Dynamic Cluster Memory System

Fei Ye[a], Adrian. G. Bors[1]

[a]*School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, China, E-mail: feiye@uestc.edu.cn*

Abstract

Continual learning has emerged as a critical area in machine learning, enabling models to incrementally assimilate novel information while preserving previously acquired representations. While the majority of existing research emphasizes supervised learning frameworks, these approaches exhibit limited generalizability to real-world contexts where annotated data is scarce or unavailable. In this work, we propose the Dynamic Cluster Memory System (DCMS), an innovative architecture designed to facilitate both supervised and unsupervised continual learning by dynamically generating memory clusters that adapt to shifting data distributions. To enhance the scalability of DCMS, we introduce a memory expansion protocol that quantifies representational gaps between incoming data streams and stored exemplars, leveraging this metric to trigger adaptive memory augmentation. Additionally, we propose a sample selection strategy that judiciously allocates new instances to relevant memory clusters, thereby fostering inter-cluster knowledge heterogeneity while also capturing semantically salient features. To address the challenge of memory saturation, we implement a pruning algorithm that systematically eliminates redundant clusters through graph-based relational analysis, ensuring the retention of diverse data within a constrained memory budget. The DCMS framework is inherently model-agnostic and seamlessly integrates with both supervised and unsupervised learning pipelines. Comprehensive experimental evaluations demonstrate that the proposed memory optimization paradigm achieves state-of-the-art results across a range of continual learning benchmarks.

Keywords: Continual Learning, Generative Modelling, Memory System

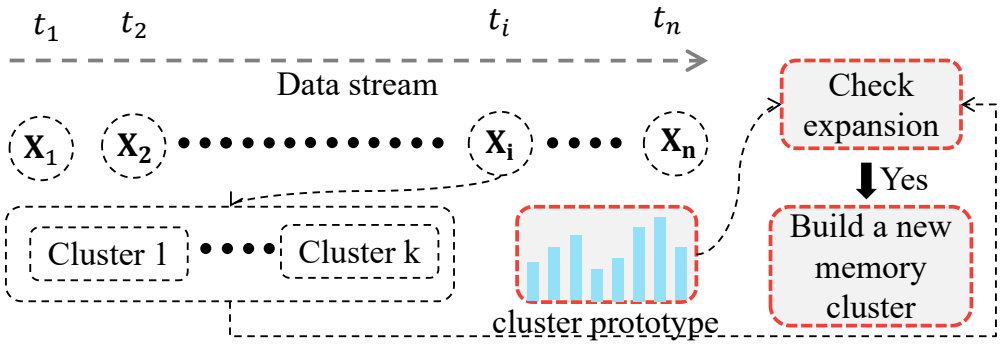


Figure 1: The Dynamic Cluster Memory System (DCMS) pipeline. At the training time t_i, we evaluate the novelty of each new data sample and check the memory expansion process using Eq. (12). Once the memory expansion criterion is satisfied we dynamically create and add new memory clusters into the memory system.

1. Introduction

Modern deep learning methods have obtained significant performance improvements in a wide range of applications [1, 2, 3]. Their efficacy is contingent upon the availability of extensive datasets [4, 5] together with considerable computational resources for learning from such datasets. However, in numerous real-world applications, models typically operate with limited training samples, and prior data is inaccessible. This learning paradigm, termed Continual Learning (CL), has gained prominence within deep learning research [6]. When retraining a model on new data, its parameters are rewritten and the model is not able to recall what it learned before, which is called catastrophic forgetting. Catastrophic forgetting results in significant performance degradation when a model has to assimilate a sequence of tasks, where each task is associated with learning different data.

Numerous studies on continual learning have introduced various methodologies to mitigate catastrophic forgetting, which can be categorized into three primary paradigms: memory replay [7], regularization-based techniques [8], and dynamic network architectures [9]. Memory replay methods typically utilize a fixed-capacity memory buffer to store crucial past examples [7], subsequently retraining the model on both incoming and memorized samples. Regularization-based approaches typically incorporate a penalty term into the primary objective function, thereby regulating parameter updates during optimization [8]. This strategy can be integrated with memory-based methods to enhance continual learning performance. However, both regularization and memory-based methods are constrained

by a fixed model capacity, which limits their efficacy when learning extended task sequences. To address this, dynamic expansion methods incrementally add new sub-networks to augment the model's capacity for adapting to novel tasks, potentially yielding superior performance in learning long task sequences.

While existing continual learning methods show encouraging performances, they require the task information access for memory or architectural expansion. Consequently, these approaches are inadequate for Task-Free Continual Learning (TFCL) that represents a more challenging learning scenario in which the model cannot access any task boundaries and information during training and testing [10]. Recent memory-based methods address TFCL by introducing novel sample selection techniques to retain crucial past examples without considering any task information. However, these methods still require class label access, limiting their utility in unsupervised learning applications. Specifically, image synthesis, representing a critical application in unsupervised learning, remains underexplored in the TFCL scenario [11]. Researching image synthesis within the TFCL framework offers significant advantages: (1) It enables performing many image applications, such as image generation and editing, in an online manner; (2) It enables understanding forgetting behaviors of image generative systems. However, existing continual learning methods cannot be applied in image synthesis, under the TFCL scenario, because supervised signals are not available.

In this paper, we aim to simultaneously implement image synthesis and classification tasks in TFCL by proposing a novel approach called the Dynamic Cluster Memory System (DCMS) which is designed from an entirely different angle from other continual learning methods. Specifically, instead of preserving critical samples into a single memory buffer, as most memory-based continual learning methods do, we organize the memory system into several different memory clusters, each aiming to preserve information corresponding to a unique underlying data distribution, as illustrated in Fig. 1. To achieve this goal, we propose a novel Dynamic Memory Optimization (DMO) that computes the prototype for each memory cluster and evaluates the distance among memory clusters using such cluster prototypes. When incoming samples contain entirely different semantic information with respect to the memorized samples, DMO dynamically builds new memory clusters to store the novel information. This approach manages a compact memory architecture. In order to encourage each memory cluster to capture a unique underlying data distribution, we propose a novel sample selection approach that evaluates the distance between each incoming sample and the prototype of each memory cluster and then decides when to store new data into the relevant memory cluster. The proposed sample selection approach enables the

preservation of diverse semantic information within a fixed memory capacity.

In many practical applications, continual learning models such as DCMS are deployed on resource-constrained device such as those using Field-Programmable Gate Arrays (FPGAs) for example. In such learning scenarios, the proposed DCMS would suffer from memory overload issues, missing new data samples due to capacity constraints. To address this issue, we propose a novel Memory Pruning Process (MPP) that automatically removes non-essential data samples when the proposed DCMS is overloaded. Specifically, MPP constructs and evaluates a graph relation matrix, aiming to find a pair of memory clusters that represent overlapping information and then removes one of them through a diversity evaluation mechanism. This approach can maintain a fixed capacity for the proposed DCMS and enable it to incrementally preserve new information over time. To implement image synthesis in TFCL, we employ the proposed DCMS to optimize a Denoising Diffusion Probabilistic Model (DDPM) [12] for continual image generation.

The primary contributions of this paper are summarized as :

- We propose a novel Dynamic Cluster Memory System (DCMS), for learning classification and image generation TFCL tasks, that can incrementally preserve diverse information without requiring any supervised signals. DCMS can be applied to both supervised and unsupervised learning.

- We propose a novel memory optimization mechanism to incrementally create and add new memory clusters for DCMS by evaluating the novelty of incoming data. This approach can appropriately expand the memory buffer and thus can ensure a compact memory structure.

- This paper introduces a new sample selection approach that evaluates the knowledge similarity between incoming samples and each memory cluster and uses this measure to selectively store new data into the appropriate memory clusters. Thus, the knowledge diversity is further improved within the proposed framework.

- In order to capture new information within an endless data stream, we introduce a new memory pruning method that evaluates a graph relation matrix to find and select knowledge-overlapping memory clusters. Clusters bearing redundant information are then removed.

