footprint, making it particularly suitable for resource-constrained environments. In contrast, advanced architectures like ViT and YAMNet demand greater computational resources due to high-dimensional data processing and additional layer training requirements.

End-to-end Latency & Energy Consumption. Table 1 also presents the operational efficiency of Meta-CL methods implemented on Raspberry Pi 3B+, focusing on end-to-end latency and energy consumption. The end-to-end latency comprises four components: model loading (5%), inference (30%), backpropagation (61%), and data processing overhead (4%). First, we measured the end-to-end latency of CL methods on the Raspberry Pi 3B+ across all classes (30 samples per class). LifeLearner achieved fast end-to-end latency. Oracle exhibited higher training latency due to i.i.d. Training over multiple epochs until it converges. AIM's implementation of attention mechanism selection during backpropagation increased computational overhead. LifeLearner reduces training and memory latency through latent replay and model freezing. Due to the large amount of memory required by the AIM series, crashes occurred several times during the test due to memory exhaustion.

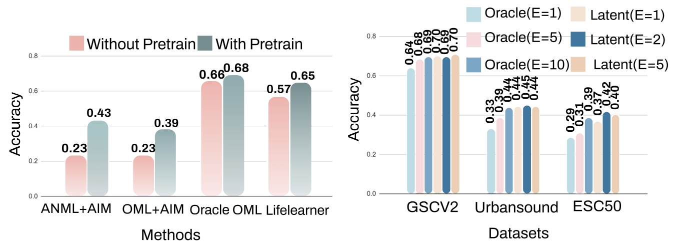
To establish generalizability, we extended our evaluation to diverse edge devices with varying computational capabilities (Jetson Nano with enhanced specifications and Pi Zero 2 with reduced capacity). The table 2 below shows the end-to-end latency and energy consumption results across these three edge devices, confirming that LifeLearner achieves a 7.5-fold reduction in latency compared to Oracle.

4.2 Analysis and Discussion

Effectiveness of Pre-training. We examine the effectiveness of pre-training on the performance of Meta-CL methods. Our evaluation across five datasets without pre-training reveals varying performance levels: while the two image datasets and GSCv2 demonstrated satisfactory accuracy, Urbansound and ESC50 showed suboptimal performance. Following the approach used in advanced models, we implemented pre-training before meta-training for each method. As shown in Figure 3a, the incorporation of pre-training significantly enhanced model performance. Notably, Meta-CL methods achieved an average performance improvement of 13.98% on the Urbansound and ESC50 datasets after pre-training. Our analysis revealed that CNNs exhibited considerable performance degradation

Table 2: End-to-end latency and energy consumption results of running different Meta-CL methods on three different edge devices

Device	Method	Latency	Energy
Raspberry Pi 3B+	Pretrained	71s	0.32kJ
	ANML	71s	0.32kJ
	Oracle ANML	570s	2.60kJ
	LifeLearner	76s	0.35kJ
Jetson Nano	Pretrained	79s	0.36kJ
	ANML	79s	0.36kJ
	Oracle ANML	633s	2.00kJ
	LifeLearner	84s	0.39kJ
Raspberry Pi Zero 2	Pretrained	74s	0.33kJ
	ANML	75s	0.33kJ
	Oracle ANML	601s	2.64kJ
	LifeLearner	80s	0.35kJ



(a) With and Without pre-training on CNN

(b) Number of replay epochs

Figure 3: The analysis of Meta-CL Methods for the audio datasets

when meta-training was conducted without pre-training. Given that advanced models inherently contain pre-trained weights, additional pre-training did not yield substantial accuracy improvements. These findings establish pre-training as a crucial step in optimizing model accuracy and demonstrate that the sequential implementation of pre-training followed by meta-training is essential for achieving superior performance across all methods.

The Number of Replay Epochs. We investigated the relationship between replay frequency and accuracy, acknowledging that increased replays correspond to higher latency and energy consumption. As illustrated in Figure 3b, Oracle OML requires more than five replays to achieve high accuracy levels, resulting in increased energy consumption during training. In contrast, Latent OML demonstrates superior efficiency, achieving comparable accuracy with only one or two replays, thus reducing system overhead. Within the same dataset, Latent OML's performance after five replays approximates that of Oracle OML after ten replays.

Multi Compression Module. In CNN architectures, ANML with AIM demonstrates substantial accuracy improvements (from 26.0% to 34.6% on CIFAR100), albeit at the cost of increased memory requirements (from 39.69MB to 1,093MB). LifeLearner presents an optimal balance between accuracy and memory utilization in resource-constrained environments. The integration of AIM, while beneficial for accuracy, introduces significant computational overhead, particularly in Raw ANML implementations (11,424s latency, 52.5KJ energy consumption on CIFAR100). Oracle ANML provides a balanced compromise between performance enhancement and resource efficiency.

Vision Transformer (ViT) and YAMNet architectures exhibit analogous performance-resource tradeoffs. ViT implementations with OML and AIM show improved accuracy but incur substantial computational costs (22,968s latency, 103KJ energy consumption on CIFAR100). Similarly, YAMNet with AIM achieves superior accuracy (from 42.9% to 71.0% on GSCv2) while requiring increased memory utilization (from 10.16MB to 608.2MB). Pretrained YAMNet models demonstrate superior resource efficiency metrics.

5 Practical Guidelines

Our extensive empirical analysis provides evidence-based guidelines for researchers and practitioners implementing Meta-CL on edge devices. Although this investigation focuses on specific Meta-CL methods and may not encompass all learning approaches, our results and findings show consistent patterns across different architectures and datasets, which lead to the following key guidelines.

Resource-Aware Method Selection. The choice of Meta-CL implementation should be guided by available system resources. For systems with adequate storage capacity, LifeLearner or Latent OML implementations provide optimal accuracy across datasets. Resource-constrained environments benefit from Raw ANML or Raw OML methods combined with hardware-friendly optimizations. In severely limited environments, efficient integration of the quantization and compression module of LifeLearner with advanced models (YAMNet and ViT) would offer a viable compromise between computational efficiency and storage requirements.

Edge Deployment Considerations. Latent methods consistently demonstrate superior performance across image and audio datasets, balancing accuracy with resource efficiency. This characteristic positions these methods as particularly advantageous for IoT applications, facilitating real-time category integration, efficient personalized processing, and privacy-preserving local computation without additional communication overhead.

6 Conclusion

In this paper, we investigate how to solve the CF problem based on five representative image and audio datasets (i.e., CIFAR100, MiniImageNet, GSCV2, UrbanSound8K, and ESC50) using three different network architectures with different complexities using six well-known Meta-CL methods. We also created a comprehensive Meta-CL benchmark framework on an edge device to assess the system overheads occurring on the device and examine the trade-offs between performance, computational costs, memory usage, and storage of different Meta-CL approaches. We first found that the combination of pre-training plus meta-training plus evaluation maximizes performance. Advanced methods (YAMNet and ViT) already include pre-trained, so the extra pre-training will not affect the results much.

References

[Ahn *et al.*, 2019] Hongjoon Ahn, Sungmin Cha, Donggyu Lee, and Taesup Moon. Uncertainty-based continual learning with adaptive regularization. *Advances in neural information processing systems*, 32, 2019.

[Arik *et al.*, 2018] Sercan Arik, Jitong Chen, Kainan Peng, Wei Ping, and Yanqi Zhou. Neural voice cloning with a few samples. *Advances in neural information processing systems*, 31, 2018.

[Beaulieu *et al.*, 2020] Shawn Beaulieu, Lapo Frati, Thomas Miconi, Joel Lehman, Kenneth O Stanley, Jeff Clune, and Nick Cheney. Learning to continually learn. *arXiv preprint arXiv:2002.09571*, 2020.

[Chaudhry *et al.*, 2019] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, P Dokania, P Torr, and M Ranzato. Continual learning with tiny episodic memories. In *Workshop on Multi-Task and Lifelong Reinforcement Learning*, 2019.

[Chauhan *et al.*, 2020] Jagmohan Chauhan, Young D Kwon, Pan Hui, and Cecilia Mascolo. Contauth: Continual learning framework for behavioral-based user authentication. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(4):1–23, 2020.

[Chauhan *et al.*, 2022] Jagmohan Chauhan, Young D Kwon, and Cecilia Mascolo. Exploring on-device learning using few shots for audio classification. In *2022 30th European Signal Processing Conference (EUSIPCO)*, pages 424–428. IEEE, 2022.

[Chen *et al.*, 2022] Ke Chen, Xingjian Du, Bilei Zhu, Zejun Ma, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Hts-at: A hierarchical token-semantic audio transformer for sound classification and detection. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 646–650. IEEE, 2022.

[Cheraghian *et al.*, 2021] Ali Cheraghian, Shafin Rahman, Pengfei Fang, Soumava Kumar Roy, Lars Petersson, and Mehrtash Harandi. Semantic-aware knowledge distillation for few-shot class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2534–2543, 2021.

[Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[Douillard *et al.*, 2020] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *Computer vision–ECCV 2020: 16th European conference, Glasgow, UK, August 23–28, 2020, proceedings, part XX 16*, pages 86–102. Springer, 2020.

[Fernando *et al.*, 2017] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

[Finn *et al.*, 2017] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[Fritzke, 1994] Bernd Fritzke. A growing neural gas network learns topologies. *Advances in neural information processing systems*, 7, 1994.

[He and Zhu, 2021] Jiangpeng He and Fengqing Zhu. Online continual learning for visual food classification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2337–2346, 2021.

[Hersche *et al.*, 2022] Michael Hersche, Geethan Karunaratne, Giovanni Cherubini, Luca Benini, Abu Sebastian, and Abbas Rahimi. Constrained few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9057–9067, 2022.

[Holla *et al.*, 2020] Nithin Holla, Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. Meta-learning with sparse experience replay for lifelong language learning. *arXiv preprint arXiv:2009.04891*, 2020.

[Hospedales *et al.*, 2021] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5149–5169, 2021.

[Hou *et al.*, 2019] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 831–839, 2019.