- The proposed DCMS enables continual image generation using DDPM in the TFCL scenario, which achieves significant performance improvements.

2. Related Work

2.1. Continual Learning

Lifelong/Continual Learning (CL) presents a more complex training challenge compared to conventional deep learning [13]. In particular, the absence of prior data samples in continual learning requires using a fixed-size memory system to retain crucial past examples for memory-replay, [7, 14]. The selection of samples within memory-based methods is critical for model performance, with the goal of strategically storing important data samples, [15]. One typical memory-based method is Reservoir Sampling [16], which randomly stores new data samples into the memory buffer, aiming to preserve critical past information. Gradient Episodic Memory (GEM) [17] aims to add individual constraints based on the old examples for each task. An improved version of GEM called AGEM aims to replace the individual limitations with a global loss of all tasks, which can significantly improve training efficiency. More recently, storing diverse samples for representing the fuzzy task boundary was proposed in [7]. A diversity-aware sampling method is used for optimizing the memory buffer. Furthermore, numerous studies have explored integrating memory-based methods with regularization-based approaches to enhance continual learning performance [18]. However, storing actual data samples in memory raises data privacy concerns. To mitigate this, several studies have proposed training a deep generative model as a memory system capable of storing and replaying past examples [19]. These studies typically use Variational Autoencoders (VAEs) [20] or Generative Adversarial Networks (GANs) [21], with the quality of the generated replay samples heavily dependent on the models' generation capabilities. Another challenge for generative replay-based methods is the potential for mode collapse in generative models like GANs, which can result in poor generated images.

2.2. Online Task-Free CL

The prevailing continual learning paradigm typically operates under the assumption that task boundaries and information are unavailable throughout the optimization process. However, a more realistic learning paradigm, introduced in [10, 22] and termed Task-Free Continual Learning (TFCL), adopts an online learning approach devoid of task-specific information. To mitigate network forgetting in TFCL, a fixed-size memory buffer retaining historical samples was considered in [10], concurrently training the classifier on both memorized and incoming samples. This approach was subsequently enhanced through the development of the Maximal Interfered Retrieval (MIR) criterion [23], which evalu-

ates loss values for critical sample selection. MIR is applicable to both classifier and VAE model training. The Gradient Sample Selection (GSS) [24], addresses network forgetting in TFCL by formulating sample selection as a constrained optimization problem. The Continual Prototype Evolution (CoPE) [25] represents a memory-based approach designed to continually incorporate new data samples while maintaining category balance within the memory system. The Gradient-based Memory Editing (GMED) [26] modifies stored samples to enhance model's generalization. Despite these memory-based methods showing strong performances in TFCL, their efficacy diminishes when increasing the number of data due to model's capacity limitations. To address this, several studies have explored Dynamic Expansion Models (DEM) to incrementally expand model capacity in TFCL. The Continual Unsupervised Representation Learning (CURL) [27], dynamically creates and adds new VAE-based inference networks to capture novel visual concepts from data streams. However, CURL's focus on expanding inference model capacity while maintaining a fixed number of generative model parameters results in suboptimal generation results over long-term data streams. Another DEM-based method introduces a Dirichlet process-based expansion mechanism to control model expansion [28], dynamically increasing model capacity by adding parameters to the generative model and classifier, leading to increasing model complexity [29]. While existing DEM-based methods show promising results in TFCL, their deployment on resource-constrained devices is limited due to requiring expanding numbers of parameters. This paper aims to address this limitation by developing a novel memory system that maintains a fixed memory capacity while maximizing information capture.

2.3. Continual Generative Modeling

Existing research in continual learning primarily focuses on optimizing classifiers for classification tasks, with limited exploration of deep generative models [11]. Implementing continual generative systems offers advantages for applications like image generation and editing, enabling online learning capabilities. Consequently, several studies have investigated deep generative models in continual learning [30, 31, 32]. Early work in continual image generation considers pairing two Variational Autoencoders (VAEs) as a teacher and student, which learn from each other to continually acquire and retain new information [31]. However, VAE-based methods often produce less sharp, lower-quality image generations [33], leading to performance degradation when increasing the number of tasks [33]. This limitation has prompted the exploration of more powerful deep generative models, such as Generative Adversarial Networks (GANs), capable of gener-

ating high-quality images [19, 30, 34, 35]. Shin *et al.* [36] proposed a dual-model framework comprising a deep generative model and a classifier. This architecture mitigates catastrophic forgetting by leveraging the generative model to replay previously encountered data, integrating synthetic samples with new task data when training the classifier. However, this method is limited to classification tasks and fails to learn robust feature representations. Ramapuram *et al.* [30] introduced Lifelong Generative Modeling (LGM), which utilizes a teacher-student paradigm based on the VAE. In this framework, the teacher and student networks iteratively learn from each other to assimilate and retain novel information. Achille *et al.* [31] developed the Variational Autoencoder with Shared Embeddings (VASE) designed to extract disentangled representations by leveraging the Minimum Description Length (MDL) principle, thereby capturing salient latent variables across diverse domains. Despite these advancements, such models often exhibit suboptimal performance with complex data due to inadequate replay fidelity from previously learned distributions. To address this, Seff *et al.* [37] advocated for the use of more expressive generators, such as GANs, within the generative replay paradigm. Nevertheless, GAN-based methods are susceptible to mode collapse, limiting their ability to generate high-quality, diverse historical samples. Furthermore, many continual learning models depend heavily on task-specific cues and memory management strategies. Recent developments in TFCL have introduced innovative mechanisms for updating deep generative models without explicit task boundaries. Notably, Continual Generative Knowledge Distillation (CGKD) [11] presents a dynamic expansion framework wherein each module is a generative model. CGKD employs a Fréchet Inception Distance (FID)-driven expansion protocol, quantifying the divergence between a memory buffer and the outputs of each module to guide model growth. However, the computation of FID scores imposes substantial overhead, particularly when is applied at every training iteration to assess expansion criteria.

While Diffusion Probabilistic Model (DDPM) models have achieved notable success across various domains, their performance is contingent upon the availability of large-scale datasets. DDPM models are prone to catastrophic forgetting, resulting in a decline in the generation quality of previously learned images. Prior studies, such as [38], have investigated the incorporation of DDPM models as generative replay networks within a continual learning framework. Similarly, the application of diffusion models for distillation and replay mechanisms was explored in [39]. However, these methods are computationally expensive, particularly for to the generative replay process. Moreover, they are designed for task-aware continual learning scenarios, restricting their applicability to more realistic settings,

such as Online Task-Free Continual Learning (OTFCL) [10]. In contrast, the proposed Dynamic Cluster Memory System (DCMS) presents an efficient training paradigm for DDPM models within the OTFCL setting, eliminating the need for DDPM sampling and supervised signals.

2.4. Discussion

The main difference between our approach and other methods is that DCMS is designed for the task-free continual learning setting without relying on task boundaries or supervised labels, while many prototype-based continual learning methods construct or update prototypes using class labels, task identities, or classifier feedback. In contrast, DCMS builds memory clusters according to the distributional discrepancy between incoming samples and existing memory prototypes. Therefore, the prototypes in DCMS are not class prototypes, but cluster-level unsupervised representatives used to estimate whether incoming data contain sufficiently novel semantic information.

Compared to CoPE [25], which mainly maintains and evolves class-related prototypes for continual classification, DCMS focuses on dynamically organizing the memory space itself. Specifically, DCMS introduces a dynamic memory expansion mechanism to create new clusters when the discrepancy between incoming samples and existing cluster prototypes exceeds a threshold. This enables the memory system to capture newly emerging distributions without requiring class labels. Moreover, DCMS can be applied not only to classification but also to unsupervised generative learning, such as continual image synthesis with DDPM, which is not the main focus of CoPE.

Compared with general memory clustering methods [40], DCMS is not a static clustering or post-processing strategy. Instead, clustering, memory expansion, sample assignment, and memory pruning are jointly integrated into an online continual learning framework. Each cluster is dynamically created and updated during the data stream, and the proposed pruning mechanism removes redundant or overlapping clusters when the memory budget is exceeded. This design allows DCMS to maintain a compact but diverse memory structure under a fixed memory capacity.

DCMS differs from dynamic memory replay methods [41, 7], in how memory is expanded and controlled. Existing dynamic replay methods typically focus on selecting or replaying representative samples from a memory buffer. DCMS further structures a memory buffer system by considering multiple distribution-aware clusters and uses prototype-based discrepancy estimation to decide where

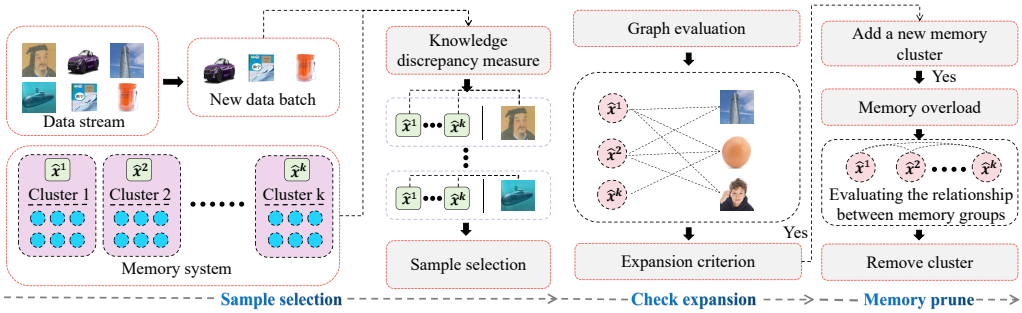


Figure 2: The proposed memory framework operates through three primary stages during each training iteration. Initially, the framework calculates the distance between the centroid of each memory cluster and incoming samples using Eq. (8), with the objective of selectively integrating novel information into the most appropriate memory cluster. Subsequently, the memory expansion criterion is evaluated via Eq. (12); when this criterion is fulfilled, a new memory cluster is instantiated and incorporated into the system. Finally, if the total number of memory clusters exceeds the predefined threshold, $|\mathcal{M}| > |\mathcal{M}|^{Max}$, the redundant or overlapping clusters are removed, thus leveraging the available memory against the learned information.

new samples should be stored and when a new cluster should be created. In addition, the graph-based pruning process explicitly considers inter-cluster redundancy, which helps preserve knowledge diversity while preventing uncontrolled memory growth.

Recently, continual learning has also been investigated in multimodal tasks such as continual visual question answering (CVQA) [42]. MacVQA [42] proposes an adaptive memory allocation strategy together with global noise filtering to alleviate catastrophic forgetting, while HyLoVQA [43] introduces a hypernetwork-based low-rank adaptation mechanism for parameter-efficient continual adaptation. Although these methods demonstrate promising performance for vision-language models, they are specifically designed for supervised CVQA with multimodal annotations. In contrast, the proposed DCMS targets the more general TFCL setting and provides a model-agnostic memory organization mechanism that is applicable to both supervised classification and unsupervised generative learning.

3. Methodology

3.1. Preliminary

Problem statement. Continual learning defines a challenging training paradigm where the model has to be updated on new data samples and cannot access all previously seen examples. Under the continual learning scenario, a training dataset

$\mathcal{D}_c^S = \{\mathbf{x}_j, \mathbf{y}_j\}_{j=1}^{n_t}$ (the c-th dataset) is split into C parts $\mathcal{D}_c^S = \{\mathcal{D}_{c,1}^S, \cdots, \mathcal{D}_{c,C}^S\}$ according to the category information [11], where $\mathbf{x}_j \in \mathcal{X}$, and $\mathbf{y}_j \in \mathcal{Y}$ denote the j-th image, and the associated category label over the data samples $\mathcal{X}$ and category space $\mathcal{Y}$, respectively. Each subset $\mathcal{D}_{c,t}^S$ collects data samples from one or several neighboring categories. Since a model in TFCL does not have task information, we build a data stream consisting of these subsets, expressed as :

$$\mathcal{S} = \{\mathcal{D}_{c,1}^S \cup \mathcal{D}_{c,2}^S \cup \cdots \mathcal{D}_{c,C}^S\}\,. \tag{1}$$

Training a model in a data stream $\mathcal{S}$ can be formulated as an online batch-to-batch learning scenario. Specifically, let us define a set of time intervals $\mathbf{t} = \{t_1, \cdots, t_n\}$, when the training takes place in a continual learning stream, and we divide the data stream $\mathcal{S}$ into n data batches $\{\mathbf{X}_i, \cdots, \mathbf{X}_n\}$, where each data batch contains probabilistically distinct data with respect to others. During the training process at a certain time interval t_i, only a data batch $\mathbf{X}_i$ that contains b samples is available and we cannot access all prior data batches $\{\mathbf{X}_1, \cdots, \mathbf{X}_{i-1}\}$. When the model finishes learning all tasks, we can evaluate the model's performance using the classification accuracy criterion on the testing dataset $\mathcal{D}_c^T$. In an unsupervised learning scenario, none of the data samples from $\mathcal{S}$ has class label. We evaluate the model's performance on the stream $\mathcal{S}$ using the image generation criterion.

In addition to a simple class-incremental learning, this paper also explores a more challenging learning scenario in which the training data could come from different data domains. Under this learning paradigm, we build a data stream by considering data samples from different data domains, expressed as :

$$\mathcal{S} = \{\mathcal{D}_1^S \cup \cdots, \cup \mathcal{D}_m^S\}\,, \tag{2}$$

where m denotes the number of datasets, each representing a domain, that are used in the data stream.

Diffusion model. The Denoising Diffusion Probabilistic Model (DDPM) has become the prevailing methodology in the domain of image generation, due to the superior quality of generated images when compared to GANs [21] and VAEs [20]. DDPM is defined by a dual process: a forward diffusion process that progressively adds noise to a generated image, and a reverse diffusion process that iteratively denoises the data to reconstruct the original image, [12, 44]. The forward diffusion process comprises a Markov chain of noise-adding steps that incrementally perturb the data :

$$q(\tilde{\mathbf{x}}_{1:T} \,|\, \tilde{\mathbf{x}}_0) = \prod_{t=1}^{T} q(\tilde{\mathbf{x}}_t \,|\, \tilde{\mathbf{x}}_{t-1})\,, \tag{3}$$

where $p(\tilde{\mathbf{x}}_0)$ denotes an empirical data distribution, the initial data is represented as $\tilde{\mathbf{x}}_0$. The forward diffusion process is executed over T timesteps. The conditional probability $q(\tilde{\mathbf{x}}_t \,|\, \tilde{\mathbf{x}}_{t-1})$ is formulated as a Gaussian distribution, specifically, $\mathcal{N}(\tilde{\mathbf{x}}_t; \sqrt{1-\beta_t}\tilde{\mathbf{x}}_{t-1}, \beta_t \mathbf{I})$. The variance schedule, denoted as $\{\beta_t \in (0,1) \,|\, t = 1, \cdots, T\}$, controls the diffusion step magnitude. The result of Eq. (3), is a sequence of noisy images, represented as $\{\tilde{\mathbf{x}}_1, \cdots, \tilde{\mathbf{x}}_T\}$.