[Howard *et al.*, 2017] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[Hu *et al.*, 2022] Shell Xu Hu, Da Li, Jan Stühmer, Minyoung Kim, and Timothy M Hospedales. Pushing the limits of simple pipelines for few-shot learning: External data and fine-tuning make a difference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9068–9077, 2022.

[Hung *et al.*, 2019] Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. *Advances in neural information processing systems*, 32, 2019.

[Jacob *et al.*, 2018] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[Javed and White, 2019] Khurram Javed and Martha White. Meta-learning representations for continual learning. *Advances in neural information processing systems*, 32, 2019.

[Jerfel *et al.*, 2019] Ghassen Jerfel, Erin Grant, Tom Griffiths, and Katherine A Heller. Reconciling meta-learning and continual learning with online mixtures of tasks. *Advances in neural information processing systems*, 32, 2019.

[Jha *et al.*, 2021] Saurav Jha, Martin Schiemer, Franco Zambonelli, and Juan Ye. Continual learning in sensor-based human activity recognition: An empirical benchmark analysis. *Information Sciences*, 575:1–21, October 2021.

[Johnson *et al.*, 2019] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, pages 1–1, 2019.

[Keshari *et al.*, 2018] Rohit Keshari, Mayank Vatsa, Richa Singh, and Afzel Noore. Learning structure and strength of cnn filters for small sample size training. In *proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9349–9358, 2018.

[Kirkpatrick *et al.*, 2017] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[Kozerawski and Turk, 2018] Jedrzej Kozerawski and Matthew Turk. Clear: Cumulative learning for one-shot one-class image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3446–3455, 2018.

[Krizhevsky *et al.*, 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[Kwon *et al.*, 2021a] Young D Kwon, Jagmohan Chauhan, Abhishek Kumar, Pan Hui, and Cecilia Mascolo. Exploring System Performance of Continual Learning for Mobile and Embedded Sensing Applications. In *ACM/IEEE Symposium on Edge Computing*. Association for Computing Machinery (ACM), 2021.

[Kwon *et al.*, 2021b] Young D. Kwon, Jagmohan Chauhan, and Cecilia Mascolo. FastICARL: Fast Incremental Classifier and Representation Learning with Efficient Budget Allocation in Audio Sensing Applications. In *Proc. Interspeech 2021*, pages 356–360, 2021.

[Kwon *et al.*, 2024a] Young D. Kwon, Jagmohan Chauhan, Hong Jia, Stylianos I. Venieris, and Cecilia Mascolo. LifeLearner: Hardware-Aware Meta Continual Learning System for Embedded Computing Platforms. In *Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems*, SenSys '23, page 138–151, New York, NY, USA, 2024. Association for Computing Machinery.

[Kwon *et al.*, 2024b] Young D. Kwon, Rui Li, Stylianos Venieris, Jagmohan Chauhan, Nicholas Donald Lane, and Cecilia Mascolo. TinyTrain: Resource-Aware Task-Adaptive Sparse Training of DNNs at the Data-Scarce Edge. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 25812–25843. PMLR, 21–27 Jul 2024.

[Lake *et al.*, 2015] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[Lee *et al.*, 2021] Eugene Lee, Cheng-Han Huang, and Chen-Yi Lee. Few-shot and continual learning with attentive independent mechanisms. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9455–9464, 2021.

[Li and Hoiem, 2017] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[Li and Hoiem, 2018] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018.

[Lin *et al.*, 2022] Ji Lin, Ligeng Zhu, Wei-Ming Chen, Wei-Chen Wang, Chuang Gan, and Song Han. On-device training under 256kb memory. *Advances in Neural Information Processing Systems*, 35:22941–22954, 2022.

[Mai *et al.*, 2022] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 469:28–51, 2022.

[Mallya and Lazebnik, 2018] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7765–7773, 2018.

[Marco *et al.*, 2024] N. Di Marco, Edoardo Loru, Anita Bonetti, Alessandra Olga Grazia Serra, Matteo Cinelli, and Walter Quattrociocchi. Patterns of linguistic simplification on social media platforms over time. *Proceedings of the National Academy of Sciences*, 121(50):e2412105121, 2024.

[McCloskey and Cohen, 1989] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press, 1989.

[Michieli *et al.*, 2023] Umberto Michieli, Pablo Peso Parada, and Mete Ozay. Online Continual Learning in Keyword Spotting for Low-Resource Devices via Pooling High-Order Temporal Statistics. In *Proc. INTERSPEECH 2023*, pages 1628–1632, 2023.

[Parisi *et al.*, 2019] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca

Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[Pellegrini *et al.*, 2020] Lorenzo Pellegrini, Gabriele Graffieti, Vincenzo Lomonaco, and Davide Maltoni. Latent replay for real-time continual learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10203–10209. IEEE, 2020.

[Piczak, 2015] Karol J Piczak. Esc: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1015–1018, 2015.

[Pons *et al.*, 2019] Jordi Pons, Joan Serrà, and Xavier Serra. Training neural audio classifiers with few data. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 16–20. IEEE, 2019.

[Purwins *et al.*, 2019] Hendrik Purwins, Bo Li, Tuomas Virtanen, Jan Schlüter, Shuo-Yiin Chang, and Tara Sainath. Deep Learning for Audio Signal Processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):206–219, May 2019.

[Rannen *et al.*, 2017] Amal Rannen, Rahaf Aljundi, Matthew B. Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[Rebuffi *et al.*, 2017] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[Rodríguez *et al.*, 2021] Sandra Servia Rodríguez, Cecilia Mascolo, and Young D. Kwon. Knowing when we do not know: Bayesian continual learning for sensing-based analysis tasks. *CoRR*, abs/2106.05872, 2021.

[Rolnick *et al.*, 2019] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[Rusu *et al.*, 2016] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[Rusu *et al.*, 2018] Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*, 2018.

[Salamon *et al.*, 2014] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. A dataset and taxonomy for urban sound research. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 1041–1044, 2014.

[Schwartz *et al.*, 2018] Eli Schwartz, Leonid Karlinsky, Joseph Shtok, Sivan Harary, Mattias Marder, Abhishek Kumar, Rogerio Feris, Raja Giryes, and Alex Bronstein. Delta-encoder: an effective sample synthesis method for few-shot object recognition. *Advances in neural information processing systems*, 31, 2018.

[Shen *et al.*, 2019] Wei Shen, Ziqiang Shi, and Jun Sun. Learning from adversarial features for few-shot classification. *arXiv preprint arXiv:1903.10225*, 2019.

[Shi *et al.*, 2021] Guangyuan Shi, Jiaxin Chen, Wenlong Zhang, Li-Ming Zhan, and Xiao-Ming Wu. Overcoming catastrophic forgetting in incremental few-shot learning by finding flat minima. *Advances in neural information processing systems*, 34:6747–6761, 2021.

[Shin *et al.*, 2017] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

[Song *et al.*, 2000] Sen Song, Kenneth D Miller, and Larry F Abbott. Competitive hebbian learning through spike-timing-dependent synaptic plasticity. *Nature neuroscience*, 3(9):919–926, 2000.

[Su *et al.*, 2019] Yu Su, Ke Zhang, Jingyu Wang, and Kurosh Madani. Environment sound classification using a two-stream cnn based on decision-level fusion. *Sensors*, 19(7):1733, 2019.

[Sun *et al.*, 2019] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 403–412, 2019.

[Tao *et al.*, 2020] Xiaoyu Tao, Xinyuan Chang, Xiaopeng Hong, Xing Wei, and Yihong Gong. Topology-preserving class-incremental learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIX 16*, pages 254–270. Springer, 2020.

[Thuseethan *et al.*, 2021] Selvarajah Thuseethan, Sutharshan Rajasegarar, and John Yearwood. Deep continual learning for emerging emotion recognition. *IEEE Transactions on Multimedia*, 24:4367–4380, 2021.

[Vinyals *et al.*, 2016] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

[Wang *et al.*, 2020] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys (csur)*, 53(3):1–34, 2020.

[Wang *et al.*, 2024] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A Comprehensive Survey of Continual Learning: Theory, Method and Application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5362–5383, 2024.

[Warden, 2018] Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

[Wu *et al.*, 2019] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 374–382, 2019.

[Yoo *et al.*, 2018] Donghyun Yoo, Haoqi Fan, Vishnu Boddeti, and Kris Kitani. Efficient k-shot learning with regularized deep networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[Yoon *et al.*, 2017] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

[Yosinski *et al.*, 2014] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

[Zamir, 1998] Ram Zamir. A proof of the fisher information inequality via a data processing argument. *IEEE Transactions on Information Theory*, 44(3):1246–1250, 1998.

[Zenke *et al.*, 2017] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pages 3987–3995. PMLR, 2017.

[Zhai *et al.*, 2019] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

[Zhang *et al.*, 2020] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1131–1140, 2020.

[Zhang *et al.*, 2021] Chi Zhang, Nan Song, Guosheng Lin, Yun Zheng, Pan Pan, and Yinghui Xu. Few-shot incremental learning with continually evolved classifiers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12455–12464, 2021.

[Zhou *et al.*, 2022] Da-Wei Zhou, Fu-Yun Wang, Han-Jia Ye, Liang Ma, Shiliang Pu, and De-Chuan Zhan. Forward compatible few-shot class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9046–9056, 2022.

[Zhu *et al.*, 2021] Kai Zhu, Yang Cao, Wei Zhai, Jie Cheng, and Zheng-Jun Zha. Self-promoted prototype refinement for few-shot class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6801–6810, 2021.

Supplementary Material

MetaCLBench: Meta Continual Learning Benchmark on Resource-Constrained Edge Devices

Table of Contents

A Detailed Benchmark Framework **13**

A.1 Additional Baselines & Methods . . . 13

A.2 Datasets . . . 14

A.3 Model Architectures . . . 15

A.4 Detailed Experimental Setup . . . 15

B Additional Results **16**

B.1 Accuracy . . . 16

B.2 Peak Memory Footprint . . . 17

B.3 End-to-end Latency & Energy Consumption . . . 18

C Further Analysis and Discussion **18**

C.1 With and Without pre-training on CNN . . . 18

C.2 Number of replay epochs . . . 18

C.3 Multi Compression Module. . . . 19

D Extended Related Work **20**

D.1 Continual Learning . . . 20

D.2 Meta Learning . . . 21

D.3 Meta Continual Learning . . . 22

A Detailed Benchmark Framework

This section provides additional information on our MetaCLBench framework.