The estimation of the reverse diffusion process $q(\tilde{\mathbf{x}}_{t-1} \,|\, \tilde{\mathbf{x}}_t)$ is computationally intensive, requiring the entire dataset. To address this intractable optimization problem, a model $p_\theta(\tilde{\mathbf{x}}_{t-1} \,|\, \tilde{\mathbf{x}}_t)$ parameterized by θ, facilitates an approximation of the backward process for Eq. (3), as :

$$p_\theta(\tilde{\mathbf{x}}_{0:T}) = p(\tilde{\mathbf{x}}_T) \prod_{t=1}^{T} p_\theta(\tilde{\mathbf{x}}_{t-1} \,|\, \tilde{\mathbf{x}}_t)\,, \tag{4}$$

where $p_\theta(\tilde{\mathbf{x}}_{t-1} \,|\, \tilde{\mathbf{x}}_t) = \mathcal{N}(\tilde{\mathbf{x}}_{t-1}; \boldsymbol{\mu}_\theta(\tilde{\mathbf{x}}_t, t), \boldsymbol{\Sigma}_\theta(\tilde{\mathbf{x}}_t, t))$ and $p(\tilde{\mathbf{x}}_T) = \mathcal{N}(\mathbf{0}, \mathbf{I})$, where $\boldsymbol{\Sigma}_\theta(\cdot,\cdot)$ and $\boldsymbol{\mu}_\theta(\cdot,\cdot)$ are parameterized via a deep neural network, the training of a diffusion model requires a noise estimator $\boldsymbol{\epsilon}_\theta(\cdot,\cdot)$ to predict $\boldsymbol{\epsilon}$ from $\tilde{\mathbf{x}}_t$ and t, [12]. We utilize the Improved DDPM (IDDPM) objective function [45] to train $\boldsymbol{\epsilon}_\theta$ as :

$$\begin{aligned} \mathcal{L}_{\text{IDDPM}} = \lambda \mathbb{E}_{q(\tilde{\mathbf{x}}_{0:T} \,|\, \tilde{\mathbf{x}}_0)} \Big[& -\log p_\theta(\tilde{\mathbf{x}}_0 \,|\, \tilde{\mathbf{x}}_1) \\ & + \sum\nolimits_{t=1}^{T-2} \{\mathcal{L}_t\} + D_{KL}[q(\tilde{\mathbf{x}}_T \,|\, \tilde{\mathbf{x}}_0) \,||\, p_\theta(\tilde{\mathbf{x}}_T)] \Big] \\ & + \mathbb{E}_{t, \tilde{\mathbf{x}}_0, \boldsymbol{\epsilon}} \Big[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\widehat{\alpha}_t}\tilde{\mathbf{x}}_0 + \sqrt{1-\widehat{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2 \Big]\,, \end{aligned} \tag{5}$$

where $\mathcal{L}_t$ is defined as :

$$\mathcal{L}_t = D_{KL}\big[q(\tilde{\mathbf{x}}_t \,|\, \tilde{\mathbf{x}}_{t+1}, \tilde{\mathbf{x}}_0) \,||\, p_\theta(\tilde{\mathbf{x}}_t \,|\, \tilde{\mathbf{x}}_{t+1})\big]\,, \tag{6}$$

where $\tilde{\mathbf{x}}_T$ signifies a noise vector drawn from a normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$ and $D_{KL}(\cdot,\cdot)$ is the Kullback–Leibler (KL) divergence. The forward and reverse diffusion processes are delineated by the distributions $q(\tilde{\mathbf{x}}_T \,|\, \tilde{\mathbf{x}}_0)$ and $p_\theta(\tilde{\mathbf{x}}_0 \,|\, \tilde{\mathbf{x}}_1)$, correspondingly. $q(\tilde{\mathbf{x}}_t \,|\, \tilde{\mathbf{x}}_{t+1}, \tilde{\mathbf{x}}_0)$ is formulated as in [12] :

$$\begin{aligned} q(\tilde{\mathbf{x}}_t \,|\, \tilde{\mathbf{x}}_{t+1}, \tilde{\mathbf{x}}_0) = \mathcal{N}(\tilde{\mathbf{x}}_t; & \frac{\sqrt{\widehat{\alpha}_t}\beta_{t+1}}{1-\widehat{\alpha}_t}\tilde{\mathbf{x}}_0 \\ & + \frac{\sqrt{\widehat{\alpha}_{t+1}}(1-\widehat{\alpha}_t)}{1-\widehat{\alpha}_t}\tilde{\mathbf{x}}_{t+1}, \hat{\beta}_{t+1}\mathbf{I})\,, \end{aligned} \tag{7}$$

where $\hat{\beta}_{t+1} = \frac{(1-\widehat{\alpha}_t)}{1-\widehat{\alpha}_{t+1}}\beta_{t+1}$, where $\widehat{\alpha}_t := \prod_{s=1}^{t}\{\alpha_s\}$ and $\alpha_t := 1 - \beta_t$, and $\boldsymbol{\epsilon}$ is a random vector drawn from $\mathcal{N}(\mathbf{0}, \mathbf{I})$, we consider $\lambda = 0.001$ in the experiments to attenuate the influence of the first term from (5), as in [45].

3.2. Clustering Memory System

Prior memory buffer-based approaches to continual learning often leverage class or task metadata [46, 47] or depend on classifier optimization feedback [48], which is not applicable to unsupervised learning scenarios. We introduce a novel memory system buffer designed to enable the training of heterogeneous models for both supervised and unsupervised learning within the TFCL paradigm, without requiring network architectural alterations. Given the absence of class and task information during TFCL learning, the preservation of diverse samples within a bounded memory buffer is critical for mitigating catastrophic forgetting. We propose a novel dynamic clustering memory system that adaptively creates new clusters to capture temporal shifts in data distributions.

Let us define $\mathcal{M}$ as a dynamic memory system and we assume that $\mathcal{M}$ consists of k memory clusters $\mathcal{M} = \{\mathcal{M}^1, \cdots, \mathcal{M}^k\}$ at a certain training time t_i. Specifically, each memory cluster $\mathcal{M}^j$ has a fixed capacity, denoted as π, which represents the maximum number of memorized samples. For each memory cluster $\mathcal{M}^j$ we aim to define a prototype sample $\widehat{\mathbf{x}}^j$, representing a compact knowledge representation for that cluster. As a result, we can effectively evaluate the knowledge similarity among memory clusters using these prototypes.

Before we introduce the proposed dynamic memory expansion process, we first propose a new distance measure approach called Knowledge Discrepancy Measure (KDM) that calculates the discrepancy score between two data samples. The KDM approach the discrepancy score between two data samples $\mathbf{x}_g$ and $\mathbf{x}_h$ is defined as :

$$F_{\mathrm{KDM}}(\mathbf{x}_g, \mathbf{x}_h) = F_d(G_t(\mathbf{x}_g), G_t(\mathbf{x}_h)) \,, \tag{8}$$

where $G_t(\cdot)$ denotes an information representation function. $F_d(\cdot)$ represents a distance measure implemented using a loss function or a probability distance. In this paper, we consider the Jensen–Shannon divergence and the square loss to implement the KDM criterion.

Square Error (SE)-based KDM. One of the most commonly used distance measures is the square error that is used to evaluate the discrepancy between two vectors and has been widely applied as the cost function for deep generative models such as DDPM [12] and the Variational Autoencoder (VAE) [20]. In this study, $G_t(\cdot)$ is implemented as an identity function that always outputs the original input $\mathbf{x}_g = G_t(\mathbf{x}_g)$, while $F_d(\cdot, \cdot)$ is implemented using the square error function. As a result, the process of the proposed SE-based KDM is defined by :

$$F^{\mathrm{SE}}_{\mathrm{KDM}}(\mathbf{x}_g, \mathbf{x}_h) = \sum\nolimits_{j=1}^{d'} (\mathbf{x}_g[j] - \mathbf{x}_h[j])^2 \,, \tag{9}$$

where $\mathbf{x}_g[j]$ represents the j-th dimension of $\mathbf{x}_g$ and d' denotes the dimension of the data. We call the proposed Dynamic Cluster Memory System (DCMS) using the SE-based KDM (DCMS-SE).

Jensen–Shannon (JS) divergence-based KDM. The JS divergence is mainly applied to measure the discrepancy between two probability distributions [20]. However, we cannot directly apply the JS divergence to implement $F_d(\cdot)$ in Eq. (8) because the target distributions has no explicit function. In order to address this issue, we propose to implement the function $G_t(\cdot)$ using a forward diffusion process that can transfer the example $\mathbf{x}_g$ to a Gaussian distribution $q(\tilde{\mathbf{x}}_g^{t^\star} \mid \mathbf{x}_g) = \mathcal{N}(\tilde{\mathbf{x}}_g^{t^\star}; \sqrt{\widehat{\alpha}_{t^\star}}\mathbf{x}_g, (1-\widehat{\alpha}_{t^\star})\mathbf{I})$, where $t^\star$ denotes a hyper-parameter to decide the forward steps. In this paper, $t^\star$ is set to be small enough in order to allow the transformed distribution $q(\tilde{\mathbf{x}}_g^{t^\star} \mid \mathbf{x}_g)$ to closely approximate the real data distribution $p(\mathbf{x}_g)$. As a result, the distance measure function $F_d(\cdot,\cdot) = D_{\mathrm{JS}}(\cdot \,||\, \cdot)$ is implemented using the JS divergence and the KDM criterion is expressed as :

$$F_{\mathrm{KDM}}^{\mathrm{JS}}(\mathbf{x}_g, \mathbf{x}_h) = D_{\mathrm{JS}}[q(\tilde{\mathbf{x}}_g^{t^\star} \mid \mathbf{x}_g) \,||\, q(\tilde{\mathbf{x}}_h^{t^\star} \mid \mathbf{x}_h)]\,. \tag{10}$$

The diffusion step $t^\star$ plays a role analogous to a controllable smoothing kernel in representation learning. A small $t^\star$ is used to ensure that the transformed distribution remains close to the data manifold while introducing sufficient stochasticity to suppress pixel-level sensitivity. This design allows KDM to reflect structural rather than purely pixel-wise differences. In this sense, $t^\star$ controls the trade-off between local fidelity and global distributional smoothing. We further analyze its impact in the ablation study.

Unlike pixel-wise distances, JS divergence is computed in a stochastic representation space induced by diffusion, where each sample is mapped to a local Gaussian neighborhood around the data manifold. This allows the discrepancy measure to capture distributional consistency rather than exact pixel alignment, making it more suitable for detecting semantic-level shifts in streaming data. We call the proposed DCMS using the JS divergence-based KDM as DCMS-JS. In the following section, we introduce a new dynamic memory optimization approach that is used to expand the memory system when new information is detected.

3.3. Dynamic Memory Optimization

To incrementally capture and learn data distribution shifts, it is necessary to develop mechanisms that dynamically build and add new memory clusters into the proposed DCMS framework when necessary. The optimization goal is to preserve the current knowledge while learning new information. This leads to the

acquisition of diverse knowledge with a compact memory capacity. To implement this goal, we can calculate the knowledge similarity between each incoming sample and all data samples stored in the memory clusters using the KDM distance measure from Eq. (8). A large KDM distance measure indicates a signal for expanding the memory system. However, evaluating this distance for every sample from each memory cluster is computationally expensive, and that's why we instead consider the distances between new data samples and the cluster prototypes $\widehat{\mathbf{x}}^k$, calculated as the vector median [49] :

$$c^\star = \underset{c=1,\cdots,|\mathcal{M}^k|}{\arg\min} \sum_{j=1,c\neq j}^{|\mathcal{M}^k|} F_{\mathrm{KDM}}(\mathcal{M}^k[c], \mathcal{M}^k[j]) \,, \tag{11}$$

where $|\mathcal{M}^k|$ represents the number of existing examples stored in $\mathcal{M}^k$ and $F_{\mathrm{KDM}}(\cdot)$ is provided in Eq. (8). We consider the cluster prototype $\widehat{\mathbf{x}}^k = \mathcal{M}^k[c^\star]$, representing that datum from the cluster with the shortest distance with respect to all other examples from the memory cluster $\mathcal{M}^k$. This represents a robust estimation of the cluster prototype, which removes the outliers, and is defined only once for each cluster in order to reduce computational costs. The prototype is computed when a memory cluster is created and remains fixed during subsequent updates. This design intentionally trades a small amount of representational flexibility for significantly lower computational complexity, as repeatedly recomputing the vector median after every sample insertion or removal would incur substantial computation overhead. Since the proposed sample assignment strategy continuously allocates semantically similar samples to each cluster, the cluster distribution evolves gradually, allowing the prototype to remain a stable representative in practice.

Then we employ the cluster prototypes to estimate the expansion signal, as :

$$\min_{j=1,\cdots,b} \{F_{\mathrm{KDM}}(\mathbf{x}_{i,j}, \widehat{\mathbf{x}}^1), \cdots, F_{\mathrm{KDM}}(\mathbf{x}_{i,j}, \widehat{\mathbf{x}}^k)\} > \lambda, \tag{12}$$

where $\mathbf{x}_{i,j}$ represents the j-th data sample collected from the data batch $\mathbf{X}_i$ drawn from S at the training time t_i. λ represents an expansion threshold used to regulate the memory expansion process. When using a small threshold λ, the proposed memory expansion mechanism frequently creates new memory clusters, resulting in memory overload. In contrast, for a large threshold λ, the proposed approach slowly expands the memory system, ensuring a compact memory structure. Each time when the memory expansion criterion defined in Eq. (12) is satisfied at a certain training time t_i, we build and add a new memory cluster $\mathcal{M}^{k+1}$ into the memory system while $\mathbf{x}_{i,j^\star}$ is used as the prototype of $\mathcal{M}^{k+1}$, where $j^\star = \arg\max_{j=1,\cdots,b}\{F_{\mathrm{KDM}}(\mathbf{x}_{i,j}, \widehat{\mathbf{x}}^1), \cdots, F_{\mathrm{KDM}}(\mathbf{x}_{i,j}, \widehat{\mathbf{x}}^k)\}$.

3.4. Sample Selection

Most memory-replay-based methods [46, 47] address only supervised or task-aware continual learning scenarios, which consider class label or task information to implement the sample selection strategy [50, 51, 52, 53]. However, these methods do not consider image synthesis in continual learning. To address image synthesis in TFCL, we introduce a novel sample selection method that does not require any supervised signals. Let us assume that at a certain training time t_i we have a memory system defined by k memory clusters $\mathcal{M} = \{\mathcal{M}^1, \cdots, \mathcal{M}^k\}$. When we get a new data batch $\mathbf{X}_i$ at the i-th training time t_i, we evaluate the knowledge similarity between each new sample $\mathbf{x}_{i,j} \in \mathbf{X}_i$, $j = 1, \ldots, |\mathbf{X}_i|$, where $|\mathbf{X}_i|$ is the total number of samples from the data batch $\mathbf{X}_i$, and each cluster prototype $\widehat{\mathbf{x}}^c$, evaluated by Eq. (11), as :

$$d^\star = \arg\min_{c=1}^{k} F_{\mathrm{KDM}}(\mathbf{x}_{i,j}, \widehat{\mathbf{x}}^c),\ \mathbf{x}_{i,j} \in \mathcal{M}^{d^\star}, \tag{13}$$

where $\mathcal{M}^{d^\star}$ represents the chosen memory cluster. This procedure ensures adding novel samples to those memory clusters with which they are compatible. Once the number of memorized samples reaches the maximum memory cluster capacity size $|\mathcal{M}|^{Max}$, we remove the data samples with the shortest distance with respect to the cluster prototype $\widehat{\mathbf{x}}^k$:

$$e^\star = \arg\min\nolimits_{e=1,\cdots,|\mathcal{M}^k|} \{F_{\mathrm{KDM}}(\mathbf{x}'_{k,e}, \widehat{\mathbf{x}}^k)\}\,, \tag{14}$$

where $e^\star$ is the index of the sample that is removed from $\mathcal{M}^k$. This sample-removing procedure enables the continual refresh of the memorized samples with new ones, thus adapting well to shifts in data distributions.