A.1 Additional Baselines & Methods

Pretrained: To initialize the model weights, this baseline performs conventional deep neural network (DNN) training without meta-learning procedure and/or Meta-CL methods during the meta-training phase. After that, it finetunes the model weights with a few samples during a meta-testing phase, similar to other Meta-CL methods. As Pretrained typically shows low performance, it serves as the *lower bound* in our evaluation. Also, a comparison between this baseline and other methods can reveal the effectiveness of the meta-learning procedure in a CL task.

Oracle: This baseline represents the case where it has access to all the classes at once in an i.i.d. fashion. Also, we assume that Oracle would perform DNN training for multiple epochs until the model converges. As a result, Oracle often shows superior accuracy, and thus its accuracy represents the *upper bound* of our evaluation.

OML [Javed and White, 2019]: This method constantly optimizes representations during online updates to quickly acquire predictions for new tasks, facilitating seamless and efficient CL. This approach uses CF as signals to justify the use of storage space to accommodate new tasks. The OML can accomplish CL in more than 200 consecutive classes. It is an online updating method that constantly learns representations and often outperforms rehearsal-based continuous learning methods. The fundamental difference between OML and MAML-Rep lies in their approach to updates. OML employs a single data point for each update, while MAML-Rep utilizes an entire batch. By adopting this method, OML is able to more accurately consider the impact of online learning, including the phenomenon of CF. The objective of OML is to implement the computation graph for live updates and to discover representations that maintain high accuracy even with ongoing changes. It acquires sparse representations without any inactive neurons, surpassing traditional and rehearsal-based strategies for ongoing learning. The technique entails meta-training, with an emphasis on avoiding interference and facilitating rapid adaption during online updates.

ANML [Beaulieu *et al.*, 2020]: This is an approach inspired by the neuromodulation process in the brain. This method accomplishes selective activation (forward pass) and selective plasticity (backward pass) in DNNs by meta-learning through a CL process within the prediction network. Optimizing activation timing by controlling the activation of a predictive model based on input conditions enhances CL. It is often reported to be superior to OML, as it can be optimized to reduce both forward and backward interference. ANML's architecture comprises two parallel networks: the NM network governs the activations and plasticity of the PLN. ANML employs a unique method where it adjusts specific components of the PLN based on whether it is in the meta-training or meta-testing phase, distinguishing it from conventional methods. The results of the experiments demonstrate that ANML surpasses other approaches by a large margin. It maintains a high level of accuracy even when faced with 600 consecutive tasks. This is attributed to its capability to generate sparse yet powerful representations and effectively manage sequential learning without erasing previously acquired knowledge.

OML+AIM [Lee *et al.*, 2021]: This is a Meta-CL approach that integrates Attentive Independent Mechanisms (AIM), which is a modular component of the OML framework. AIM increases the speed of learning a new task by selecting out the mechanism that best explains the representation of the inner loop to activate. AIM is comprised of a collection of autonomous mechanisms, with each mechanism being defined by its own unique set of parameters. Each mechanism functions as an autonomous specialist that cooperates with other specialists to solve assignments. AIM is situated between the feature extractor and the output classifier in a deep neural network. This approach incorporates higher-level data representation and employs rapidly updated weights within the inner loop of a meta-learning process. During the training process, AIM mechanisms engage in competition and focus on input representations through cross-attention. This allows them to select a limited number of mechanisms for task-specific learning.

ANML+AIM [Lee *et al.*, 2021]: This is a Meta-CL approach that incorporates AIM into the ANML framework. The accuracy of SOTA Meta-CL continues to improve beyond the performance of the previous three approaches by integrating AIM into the existing CL framework. AIM is positioned following a feature extractor and preceding an output classifier, allowing it to effectively process intricate representations. AIM, in contrast to RIMs, does not employ LSTM-based temporal modeling, rendering it a static module. AIM's processes engage in a targeted manner according to the relevance of the information, which helps to create a concise representation and enhance the ability to adapt to new activities while reducing the risk of CF. It has demonstrated substantial enhancements in accuracy for tasks involving few-shot and ongoing learning, surpassing existing approaches without increasing the number of parameters.

Lifelearner [Kwon *et al.*, 2024a]: This module is based on ANML and combines both lossy and lossless compression to achieve high compression rates and minimize footprint. The Lifelearner approach not only achieves optimal CL efficiency but also significantly reduces energy consumption and latency compared to the SOTA. LifeLearner consists of two distinct phases: meta-training, which takes place on a server to establish a strong initial weight configuration, and meta-testing, which occurs on devices to continuously acquire knowledge of new classes while retaining information from prior ones. The main characteristics consist of an enhanced compression module that minimizes memory consumption, a feature extractor that remains unchanged, and a classifier that is constantly updated through learning. And it achieves efficient learning even on resource-constrained edge devices.

Latent OML: This module is based on OML and incorporates the Compression modular from Lifelearner to achieve high compression rates and minimal footprint. It is used in the testing of advanced architectures (YAMNet and ViT).

A.2 Datasets

To complement the relative simplicity, limited variety, and lack of generalizability of the datasets tested in previous studies, we added three audio datasets to the single-label image datasets (*CIFAR-100* [Krizhevsky *et al.*, 2009], *MiniImageNet* [Vinyals *et al.*, 2016]) from previous studies. Specifically, we used the *UrbanSound8K* [Salamon *et al.*, 2014] dataset with fewer classes, the *ESC-50* [Piczak, 2015] dataset with more classes, and the *GSCv2* [Warden, 2018] dataset for keyword spotting (KWS). We demonstrate trade-offs between performance and system resources on resource-constrained devices by extensively evaluating these datasets in different Meta-CL approaches. The details of each target dataset employed in our study are described below.

CIFAR-100 [Krizhevsky *et al.*, 2009]. This dataset, developed by researchers supported by the Canadian Institute for Advanced Research, has 60,000 photos categorized into 100 classes, with each class including 600 images. The creation of this used the identical technique as CIFAR-10. The images were obtained by utilizing search phrases from WordNet, and any duplicate or irrelevant images were eliminated. The dataset is structured into 20 superclasses, with each superclass consisting of 5 associated classes. The labelers meticulously inspected and assigned labels to each image based on precise instructions, guaranteeing that each class was distinct from the classes in CIFAR-10. This ensures that CIFAR-100 may be effectively used as a negative example for CIFAR-10. The dataset is designed for the purpose of training models in object recognition, specifically focusing on small, natural photos.

MiniImageNet [Vinyals *et al.*, 2016]. This dataset was formed by randomly choosing 100 classes from the complete ImageNet dataset. Each class in the Mini-ImageNet dataset consists of 600 color images, each having dimensions of 84x84 pixels. The classes were partitioned into 80 for training and 20 for testing, with the guarantee that the test classes were never exposed during the training process. This dataset is intentionally more intricate than CIFAR-10, but still feasible for contemporary machine memory. It enables quick development and experimentation. ImageNet is an extensive visual recognition dataset comprising more than 14 million photos categorized based on the WordNet hierarchy. Every image is annotated with the corresponding thing it depicts. The dataset facilitates a range of computer vision tasks, such as object detection, classification, and segmentation. For the purpose of classification, it includes a collection of photos spanning 1,000 different categories. ImageNet's extensive size and wide range of variations have established it as a standard for training and assessing deep learning models.

UrbanSound8K [Salamon *et al.*, 2014]. The development of UrbanSound required the compilation of a wide-ranging and comprehensive dataset of urban noises. The researchers made use of the Freesound archive, which contains more than 160,000 recordings contributed by users, with a significant number of them being from metropolitan environments. They conducted a search and retrieved recordings by utilizing specified class names, manually weeding out non-field recordings. The outcome of this was the acquisition of 1302 recordings, amounting to a total of 27 hours of audio. Out of these, 18.5 hours were dedicated to manually annotating sound occurrences in 10 different categories, namely air conditioner, car horn, children playing, dog bark, drilling, engine idling, pistol shot, jackhammer, siren, and street music. Every recording was marked with annotations indicating the precise start and end times of sound events, as well as their prominence, whether they were in the foreground or background. UrbanSound8K is a subset of a dataset specifically created to train sound classification algorithms. It contains 8.75 hours of audio that has been divided into 4-second segments. The taxonomy classifies sounds into four main groups: human, nature, mechanical, and music, enabling thorough and methodical examination of urban noises.

ESC-50 [Piczak, 2015]. The primary objective of creating the ESC-50 dataset was to alleviate the limited availability of freely accessible datasets containing ambient sounds. The dataset comprises 2,000 audio clips, each tagged and lasting 5 seconds. These clips are categorized into 50 classes, which are further separated into five key categories: animal sounds, natural soundscapes and water sounds, human non-speech sounds, interior/domestic sounds, and exterior/urban noises. The selection of clips was carefully picked to prioritize the prominence of sound events while minimizing the presence of background noise. The quality control process included CrowdFlower's methods and additional validation by the author, resulting in approximately twelve human categorization entries per clip. The dataset was generated with the purpose of aiding research in the field of environmental sound categorization, providing a reliable benchmark for the evaluation of automatic sound recognition systems. The dataset is accessible to the public and contains an additional set of 250,000 video clips without labels, which can be used for unsupervised learning experiments. The ESC-50 dataset is a valuable resource for investigating machine learning methods in the field of ambient sound classification.