3.5. Memory Pruning

Memory-based continual learning systems, especially those deployed on resource-constrained devices, have to consider their memory capacity limitations. Through the threshold λ from Eq. (12) we ensure the memory expansion process, for the proposed DCMS. However, this can lead to memory overload and then DCMS would not able to learn new information during the subsequent training. To address this issue, we propose a novel Memory Pruning Procedure (MPP) that aims to automatically find and remove memory clusters storing non-essential information. The primary idea of the proposed MPP is that if several memory clusters contain significant overlapping information, we can keep only a single memory

cluster while removing the redundant ones. Let us define $M \in [1, 20]$ as a predefined maximum number of memory clusters, considering also the maximum memory cluster capacity $|\mathcal{M}|^{Max}$ according to the device's resource constraints. Let us assume that the proposed DCMS, through the increase in the number of clusters, reaches $k > M$ memory clusters. The proposed MPP first evaluates a relation matrix $\mathbf{B} \in \mathbf{R}^{k \times k}$ describing the correlation between memory clusters, where $B[g, h]$ represents the relation score between the g-th and h-th memory cluster, respectively. Each $B[g, h]$ is calculated using $1/F_{\text{KDM}}(\widehat{\mathbf{x}}^g, \widehat{\mathbf{x}}^h)$ and $B[g, h] = B[h, g]$. By using this relationship matrix, we find a pair of memory clusters that contain significant overlapping information through :

$$F_{\text{select}}(\mathbf{B}) = \underset{\{\{g,h\} \,|\, g,h=1,\cdots,k, g \neq h\}}{\arg\max} B[g, h]\,, \tag{15}$$

where $\{g^\star, h^\star\} = F_{\text{select}}(\mathbf{B})$ denotes the indices of chosen memory clusters $\{\mathcal{M}^{g^\star}, \mathcal{M}^{h^\star}\}$. Removing one of the memory clusters $\{g, h\}$ preserves the knowledge discrepancy among the remaining ones. To achieve this goal, we propose a novel discrepancy evaluation function that selectively removes one of these selected memory clusters, expressed as :

$$F_{\text{dis}}(g^\star, \mathbf{B}) = \sum_{\substack{m=1 \\ m \neq g^\star}}^{k} \mathbf{B}[g^\star, m]\,. \tag{16}$$

where $s_g = F_{\text{dis}}(g^\star, \mathbf{B})$ and $s_h = F_{\text{dis}}(h^\star, \mathbf{B})$ denotes the discrepancy score for the chosen memory clusters $\mathcal{M}^{g^\star}$ and $\mathcal{M}^{h^\star}$. If the criterion is met $s_g > s_h$ then we delete $\mathcal{M}^h$ from the memory system $\mathcal{M}$, otherwise, we remove $\mathcal{M}^g$ from the memory system $\mathcal{M}$. Eq. (16) can ensure that removing a memory cluster maintains the diverse knowledge within the proposed DCMS.

3.6. Dynamic Expansion Model

Learning a single and static model can be deployed on a resource-constrained device but would suffer from performance degeneration when the model would aim to learn a long data stream. To maintain the performance on a long-term data stream, we propose to extend the DCMS to a dynamic expansion framework by developing a novel dynamic expansion architecture. Specifically, each component of the dynamic expansion framework is implemented using an inference model $\mathcal{A}_i = \{p_{\phi_i}(\mathbf{x} \,|\, \mathbf{z}), q_{\varphi_i}(\mathbf{z} \,|\, \mathbf{x}) \,||\, p(\mathbf{z})\}$ and a prediction model $f_{\theta_i}(\mathbf{x})$, where

i denotes the component index. The goal of the VAE modules is of checking the expansion process as well as choosing an appropriate component to be used for a given testing sample at the inference phase. Since the VAE module associated with each component can yield generations that represent the knowledge learned by the associated component, we can implement the efficient model expansion process at t_i by :

$$\min\{f_{\mathrm{MMD}}(\mathbf{Z}'_j, \mathbf{Z}_i) \,|\, j = 1, \cdots, k-1\} > \lambda^{\mathbf{expand}}, \tag{17}$$

where $f_{\mathbf{MMD}}(\cdot,\cdot)$ is the Maximum Mean Discrepancy (MMD) [54], while $\lambda^{\mathrm{expand}}$ is a threshold used to control the model expansion. $f_{\mathrm{MMD}}(\cdot,\cdot)$ is defined as :

$$\begin{aligned} f_{\mathrm{MMD}}(\mathbf{Z}'_j, \mathbf{Z}_i) = & \frac{1}{b(b-1)} \sum\nolimits_{a \neq g}^{b} \Big\{ f(\mathbf{z}_a^i, \mathbf{z}_g^i) \\ & + f(\mathbf{z'}_a^j, \mathbf{x'}_g^j) - f(\mathbf{z}_a^i, \mathbf{z'}_g^j) - f(\mathbf{z'}_g^j, \mathbf{z}_a^i) \Big\}, \end{aligned} \tag{18}$$

where $\mathbf{Z}'_j = \{\mathbf{z'}_1^j, \cdots, \mathbf{z'}_b^j\}$ denotes the latent variables yielded by the VAE model of the j-th component, where $b = 10$ represents the batch size. $\mathbf{Z}_i = \{\mathbf{z}_1^i, \cdots, \mathbf{z}_b^i\}$ denotes latent representations resulting from feeding the data batch $\mathbf{X}_i$ into the inference model $q_{\varphi_k}(\mathbf{z} \,|\, \mathbf{x})$ of the current component at t_i. The proposed dynamic expansion model using the JS-divergenced-based KDM is named Dynamic-JS.

3.7. Redundant Expert Removing Mechanism

Dynamic expansion architectures progressively instantiate additional expert modules to accommodate learning novel information during emerging tasks and facilitate continual learning. However, such adaptive expansion strategies are impractical for deployment on resource-limited hardware due to the increase in model's parameter count [55]. To mitigate this challenge, we introduce an Expert Pruning Mechanism (EPM) designed to automatically eliminate redundant experts exhibiting highly overlapping knowledge representations with other components. The core concept of the proposed EPM involves constructing a relational graph structure to identify and prune experts with significant informational overlap, thereby freeing capacity for future learning phases. Specifically, the proposed EPM constructs and assesses a component relationship matrix $\mathbf{W}^i \in \mathbb{R}^{k \times k}$ that encodes the inter-expert relationships, where k represents the current number of experts and the superscript i indicates the matrix's update at the i-th training iteration. The element $w_{a,c}^i$ quantifies the relationship between the a-th and c-th experts, defined by :

$$w_{a,c}^i = \frac{1}{F_{\mathrm{dis}}(P_{\mathcal{A}_a}, P_{\mathcal{A}_c})}, \tag{19}$$

where $P_{\mathcal{A}_a}$ and $P_{\mathcal{A}_c}$ represent the probability distributions derived from the samples generated by the a-th and c-th experts, respectively. $F_{\text{dis}}(\cdot)$ denotes a generic divergence metric, implemented via Mean Squared Error (MSE) for computational efficiency. The proposed EPM identifies a pair of experts with the highest mutual information similarity using the similarity matrix $\mathbf{W}^i$, as :

$$F_{\text{find}}(\mathbf{W}^i) = \underset{a,c=1,\cdots,k,a\neq b}{\text{argmax}} \left\{ w^i_{a,c} \right\}, \tag{20}$$

where $\{a^\star, b^\star\} = F_{\text{find}}(\mathbf{W}^i)$ indicates the indices of the chosen expert pair. Eliminating experts decreases the model's complexity and reduces redundant information in the continual learning system. For preserving knowledge diversity following the expert pruning process, we introduce a diversity assessment function to compute the diversity score for each selected expert, defined as:

$$F_{\text{diver}}(\mathcal{A}_{a^\star}, \mathcal{A}_{c^\star}) = \frac{1}{k-1} \sum\nolimits_{s=1,s\neq c^\star}^{k} w^i_{a,s}\,, \tag{21}$$

where $s_{a^\star} = F_{\text{diver}}(\mathcal{A}_{a^\star}, \mathcal{A}_{c^\star})$ and $s_{c^\star} = F_{\text{diver}}(\mathcal{A}_{c^\star}, \mathcal{A}_{a^\star})$ represent the diversity metrics for the respective expert modules $a^\star$ and $c^\star$. If $s_{a^\star} > s_{c^\star}$, then removing $\mathcal{A}_{c^\star}$ preserves the diversity of knowledge across the remaining experts; otherwise, $\mathcal{A}_{a^\star}$ is eliminated. To facilitate the deployment of the proposed framework on resource-constrained devices, the proposed ERM incorporates a predefined hyperparameter M, which constrains the maximum number of expert modules. If the current number of experts $k > M$, the expert pruning process is triggered to reduce model complexity and allocate additional capacity for subsequent learning phases while maintaining a compact architecture. We call the proposed dynamic expansion framework (Dynamic-JS) using EPM as DCMS-EPM.