GSCv2 [Warden, 2018]. This dataset, created by Pete Warden from Google Brain, seeks to simplify the process of training and evaluating keyword-detecting systems. The creation of the "Speech Commands" dataset entailed gathering spoken words from a varied collection of speakers through a web-based application that utilized the WebAudioAPI to record utterances. The dataset comprises 35 distinct word utterances, encompassing categories such as numbers, frequent instructions, and miscellaneous terms such as "tree" and "wow". The dataset is specifically intended for limited-vocabulary speech recognition tasks, with the objective of identifying instances where one of the target words is spoken from a selection of ten or fewer terms. In addition, the dataset contains recordings of background noise to use in training models to distinguish between speech and non-speech audio. The objective was to record audio that accurately represented the task of on-device trigger phrases, with a specific emphasis on 10 words: "Yes", "No", "Up", "Down", "Left", "Right", "On", "Off", "Stop", and "Go". Every user would activate a "Record" button, and a randomly selected word would appear on the screen for a duration of 1.5 seconds while audio was being captured. The audio snippets were saved as 16-bit single-channel PCM values at a 16 KHz rate in the form of one-second WAV files. The dataset was specifically curated to be appropriate for training and assessing keyword spotting systems, with an emphasis on offering a specialized dataset that distinguishes itself from conventional datasets utilized for complete sentence automatic speech recognition.

A.3 Model Architectures

Referring to previous work [Lin *et al.*, 2022], we add optimized DNN architectures and transformer architectures intended for resource-limited IoT devices to the evaluated three-layer CNN architectures, including **YAMNet** [Howard *et al.*, 2017], and **ViT** [Dosovitskiy *et al.*, 2020]. YAMNet employs the MobileNetV1, which is a highly efficient model specifically designed for applications involving mobile and embedded vision. Depthwise separable convolutions are utilized in their construction, thereby reducing computing complexity by decomposing the normal convolution operation into separate depthwise and pointwise convolutions. This efficient structure enables the development of compact deep neural networks. MobileNets additionally incorporate two hyperparameters, namely width multiplier and resolution multiplier, that allow for adjusting the balance between model size, latency, and accuracy. Developers can tailor MobileNets to match the precise resource limitations of their application by fine-tuning these hyperparameters. MobileNets have demonstrated equivalent performance to larger, more intricate models, but with a substantially smaller size and faster computational speed. They have been effectively utilized in a wide range of activities including item identification, precise categorization, facial features, and other applications. YAMNet provides a flexible and effective approach for implementing deep learning models on mobile and embedded devices. Moreover, MobileNets may be customized to meet unique application needs by modifying the hyperparameters, rendering them adaptable and effective solutions for on-device artificial intelligence. The Vision Transformer (ViT) is an architecture based on transformers, which has demonstrated encouraging outcomes in activities related to image identification. ViT, unlike typical Convolutional Neural Networks (CNNs), operates on inputs in a 2-dimensional format instead of a 1-dimensional series of patches. The model includes Axial Transformer blocks, consisting of row-self-attention followed by a multi-layer perceptron (MLP), and then column-self-attention followed by an MLP, instead of the conventional self-attention followed by an MLP. ViT has undergone pre-training using extensive datasets such as ImageNet-21k and JFT-300M. It has demonstrated exceptional performance on several image recognition benchmarks, including ImageNet, CIFAR-100, and the VTAB classification suite [Zhai *et al.*, 2019], surpassing all previous records. ViT models exhibit superior memory efficiency and outperform CNNs with reduced computational resources. They can be optimized for specific tasks such as detection and segmentation, and demonstrate potential for scenarios with limited data transport. ViT is a transformer-based model that has shown impressive results in image identification tasks, particularly when it is pre-trained on extensive datasets and then fine-tuned for specific applications.

A.4 Detailed Experimental Setup

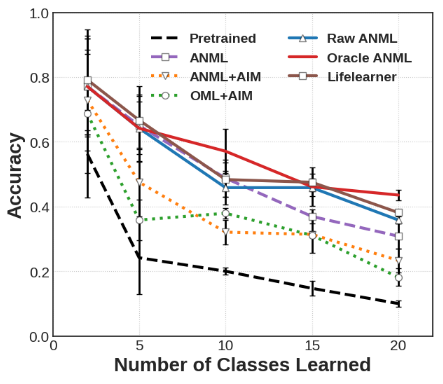
Training Detail: Consistent with prior meta-training research, we employed a batch size of 1 for the inner-loop updates and 64 for the outer-loop updates across 20,000 steps to ensure the accuracy of our results. To obtain a meta-trained DNN with the highest possible validation set accuracy, we tested with various learning rates required for both the inner and outer loops. Consequently, we established the learning rate for the inner loop (α) was set to 0.001, and the learning rate for the outer loop (β) was also set to 0.001 for all datasets. During the ESC-50 test, we conducted five training sessions, using 1600 clips as the training set and 400 clips as the test set. In the meta-testing phase, we assessed ten distinct learning rates for each technique and reported the most successful outcomes. Additionally, to evaluate the precision of replay systems, we experimented with batch sizes of 8 and 16. We noticed minimal fluctuations in the performance of CL. We selected a batch size of 8 because it requires less memory than larger batch sizes.

Evaluation Protocol: With reference to previous research [Beaulieu *et al.*, 2020], this study prioritizes the accuracy of unseen samples across new categories in CL. The accuracy rate signifies the capacity of CL to make generalizations. In addition, we analytically calculate the memory footprint required to perform CL (model parameters, optimizers, and activations memory). To obtain the peak memory footprint during the CL process, we include (1) model memory for the model itself and the weights to be updated, (2) optimizer memory for gradients, and (3) activations memory for intermediate outputs for weights update. Furthermore, we measure the end-to-end latency and energy consumption of various approaches to CL on the edge device over specified categories. To measure the end-to-end latency and energy, we include the time and energy used to: (1) load a model and(2) perform CL using all the given samples (e.g., 30) over all the given classes (e.g., 10). We test the latency and total energy expenditure for end-to-end CL by deploying MetaCLBench and baselines on an edge device, Raspberry Pi 3B+. Regarding energy, we measure the total amount of energy consumed by a device during the end-to-end CL process. This is performed by measuring the power consumption on Raspberry Pi 3B+ using a YOTINO USB power meter and deriving the energy consumption following the equation: Energy = Power × Time.

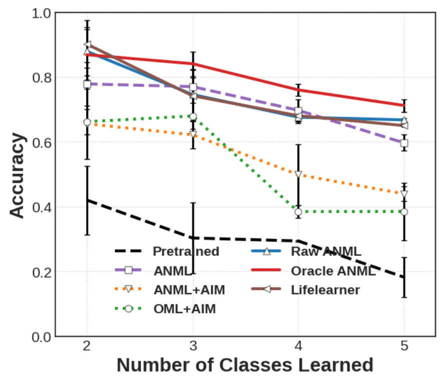
System Implementation: The initial phase of Meta-CL is the meta-training stage, which facilitates swift deployment of the model by initializing the neural network with pre-trained weights. Subsequently, the meta-testing phase incorporates optimizations compatible with hardware, enhancing system efficiency when implemented on embedded devices such as Raspberry Pi 3B+. Source code of MetaCLBench will be publicly available upon publication[2].

Embedded Device: The Raspberry Pi 3B+ is equipped with a quad-core ARM Cortex-A53 processor and has 1 GB of RAM. It should be noted that the available memory on the Raspberry Pi 3B+ while it is not in use is approximately 600 MB. This is because some of the memory is already being used by background processes, concurrent apps, and the operating system. We

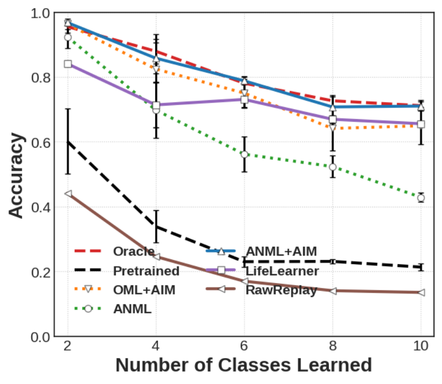
[2]https://github.com/theyoungkwon/MetaCLBench



(a) ESC-50

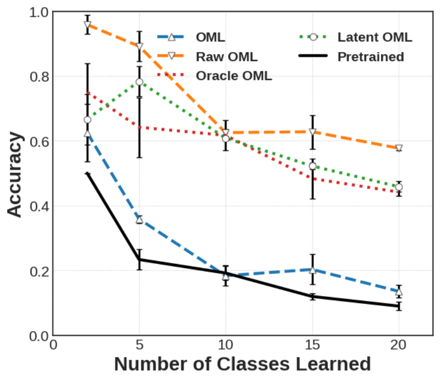


(b) Urbansound8K

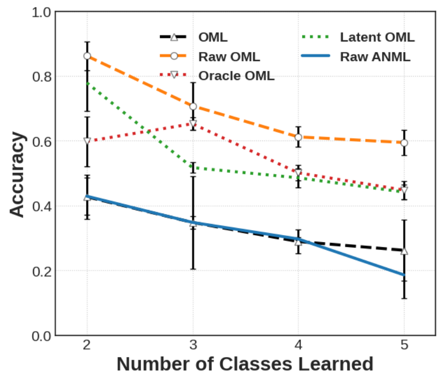


(c) GSCv2

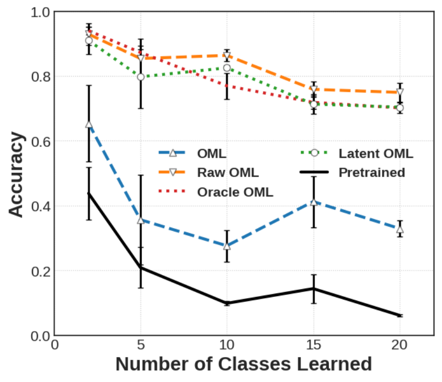
Figure 4: The accuracy of ESC-50, Urbansound8k, GSCv2 datasets using the **3-Layer CNN** architecture.



(a) ESC-50



(b) Urbansound8K



(c) GSCv2

Figure 5: The accuracy of ESC-50, Urbansound8k, GSCv2 datasets using the **YAMNet** architecture.