The Difference Between EPM and MPP. The Memory Pruning Process (MPP) introduced in Section 3.5 and the Expert Pruning Mechanism (EPM) discussed in this section adopt a similar graph-based redundancy evaluation principle. However, they are designed for completely different architecture components. While MPP is used in the standard DCMS framework to remove redundant memory clusters, EPM is introduced only for the dynamic expansion component architecture version, denoted as DCMS-EPM, to remove redundant expert modules. Therefore, MPP and EPM do not operate jointly on the same component within the same framework. Instead, they serve as independent redundancy-control mechanisms for different settings : MPP maintains a compact memory structure in the standard DCMS, while EPM controls the number of expert modules in the dynamic expansion variant.

3.8. Algorithm Implementation

In this section, we describe the proposed DCMS algorithm under the TFCL framework in detail. Fig. 2 illustrates the overall optimization workflow, where each training iteration involves four essential steps :

Step 1 (Sample selection). At the beginning of training, the first memory cluster is initialized as $\mathcal{M}^1 = \mathbf{X}_1$. The prototype (central sample) $\widehat{\mathbf{x}}^1$ of $\mathcal{M}^1$ is computed according to Eq. (11). For subsequent training stages at time t_i ($i > 1$), a new data batch $\mathbf{X}i$ is introduced. The sample selection strategy defined in Eq. (13) assigns each incoming sample $\mathbf{x}i, j \in \mathcal{X}_i$ to a suitable memory cluster.

Step 2 (Model expansion check). During training, we evaluate the memory expansion condition specified in Eq. (12). Once the condition is satisfied, a new memory cluster $\mathcal{M}^{k+1}$ is dynamically created. The selected sample $\widehat{\mathbf{x}}_{i,j}$ is then used as the prototype of this newly formed cluster.

Step 3 (Memory pruning). When the total number of stored samples exceeds the predefined memory capacity, *i.e.* $|\mathcal{M}| > |\mathcal{M}|^{\text{Max}}$, redundant clusters with overlapping information are removed based on Eq. (15) and Eq. (16).

Step 4 (Training process). We optimize the parameters of the DDPM model $\boldsymbol{\epsilon}_\theta(\cdot, \cdot)$ on memorized samples using Eq. (5).

4. Probabilistic Analysis of the Proposed Memory System

Although recently, some continual learning methods have shown good results in various applications, a comprehensive theoretical analysis explaining the underlying performance improvement is lacking. In this section, we address this deficiency by formulating an innovative theoretical construct to rigorously examine the performance dynamics of generative models operating under TFCL. Specifically, the Evidence Lower Bound (ELBO), a fundamental objective function in deep generative architectures such as Variational Autoencoders (VAEs) [20], is employed as the benchmark for evaluating image generation performance [33, 56]. The proposed framework introduces a novel ELBO formulation tailored for non-stationary data streams.

Although the experimental evaluation is conducted using DDPM, the theoretical analysis presented in this section is intentionally formulated using the variational inference framework which can be generalized to other models. The objective of this analysis is not to derive the optimization objective of DDPM, but rather to analyze the proposed Dynamic Cluster Memory System (DCMS), which is independent of the underlying generative model. Since DCMS organizes

and updates memory distributions for memory replay, its theoretical properties depend primarily on the relationship between the memory distribution and the underlying data distribution rather than on the specific likelihood model. Therefore, the following analysis characterizes the proposed memory mechanism in a model-agnostic manner and is applicable to likelihood-based generative models, including both VAEs and diffusion models.

4.1. Theoretical analysis for a model with a single memory buffer

Prior works have examined the utilization of a unified memory buffer to boost the performance of static generative models under the TFCL framework, with the objective of improving image generation. To analyze this process, we derive a lower bound that captures the variation in model performance across successive training iterations. Before introducing the detailed theoretical analysis results, we provide the necessary notations and definitions as follows.

Definition 1. For a given data stream $\mathcal{S}$, let us define P_i as the distribution of all previously seen data batches $\{\mathbf{X}_1, \cdots, \mathbf{X}_i\}$ at t_i. Let V_{θ^i} be a VAE model with the parameter set θ_i updated at t_i and $\mathcal{M}_i$ a single memory buffer updated at t_i.

Definition 2. The VAE model V_{θ_i} consists of a decoder $p_{\theta_i}(\mathbf{z} \,|\, \mathbf{x})$ and an encoder $q_{\omega_i}(\mathbf{x} \,|\, \mathbf{z})$. Training a VAE model involving the Evidence Lower Bound (ELBO) [20], defined as :

$$\mathcal{L}_{ELBO}(\mathbf{x}; \theta_i, \omega_i) = \mathbb{E}_{q_{\omega_i}(\mathbf{z} \,|\, \mathbf{x})}[\log p_{\theta_i}(\mathbf{x} \,|\, \mathbf{z})] + D_{KL}(q_{\omega_i}(\mathbf{z} \,|\, \mathbf{x}) \,||\, p(\mathbf{z}))\,, \tag{22}$$

where $p(\mathbf{z})$ is a prior distribution.

Theorem 1. The VAE model V_{θ_i} is trained on the data stream using a memory buffer $\mathcal{M}_i$ at t_i. We derive a lower bound for the VAE model at t_i, expressed as :

$$\begin{aligned} \mathbb{E}_{P_i}\big[\log p_{\theta_i}(\mathbf{x})\big] \geq \mathbb{E}_{P_{\mathcal{M}_i}}\big[\log p_{\theta_i}(\mathbf{x})\big] - D_f\big(P_i || P_{\mathcal{M}_i}\big) \\ - \mathcal{F}_{\mathrm{L}}\big(P_{\mathcal{M}_i}, P_i, P_{\theta_i}\big) + \mathcal{F}'_{\mathrm{L}}\big(P_i, P_{\mathcal{M}_i}\big)\,, \end{aligned} \tag{23}$$

where $P_{\mathcal{M}_i}$ denotes the distribution of the memory buffer $\mathcal{M}_i$ updated at t_i. The term $\mathcal{F}_{\mathrm{L}}\big(P_{\mathcal{M}_i}, P_i, P_{\theta_i}\big)$ is defined as :

$$\mathcal{F}_{\mathrm{L}}\big(P_{\mathcal{M}_i}, P_i, P_{\theta_i}\big) \triangleq \big|D_{KL}\big(P_{\mathcal{M}_i} || P_{\theta_i}\big) - D_{KL}\big(P_i || P_{\theta_i}\big)\big|\,. \tag{24}$$

while $\mathcal{F}'_{\mathrm{L}}\big(P_i, P_{\mathcal{M}_i}\big)$ is expressed as :

$$\mathcal{F}'_{\mathrm{L}}\big(P_i, P_{\mathcal{M}_i}\big) \triangleq \mathbb{E}_{P_i}\big[p_i(\mathbf{x}) \log p_i(\mathbf{x})\big] - \mathbb{E}_{P_{\mathcal{M}_i}}\big[p_{\mathcal{M}_i}(\mathbf{x}) \log p_{\mathcal{M}_i}(\mathbf{x})\big]\,, \tag{25}$$

where $p_i(\mathbf{x})$ and $p_{\mathcal{M}_i}(\mathbf{x})$ denote the probability density functions for P_i and $P_{\mathcal{M}_i}$, respectively. From Eq. (23), we can derive a new Evidence Lower Bound (ELBO) as :

$$\begin{aligned}\mathbb{E}_{P_i}\big[\log p_{\theta_i}(\mathbf{x})\big] \geq \mathbb{E}_{P_{\mathcal{M}_i}}\big[\mathcal{L}_{ELBO}(\mathbf{x};\theta_i,\omega_i)\big] \\ - D_f\big(P_i||P_{\mathcal{M}_i}\big) - \mathcal{F}_{\mathrm{L}}\big(P_{\mathcal{M}_i},P_i,P_{\theta_i}\big) \\ + \mathcal{F}'_{\mathrm{L}}\big(P_i,P_{\mathcal{M}_i}\big)\,,\end{aligned} \tag{26}$$