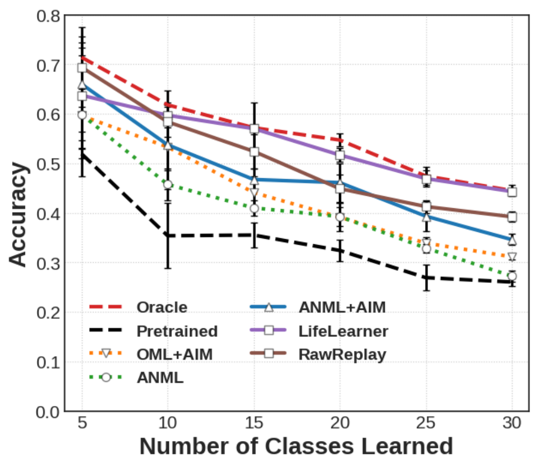
utilize Faiss (PQ Framework) [Johnson *et al.*, 2019] and PyTorch 1.13 (Deep Learning Framework) [Paszke *et al.*, 2019] as software platforms to create and assess the meta-training and meta-testing stages on embedded devices.

B Additional Results

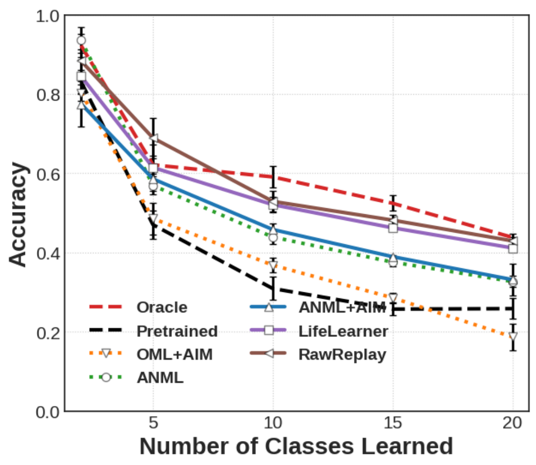
In this section, we present additional results that are not included in the main content of the paper due to the page limit.

B.1 Accuracy

To visualize the change in accuracy as the training set grows and how each Meta-CL method compares, we provide the results of accuracy on the three architectures (3-layer CNN, YAMNet, and ViT) and six Meta-CL methods based on the five datasets as shown in Figures 4, 5, 6, and 7. The results reveal that models relying solely on pre-trained weights ("Pretrained") exhibit lower accuracy, averaging 22.18% and 13.8% across the datasets for CNNs and more advanced architectures, such as YAMNet and ViT. This result indicates that traditional transfer learning fails to address the complexities of few-shot learning scenarios. Within CNNs, baseline methods like ANML demonstrated improvement over the pretrained-only approach (with an average accuracy gain of 16.7%). This improvement was further augmented by incorporating the AIM, resulting in an additional 2.54% average accuracy gain. However, these gains still fell short of those achieved by the Oracle and Lifelearner methods (by a deficit of 15.92% in accuracy). The constrained sample size in the meta-testing phase poses a challenge to attaining high accuracy levels, even considering the theoretical upper limit of accuracy. Our research also indicated that both CNNs and advanced models experience a marked decline in performance when meta-training proceeds without prior pre-training, noting an average 13.98% improvement after pre-training. Given that advanced models intrinsically include pre-trained weights, further pre-training does not substantially increase their accuracy. Thus, pre-training is affirmed as a critical step for enhancing model accuracy. This study underscores that regardless of the method employed, the procedural sequence of pre-training followed by meta-training before evaluation is indispensable for attaining heightened accuracy levels, recognizing that advanced models inherently incorporate a

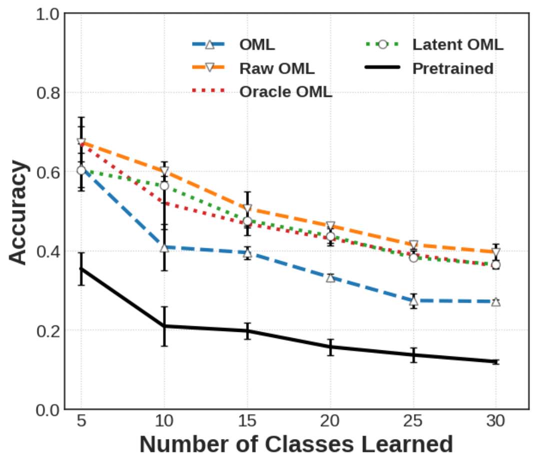


(a) CIFAR100

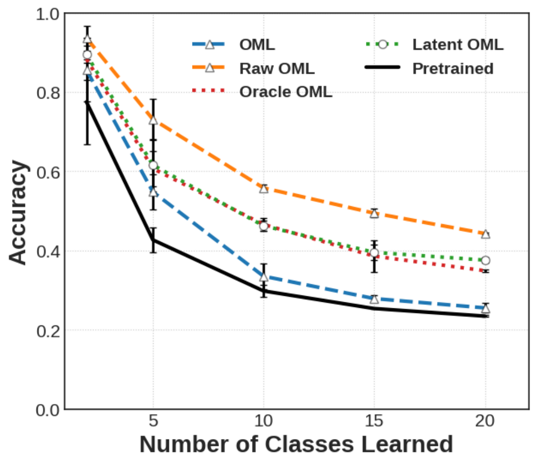


(b) MiniImgenet

Figure 6: The accuracy of CIFAR100, MiniImgenet datasets using the **3-Layer CNN** architecture.



(a) CIFAR100



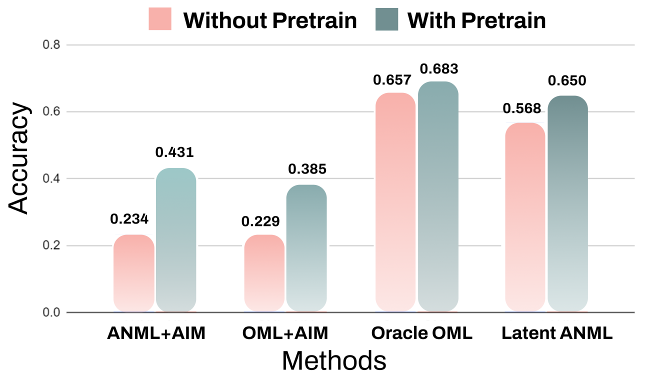
(b) MiniImgenet

Figure 7: The accuracy of CIFAR100, MiniImgenet datasets using the **ViT** architecture.

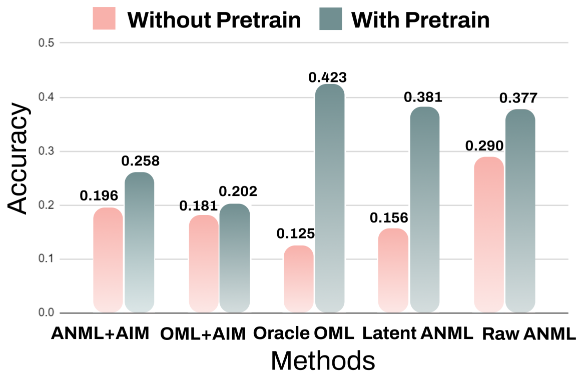
pre-training phase. These observations indicate a non-trivial trade-off between accuracy, memory, and computation for all the employed architectures in our work.

B.2 Peak Memory Footprint

Measurements were conducted to ascertain the maximum amount of RAM required for executing backpropagation and storing rehearsal samples. The memory for performing backpropagation is divided into three parts: (1) model memory, which is responsible for storing model parameters; (2) optimizer memory, which is responsible for storing gradient and momentum vectors; (3) activation memory, which is responsible for storing intermediate activations that are reused during backpropagation. Table 3 shows the various methods and the peak memory usage of our system. First, we found that although AIM improves accuracy, it takes up much memory space because there are many parameters in the AIM module. The memory required is as large as 1,562 MB for MiniImageNet and 2,648 MB for UrbanSound8K, which far exceeds the RAM capacity of embedded devices like the Pi 3B+ and causes an out-of-memory (OOM) problem. Consequently, it becomes challenging to utilize it in edge devices. OML, ANML, Oracle, Pretrained, and LifeLearner have a more minor memory requirement. However, Pretrained and ANML are less accurate. In contrast, LifeLearner is the most satisfactory, requiring only 136-496 MB of very high compression rates with high accuracy comparable to Oracle.



(a) Urbansound8k



(b) ESC-50

Figure 8: Comparison of accuracy with and without pretraining for datasets UrbanSound8K and ESC-50 using the 3-Layer CNN architecture.

B.3 End-to-end Latency & Energy Consumption

Table 3 also presents the operational efficiency of various Meta-CL methods on edge devices, with specific emphasis on end-to-end latency and energy consumption. The end-to-end latency is calculated by taking into account three different times: (1) The duration required for loading the pre-trained model, (2) The duration needed for sequentially training the model for each specified class, and (3) The duration needed for employing further optimization proposed in specific Meta-CL methods such as compression method of LifeLearner or training additional module such as AIM. First, we measured the end-to-end latency of CL for all systems on the Raspberry Pi CPU for all classes for each dataset (30 samples per class). LifeLearner achieved fast end-to-end latency. Due to the large amount of memory required by the AIM series, crashes occurred several times during the test due to memory exhaustion. While Raw OML may exhibit the highest accuracy among advanced architectures, our tests have demonstrated that Raw methods also require the most computational resources in terms of latency and energy consumption. Additionally, our findings indicate that the YAMNet architecture significantly decreases capacity requirements due to its smaller size; however, it exhibits greater delays and higher energy consumption as a result of its more intricate architecture compared to CNN. As a result, we must consider factors beyond a single result alone when selecting Meta-CL methods for edge devices.

To establish generalizability, we extended our evaluation to diverse edge devices with varying computational capabilities (Jetson Nano with enhanced specifications and Pi Zero 2 with reduced capacity). Table 2 below shows the end-to-end latency and energy consumption results across these three edge devices, confirming that LifeLearner achieves a 7.5-fold reduction in latency compared to Oracle.

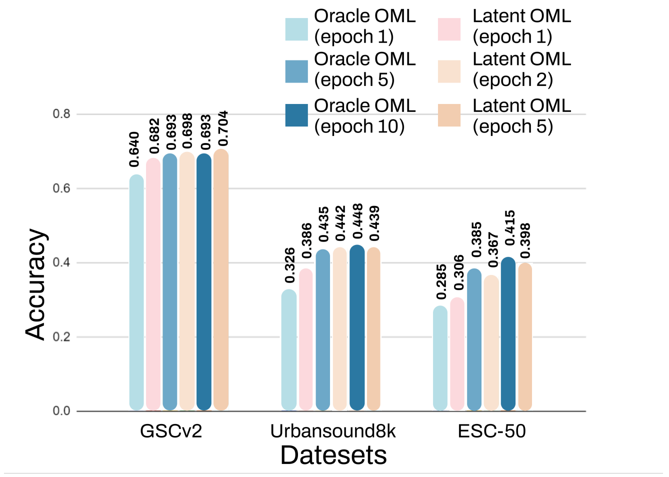
C Further Analysis and Discussion

C.1 With and Without pre-training on CNN

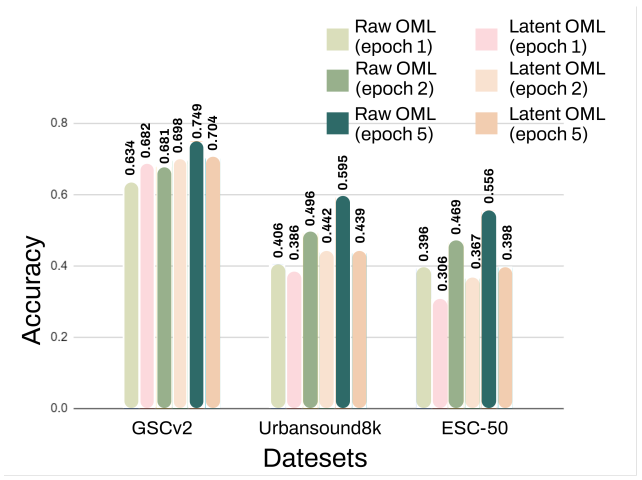
In this section, we examine the impact of hyperparameter variations on model accuracy. First, we investigate the influence of pre-training on the accuracy metric. Initially, we evaluated the five datasets without pre-training; the two image datasets and GSCv2 exhibited satisfactory accuracy. However, the performance on Urbansound8k and ESC-50 was suboptimal. Consequently, we adopted the strategy of pre-training as employed in advanced models, incorporating pre-training before meta-training for each method. Figure 8 illustrates that including pre-training significantly enhances model efficacy. Specifically, in the Urbansound8k and ESC-50 datasets, all Meta-CL methods demonstrated an average efficiency increase of 13.98% following pre-training. Consequently, pre-training plays a significant role in enhancing accuracy, particularly for fundamental algorithms such as ANML and OML, where accuracy without pre-training is a mere 5%. Advanced architectures intrinsically incorporate pre-training, obviating the need for additional pre-processing. Nonetheless, these sophisticated architectures, when incorporating pre-training, exhibit increased latency and energy consumption.

C.2 Number of replay epochs

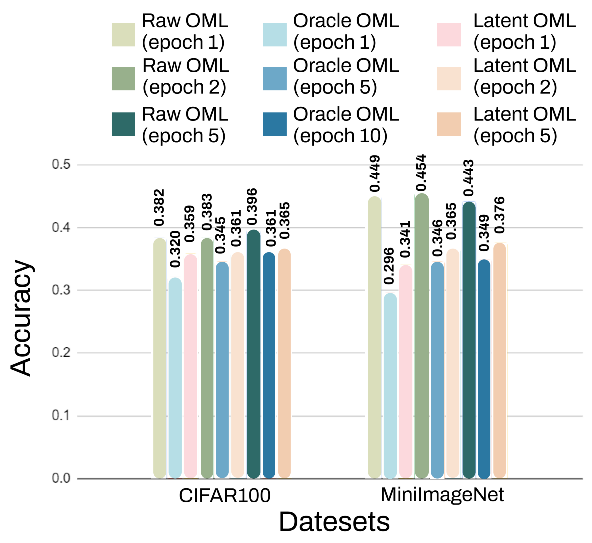
Recognizing that an increased number of replays correlates with heightened latency and energy usage, we explored the association between replay frequency and accuracy. Figure 9a indicates that Oracle OML necessitates more than five replays to attain elevated accuracy levels, thereby escalating energy expenditure throughout the training phase. Conversely, Latent OML demonstrates superior performance, requiring merely one or two replays to reach comparable accuracy, thereby diminishing system overhead. In the case of the same dataset, the efficiency of Latent OML after five replays approximates that of Oracle OML following ten replays. The same observation is evident in the CNN architecture depicted in Figure 9d; here, one or two iterations of the Latent



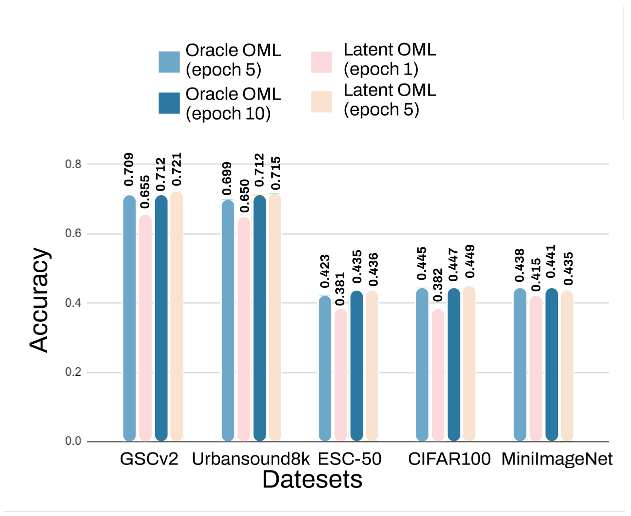
(a) Oracle& Latent under YAMNet architecture



(b) Raw& Latent under YAMNet architecture



(c) ViT architecture



(d) CNN architecture

Figure 9: Comparison of different Meta-CL methods regarding the number of replay epochs.

method attain the same level of accuracy as the Oracle method after five iterations. Additionally, we note that exceeding five iterations does not significantly enhance accuracy. Figures 9band 9cillustrate that the accuracy of Raw OML within the advanced methods surpasses that of the previously tested Latent approach.

C.3 Multi Compression Module.

In CNN architectures, ANML with AIM demonstrates substantial accuracy improvements (26.0% to 34.6% on CIFAR100), albeit with increased memory requirements (39.69MB to 1,093MB). LifeLearner presents an optimal balance between accuracy and memory utilization in resource-constrained environments. The integration of AIM, while beneficial for accuracy, introduces significant computational overhead, particularly in Raw ANML implementations (11,424s latency, 52.5KJ energy consumption on CIFAR100). Oracle ANML offers a compromise between performance enhancement and resource efficiency.

Vision Transformer (ViT) and YAMNet architectures exhibit analogous performance-resource tradeoffs. ViT implementations with OML and AIM show improved accuracy but incur substantial computational costs (22,968s latency, 103KJ energy consumption on CIFAR100). Similarly, YAMNet with AIM achieves superior accuracy (42.9% to 71.0% on GSCv2) at the expense of increased memory utilization (10.16MB to 608.2MB). Pretrained YAMNet models demonstrate superior resource efficiency metrics.

D Extended Related Work

D.1 Continual Learning

Neural network training is difficult to shift from batch learning to a continuous learning model because neural networks are different from human brain mechanisms. Deep neural network-based systems that are continuously updated with new data suffer from a severe forgetting problem, where performance on old tasks typically drops significantly, known as CF [McCloskey and Cohen, 1989; Parisi *et al.*, 2019; Wang *et al.*, 2024; Kwon *et al.*, 2021a; Jha *et al.*, 2021]. Researchers have put forward numerous approaches to prevent CF, mostly encompassing rehearsal-based, regularization-based, and parameter isolation-based strategies.

(1) Rehearsal-based approaches [Rebuffi *et al.*, 2017; Wu *et al.*, 2019; Chauhan *et al.*, 2020; Kwon *et al.*, 2024a] allow storing some of the early training data, i.e., old data, in a cache, which will be used for replay or prototype as new tasks are learned as a way of preventing CF. At this point, cache size, storage data selection, data storage form, and the way old and new data are mixed are all important influencing factors. Rehearsal-based approaches include memory-based, generation-based replay, and other approaches. Memory-based approaches generally necessitate a memory buffer for storing data instances or pertinent information from the previous task. This stored information is then retrieved and utilized during the learning of a new task to reinforce the knowledge acquired from the prior activity. Rebff et al. first proposed a strategy for learning both the classifier and the feature representations simultaneously during a continuous learning process, which became an incremental classifier(iCARL) based on representational learning [Rebuffi *et al.*, 2017]. During representation learning, iCARL is trained using a small number of data instances representative of the old category stored and instances from the new task. Wu et al. suggest that the present approaches, which demonstrate good performance on small datasets with only a few tasks, are unable to sustain their performance on huge datasets with thousands of tasks [Wu *et al.*, 2019]. Therefore, they proposed bias-corrected (BIC) methods that focus on continuous learning from large datasets. Because there is actually a strong bias in the classification layer for each new task, BIC segregates the validation set from the amalgamation of previous and new data instances and incorporates a linear bias correction layer subsequent to the classification layer. This correction layer utilizes the validation set to rectify any bias present.

Generative replay-based methods are an alternative to memory-based methods, using generative modules that can reproduce information related to previous tasks instead of memory buffers. Shin et al. proposed a deep generative replay method (DMC) that sequentially learns multiple tasks by mimicking pseudo-data generated from previous training examples by a GAN model and pairing them with corresponding labels [Shin *et al.*, 2017]. Nevertheless, a significant limitation is that the efficacy of the statement is contingent upon the caliber of the generator. Deep model consolidation (DMC) was proposed by Zhang et al [Zhang *et al.*, 2020]. The process begins by training a distinct model for the new class using labeled data. Subsequently, the old and new models are merged by utilizing publicly accessible unlabeled auxiliary data. This approach overcomes the difficulties because of the inaccessibility of legacy data and mines these auxiliary data for rich transferable representations to facilitate incremental learning.

Generative replay methods have been shown to be effective on problems with relatively simple inputs, but for more complex input problems, such as natural images, generative replay methods may face greater challenges. While generative replay still has some advantages in class incremental learning problems using natural images, these methods rely on pre-trained networks or extensive non-incremental initialization phases.

(2) Regularization-based approaches [Kirkpatrick *et al.*, 2017; Zenke *et al.*, 2017] usually impose restrictions on various model parameters and hyperparameters during the updating process to achieve the goal of consolidating knowledge learned in a straight line while learning a new task, thus mitigating CF in continuous learning. Regularization-based approaches include methods such as optimizing constraints on important model parameters to add distillation losses with respect to the model as an objective.

A common approach is to enhance the loss function by regularizing previously acquired knowledge, which is a technique that gained recognition as Enhanced Weight Consolidation (EWC) was first embodied by Kirkpatrick et al. [Kirkpatrick *et al.*, 2017]. EWC introduces a novel quadratic penalty term in the loss function to limit pattern modifications to weights that were more important for the previous learning task. In order to limit the parameter to low-loss regions that are common between tasks, instead of only updating low-loss regions for new tasks by calculating the importance of the weights using the diagonal of the Fisher information matrix [Zamir, 1998], thus mitigating the CF problem. The rebalancing uniform classifier LUCIR method was proposed by Hou et al. [Hou *et al.*, 2019]. This method proposes three loss functions to bind the bias problem present in incremental learning due to the imbalance between old and new samples. The data is rebalanced by re-assigning weights in each iteration cycle. This means that for a few class instances, higher weights will be assigned; for the majority of class instances, lower weights will be assigned. This reallocation technique ensures that the model pays more attention to the minority class, thus improving the overall classification performance.

Methods utilizing knowledge distillation [Shin *et al.*, 2017] incorporate the concept of knowledge distillation into incremental learning by extracting knowledge from a model trained on a previous task into a model trained on a new task, therefore consolidating previously acquired knowledge. Li et al. proposed a method for learning without forgetting (LWF, Learning without Forgetting) [Li and Hoiem, 2017]. Prior to acquiring new knowledge, LWF retains a duplicate of the preceding model parameters. Subsequently, the model acquired from the new task data instances is employed as the classifier target for the old task. The classifier for the new task is utilized with the objective of attaining the real value. The total of the losses from the new task and old task is calculated and employed as the ultimate loss. Douilaed et al. refer to continuous learning as representation learning and propose a distillation loss that becomes a summarized output distillation (POD) [Douillard *et al.*, 2020], which restricts the updating of the final output and the learned representation of the middle layer. The disadvantage of such methods is that after the first task, all the parameters of the hidden layer are frozen but representation learning is limited.

(3) Parameter isolation-based approaches [Yoon *et al.*, 2017; Mallya and Lazebnik, 2018] typically involve expanding the old model on new tasks and assigning different model parameters to different tasks with varying degrees of isolation to prevent subsequent tasks from interfering with previously learned knowledge.

Parameter isolation-based approaches allocate distinct model parameters to individual tasks through the dynamic modification of the network architecture. Yoon et al. introduced a Dynamically Extensible Network (DEN) [Yoon *et al.*, 2017]that leverages acquired information from previous jobs and expands the network structure when the existing knowledge is inadequate for handling new tasks. Neurons undergo processes of addition, duplication, and separation to expand the structure of the network.

In addition to using a parameter isolation-based approach to dynamically change the network architecture, the researchers also designed a fixed network architecture. This architecture can still assign different parameters to handle different tasks. Mallya et al. proposed PackNet [Mallya and Lazebnik, 2018], a method that first fixes the parameters of the old task and second trains the entire neural network for each new task. After the training is completed, unnecessary parameters are released using weight-based pruning techniques and redundant spatial models are retained. Fernando et al. proposed PathNet [Fernando *et al.*, 2017]which initiates the training process by randomly selecting a few paths within the network. A race selection genetic algorithm is then employed to choose the most optimal paths for training on the given task. Afterward, for each subsequent task, the model parameters on all the paths chosen in the prior tasks are locked, while the remaining parameters are reset and trained again using the aforementioned approach to ultimately achieve the optimal path. Afterward, for each subsequent job, the model parameters on all the previously selected paths are fixed, while the remaining parameters are reset and trained again using the aforementioned method in order to ultimately choose the optimal path.

The method based on parameter isolation is simply to cope with the learning of a new task by adding parameters to the original model, but this method does not eliminate the interference between the new parameters and the old ones, nor does it ensure that the new parameters will not overwrite the old ones. Therefore, even with the parameter isolation-based approach, incremental learning is still limited by the problem of CF.

(4) Recently, preserving the structure of previously acquired knowledge has also emerged as a compelling area of research. Tao et al. proposed the Topology Preserving Class Incremental Learning (TPCIL) approach [Tao *et al.*, 2020]to store resilient Hebbian graphs [Song *et al.*, 2000]in buffers for data playback effects rather than just stored instances of prior data. TPCIL is motivated by the concept derived from human cognitive research that the process of forgetting is a result of the destruction of topological structures in human memory, and the method mitigates CF by injecting topological retention terms into the loss function.

The 'regularization and parameter isolation based approach' mentioned above sometimes does not work as well as it should. This is because the approach is not always consistent with new incremental learning and the advantages can even be negative.

D.2 Meta Learning

Meta Learning, also known as few-shot learning, is one of the hotspots of current research. Few-shot learning aims to use few training samples to recognize new classes that are not concatenable, regardless of the model's performance in recognizing the base class, and its core problem is that deep models tend to be overfitted to a limited number of training samples [Hospedales *et al.*, 2021]. To mitigate this issue, researchers have proposed a number of methods, which mainly include three main categories: model-based fine-tuning, data-based enhancement, and migration learning [Wang *et al.*, 2020].

(1) Meta learning methods [Finn *et al.*, 2017; Arik *et al.*, 2018; Chauhan *et al.*, 2022] based on model fine-tuning are more traditional approaches, which typically pre-train models on large-scale data and then fine-tune the existing parameters through regularization. Because of the limited quantity of training samples, how to tune the parameters without leading to overfitting becomes a key issue, for which the researchers proposed the following solution. a. Early stopping: Arik et al. separated the validation set from the training set in order to oversee the training process, and the training will cease when there is no discernible enhancement in the performance of the validation set [Arik *et al.*, 2018]. b. Selective updating of parameters: Keshari and his team gave a set of pre-trained filters that only learn the parameters that are multiplied by the filters, and this selective updating of the parameters effectively mitigates overfitting [Keshari *et al.*, 2018]. c. Simultaneous updating of relevant parts of the parameters: Yoo et al. group the pre-trained neural networks so that the filters are fine-tuned based on some auxiliary information using the training set via grouped backpropagation for the purpose of jointly updating each grouping using the same update information [Yoo *et al.*, 2018]. d. Using model regression networks: Kozerawski et al. mapped sample embeddings into a transformation function for classification decision boundaries [Kozerawski and Turk, 2018]. The model regression network represents a transformation that is independent of the specific task, which translates parameter values acquired via training on a limited number of cases to parameter values acquired through training on a substantial number of samples. Although the model-based fine-tuning approach is simpler, the differences between the source dataset and the target dataset may lead to overfitting of the model on the target dataset during use.

(2) The fundamental problem of Meta learning is that the sample size is too small, the problem can be supplemented by data enhancement methods to increase the diversity of the samples and supplement its insufficient number of images. Delta encoders were proposed by Schwartz et al [Schwartz *et al.*, 2018]. The model can extract transferable intraclass deformations between training samples of the same class and apply these increments to small samples of the new class, effectively synthesizing new

class samples as it goes. However, in this method, the features usually extracted by the classification network only focus on the most discriminative regions, while ignoring other less discriminative regions, which is not conducive to the generalization of the network. To solve this problem, Shen et al. replaced the fixed attention mechanism with an indeterminate one [Shen *et al.*, 2019]. First, the input image is extracted with features, and then the average is pooled and classified to obtain the cross-entropy loss. This loss is then used to derive the gradient for the uncertain attention mechanism, thus updating the model.

(3) Transfer learning [Yosinski *et al.*, 2014; Kwon *et al.*, 2024b] is the use of knowledge already learned on one task to improve learning on another task. Its main goal is to quickly transfer what has already been learned to a new domain. a. Metric-based learning: Vinyals et al. proposed a Matching Network, which maps training samples and validation samples to corresponding labels [Vinyals *et al.*, 2016]. Then the samples are mapped into a low dimensional vector space using LSTM to compute the similarity between the new sample and each sample. Finally the predicted labels are output using the kernel density estimation function. b. Meta learning based approach: Finn et al. proposed a model-independent Model Aagnstic Meta Learning (MAML), aiming at obtaining a good general initialization parameter for the model and achieving fast fine-tuning of the model on new tasks [Finn *et al.*, 2017]. The strategy learns features that are applicable to a variety of tasks, allowing the model to quickly adjust the model's weights so that it can be generalized and quickly adapted to new tasks. However, its disadvantage makes it necessary to train with a sufficient number of tasks to converge. For this reason, Sun et al. proposed Meta Transfer Learning (MTL) for Few-shot learning. This method enables MAML to learn only the last layer as a classifier [Sun *et al.*, 2019]. It is first trained on a large-scale dataset to get a full-time deep neural network and fixes the lower-level convolutional layers as feature extractors. The parameters of the feature extractor neurons are subsequently acquired through learning to ensure their rapid adaptability to the Few-shot job. When only a small amount of labeled data is used, the method can help deep neural networks converge quickly and reduce the probability of overfitting occurring. Rusu et al. proposed the LEO method based on MAML to address the uncertainty of its use of Meta learning [Rusu *et al.*, 2018]. The method first initializes the parameters of the model for each task. Its parameters are sampled from a conditional probability distribution associated with the training data. Secondly, the parameters are updated in a low-dimensional hidden space to adapt the model more efficiently.

D.3 Meta Continual Learning

Most of the traditional methods of class Continuous Learning lead to significant performance degradation when applied directly to Meta-CL. Firstly, because of the limited number of samples in the new class during Meta-CL, the model is prone to overfitting into the new class, resulting in a loss of its capacity to generalize to large sample tests. Second is the dilemma of facing a performance tradeoff between the old and new classes. Due to the fact that in the case of very few samples, it is necessary to increase the learning rate as well as enhance the gradient of the new class loss in order to learn the relevant knowledge of the new class. However, this will make it more difficult to maintain the relevant knowledge of the old class. Therefore, researchers address Meta-CL from a new cognitive perspective.

The problem of Meta-CL was raised by Tao et al. [Tao *et al.*, 2020] at first, and they also proposed the TOpology-Preserving knowledge InCrementer (TOPIC) framework. The original version of the neural network was used for unsupervised learning during input. To accomplish the Few-shot supervised incremental learning task for each training task, the research team improved it. The improved supervised neural gas network allows for incremental growth of nodes and edges through competitive learning. Based on this, this study also designed the neural gas robustness loss function that can effectively suppress the forgetting of old categories and the neural gas adaptation function that reduces the overfitting in new categories. The TOPIC framework can alleviate the forgetting of old knowledge by stabilizing the topology of the neural gas network NGs. It also improves representation learning on a small number of new category samples by making the NG grow [Fritzke, 1994]and adapt to new training samples to prevent overfitting to new categories with fewer samples.

To solve the Few-shot incremental learning problem, Cheragian et al. proposed a Semantic-aware Knowledge Distillation approach [Cheraghian *et al.*, 2021]. The main idea is to introduce semantic information in knowledge distillation to accomplish the preservation of old things in the process of Meta-CL. The method first transforms the labels into word vectors. The input image is mapped to the space of its label word vectors to minimize the distance between the two vectors. No additional neural network parameters need to be added as new categories emerge, thus reducing the dependence of network training on the amount of data. Shi et al. proposed a method based on the discovery of flat minima (F2M) [Shi *et al.*, 2021] for model training in incremental Few-shot learning. An average minimum is when the model parameters are in a flat region and when the loss function value is minimized. This flat region prevents the model from overfitting and preserves previously learned knowledge when learning new data. A Continuous Evolutionary Classifier (CEC) was proposed by Zhang et al [Zhang *et al.*, 2021]. The method initially separates the feature extraction module from the classifier, where the feature extraction module is frozen to avoid generating forgetfulness and only the classifier is changed for each incremental task. The process is that whenever vectors for a new category are generated, the vectors obtained from all categories, old and new, will be aggregated together. Where each vector is a node and the nodes are aggregated based on an attention mechanism that utilizes the invariance of the connections between the nodes to maintain the old knowledge. Finally, the graph attention model is utilized to generate the final classifier by combining the information of the emerged categories. A self-promoting prototype refinement mechanism (SPPR) was proposed by Zhu et al [Zhu *et al.*, 2021]. The technique uses the correlation matrix between newly introduced category samples and pre-existing category prototypes to revise the current prototypes. This mechanism is realized by dynamic relational projection. The heart-tired sample representations and the just-tired prototypes are mapped into the same embedding space, and the two embeddings are computed in the above-mentioned distance metric in the space to compute the space of projection matrices between them. Ultimately, this matrix space serves as a weight for prototype refinement to guide the dynamics of the prototypes toward preserving existing knowledge and enhancing the defensibility of new categories.

Recently, Zhou et al. proposed a Forward Compatible Training (FACT) [Zhou *et al.*, 2022]. It makes it easy to merge new categories and facilitates the seamless integration of these categories into the existing model. In order to enable the model to be expandable, they designate many virtual prototypes in advance within the embedding area as reserved space and optimize these virtual circles to push instances of the same class closer together, thus reserving more space for incoming new classes. Subsequently, Constrained Few Shot Incremental Learning (C-FSCIL) was proposed by Hersche et al [Hersche *et al.*, 2022]. The method first creates and updates the average prototype vector so that it serves as the average of the samples in memory. It then polarizes the prototype vector and retraces the fully connected layer for no more than a constant number of iterations. Finally, the average prototype vector is pushed so that it is quasi-orthogonal and remains close to the average prototype at the original site.

Table 3: Performance and computational costs, memory footprint of six representative Meta-CL methods using three network architectures on five datasets of image and audio domains. OOM indicates an out-of-memory issue.

		CNN				ViT			
Dataset	**Method**	**Accuracy**	**Memory**	**Latency**	**Energy**	**Accuracy**	**Memory**	**Latency**	**Energy**
CIFAR100	Pretrained	0.260	39.69MB	305s	1.44KJ	0.119	29.8MB	615s	2.96KJ
	ANML	0.272	39.69MB	309s	1.44KJ	-	-	-	-
	ANML+AIM	0.346	1,093MB	6,390s	29.43KJ	-	-	-	-
	OML	-	-	-	-	0.271	29.8MB	628s	3.03KJ
	OML+AIM	0.311	834.1MB	1,481s	6.84KJ	-	-	-	-
	Raw ANML	0.392	99.5MB	11,424s	52.5KJ	-	-	-	-
	Raw OML	-	-	-	-	0.40	153MB	22,968s	103KJ
	Oracle ANML	0.445	39.93MB	1,866s	8.55KJ	-	-	-	-
	Oracle OML	-	-	-	-	0.361	29.5MB	3,672s	19.4KJ
	Lifelearner	0.452	15.45MB	374s	1.71KJ	-	-	-	-
	Latent OML	-	-	-	-	0.365	36.8MB	764s	3.76KJ
Mini ImageNet	Pretrained	0.258	474.5MB	1,198s	5.5KJ	0.234	336.6 MB	2,056s	11.2KJ
	ANML	0.327	474.5MB	1,152s	5.3KJ	-	-	-	-
	ANML+AIM	0.331	1,562 MB	OOM	OOM	-	-	-	-
	OML	-	-	-	-	0.255	336.6 MB	2,203s	11.4KJ
	OML+AIM	0.187	1,051 MB	1,434s	6.5KJ	-	-	-	-
	Raw ANML	0.429	897.1 MB	185,610s	810KJ	-	-	-	-
	Raw OML	-	-	-	-	0.454	1,334 MB	301,456s	1150KJ
	Oracle ANML	0.438	475.0 MB	3,414s	15.7KJ	-	-	-	-
	Oracle OML	-	-	-	-	0.349	337.7 MB	6,301s	31KJ
	Lifelearner	0.433	512.5 MB	1,343s	6.17kJ	-	-	-	-
	Latent OML	-	-	-	-	0.376	336.6 MB	2,250s	13KJ
		CNN				YAMNet			
GSCv2	Pretrained	0.213	10.16MB	71.1s	0.324KJ	0.061	39.2MB	213.6s	1.08KJ
	ANML	0.429	10.16MB	71.1s	0.324KJ	-	-	-	-
	ANML+AIM	0.710	608.2MB	394.2s	1.8KJ	-	-	-	-
	OML	-	-	-	-	0.329	39.2MB	209.8s	1.08KJ
	OML+AIM	0.649	135.2MB	157.5s	0.72KJ	-	-	-	-
	Raw ANML	0.135	45.72MB	735.6s	3.37KJ	-	-	-	-
	Raw OML	-	-	-	-	0.749	120MB	1,482s	6.85KJ
	Oracle ANML	0.712	10.20MB	569.7s	2.6KJ	-	-	-	-
	Oracle OML	-	-	-	-	0.701	39.7MB	1,685s	14.6KJ
	Lifelearner	0.713	3.40MB	75.6s	0.35KJ	-	-	-	-
	Latent OML	-	-	-	-	0.704	40.9MB	235.8s	0.43KJ
Urban Sound8K	Pretrained	0.182	1,382 MB	2,065s	5.7KJ	0.186	39.2 MB	1,365s	5.4KJ
	ANML	0.596	1,382 MB	1,053s	12.4KJ	-	-	-	-
	ANML+AIM	0.439	2,593 MB	1,098s	4.28KJ	-	-	-	-
	OML	-	-	-	-	0.262	39.2 MB	1,874s	5.36KJ
	OML+AIM	0.385	2,648 MB	167,986s	4.88KJ	-	-	-	-
	Raw ANML	0.667	1,456 MB	1,985s	6.97KJ	-	-	-	-
	Raw OML	-	-	-	-	0.595	139.3 MB	2,965s	7.45KJ
	Oracle ANML	0.710	1,384 MB	1,647s	8.35KJ	-	-	-	-
	Oracle OML	-	-	-	-	0.448	42.4 MB	1,875s	6.36KJ
	LifeLearner	0.650	496 MB	2,086s	4.98KJ	-	-	-	-
	Latent OML	-	-	-	-	0.442	39.2 MB	1,624s	4.37KJ
ESC-50	Pretrained	0.196	1,163MB	1,359s	7.2KJ	0.090	39.8MB	4,052s	19.6KJ
	ANML	0.308	1,163MB	1,374s	7.2KJ	-	-	-	-
	ANML+AIM	0.233	2,305MB	OOM	OOM	-	-	-	-
	OML	-	-	-	-	0.135	39.8MB	3.986s	19.6KJ
	OML+AIM	0.181	2,152MB	OOM	OOM	-	-	-	-
	Raw ANML	0.358	5,005MB	OOM	OOM	-	-	-	-
	Raw OML	-	-	-	-	0.577	210.8MB	28,465s	39.6
	Oracle ANML	0.435	1,167MB	4,120s	20.3KJ	-	-	-	-
	Oracle OML	-	-	-	-	0.442	42.5MB	12,473s	84.2KJ
	Latent ANML	0.381	316.3MB	1,445s	7.8KJ	-	-	-	-
	Latent OML	-	-	-	-	0.458	44.3MB	4,536s	7.4KJ