Proof. We derive the following inequality, after considering the F-divergence term $D_f(P_i||P_{\mathcal{M}_i})$ and two KL divergence terms $D_{KL}(P_{\mathcal{M}_i}||P_{\theta_i})$ and $D_{KL}(P_i||P_{\theta_i})$:

$$\begin{aligned}D_{KL}(P_i||P_{\theta_i}) \leq D_{KL}(P_{\mathcal{M}_i}||P_{\theta_i}) + |D_{KL}(P_{\mathcal{M}_i}||P_{\theta_i}) \\ - D_{KL}(P_i||P_{\theta_i})| + D_f(P_i||P_{\mathcal{M}_i})\,.\end{aligned} \tag{27}$$

The last two terms on the right-hand side ensure that Eq. (27) is held. We rewrite $D_{KL}(P_i||P_{\theta^i})$ and $D_{KL}(P_{\mathcal{M}_i}||P_{\theta^i})$ as :

$$D_{KL}(P_i||P_{\theta^i}) = \mathbb{E}_{P_i}[p_i(\mathbf{x})\log p_i(\mathbf{x})] - \mathbb{E}_{P_i}[\log p_{\theta_i}(\mathbf{x})]\,, \tag{28}$$

$$D_{KL}(P_{\mathcal{M}_i}||P_{\theta_i}) = \mathbb{E}_{P_{\mathcal{M}_i}}[p_{\mathcal{M}_i}(\mathbf{x})\log p_{\mathcal{M}_i}(\mathbf{x})] - \mathbb{E}_{P_{\mathcal{M}_i}}[\log p_{\theta_i}(\mathbf{x})]\,. \tag{29}$$

Then we replace Eq. (28) and Eq. (29) into Eq. (27), resulting in :

$$\begin{aligned}&\mathbb{E}_{P_i}[p_i(\mathbf{x})\log p_i(\mathbf{x})] - \mathbb{E}_{P_i}[\log p_{\theta_i}(\mathbf{x})] \leq \\ &\mathbb{E}_{P_{\mathcal{M}_i}}[p_{\mathcal{M}_i}(\mathbf{x})\log p_{\mathcal{M}_i}(\mathbf{x})] - \mathbb{E}_{P_{\mathcal{M}_i}}[\log p_{\theta_i}(\mathbf{x})] \\ &+ |D_{KL}(P_{\mathcal{M}_i}||P_{\theta_i}) - D_{KL}(P_i||P_{\theta_i})| \\ &+ D_f(P_i||P_{\mathcal{M}_i})\,.\end{aligned} \tag{30}$$

We rearrange Eq. (30) as :

$$\begin{aligned}&- \mathbb{E}_{P_i}[\log p_{\theta_i}(\mathbf{x})] \leq \mathbb{E}_{P_{\mathcal{M}_i}}[p_{\mathcal{M}_i}(\mathbf{x})\log p_{\mathcal{M}_i}(\mathbf{x})] \\ &- \mathbb{E}_{P_i}[p_i(\mathbf{x})\log p_i(\mathbf{x})] - \mathbb{E}_{P_{\mathcal{M}_i}}[\log p_{\theta_i}(\mathbf{x})] \\ &+ |D_{KL}(P_{\mathcal{M}_i}||P_{\theta_i}) - D_{KL}(P_i||P_{\theta_i})| + D_f(P_i||P_{\mathcal{M}_i})\,.\end{aligned} \tag{31}$$

We then multiply with -1 in both Left Hand Side (LHS) and Right Hand Side (RHS) of Eq. (31) and obtain Eq. (23).

Remark. We have several observations from Theorem 1 :

- The F-divergence term $D_f(P_i\|P_{\mathcal{M}_i})$ in Eq. (26) is essential for guaranteeing the model's robust generalization capabilities. For instance, reducing the F-divergence $D_f(P_i\|P_{\mathcal{M}_i})$ can elevate the right-hand side of Eq. (26), which consequently leads to an increase in the left-hand side of the same equation.

- When the data distribution P_i aligns with the memory distribution $P_{\mathcal{M}_i}$, Eq. (26) reduces to the conventional Evidence Lower Bound (ELBO).
- The quality of stored samples significantly impacts the model's generalization capability. If the rehearsal buffer $\mathcal{M}_i$ lacks numerous essential historical samples, the F-divergence term $D_f(P_i \,||\, P_{\mathcal{M}_i})$ becomes large, increasing the left-hand side of Eq. (26), which affects the network's overall performance, leading to model's forgetting.

4.2. Theoretical analysis of the memory expanding model

Compared to other memory-based methods, which usually collect and preserve past examples into a single unified memory buffer, the proposed DCMS framework incrementally optimizes and creates several memory clusters to capture the dynamically changing probabilistic data representation. Each memory cluster from the proposed DCMS aims to represent a unique underlying data distribution, allowing to preserve diverse information using a compact memory architecture. In this section, we extend the theoretical results from Theorem 1 to derive a new bound for analyzing the performance of the proposed DCMS framework.

Definition 3. Let P_{M_k} denote the empirical probability distribution represented by the k-th memory cluster M_k. The overall memory distribution is defined as

$$P_M(\mathbf{x}) = \sum\nolimits_{k=1}^{K} \pi_k P_{M_k}(\mathbf{x}), \tag{32}$$

where $\sum_{k=1}^{K} \pi_k = 1$ and $\pi_k = \frac{|M_k|}{\sum_j |M_j|}$. Therefore, P_{M_k} represents the empirical distribution induced by all memory clusters. Let us define $P_{\mathcal{M}^1} \otimes P_{\mathcal{M}^k}$ as the distribution of k memory clusters $\{\mathcal{M}^1, \cdots, \mathcal{M}^k\}$.

Theorem 2. Let us assume that the VAE model V_{θ_i} was trained at t_i using the proposed DCMS framework that has already managed k memory clusters. We derive a new Evidence Lower Bound (ELBO) for the VAE model V_{θ_i} at t_i as :

$$\begin{aligned}\mathbb{E}_{P_i}\big[\log p_{\theta^i}(\mathbf{x})\big] \geq \mathbb{E}_{P_{\mathcal{M}^1}\otimes P_{\mathcal{M}^k}}\big[\mathcal{L}_{ELBO}(\mathbf{x};\theta_i,\omega_i)\big] - D_f\big(P_i||P_{\mathcal{M}^1}\otimes P_{\mathcal{M}^k}\big) \\ - \mathcal{F}_{\mathrm{L}}\big(P_{\mathcal{M}^1}\otimes P_{\mathcal{M}^k}, P_i, P_{\theta_i}\big) + \mathcal{F}'_{\mathrm{L}}\big(P_i, P_{\mathcal{M}_i}\big)\,.\end{aligned} \tag{33}$$

Remark. We have the following observations from Theorem 2 :

- The proposed memory system can flexibly control the memory capacity size through the proposed DMO and MPP.

- Ensuring that the memory system manages an appropriate distribution of memory clusters, facilitating the retention of more heterogeneous information. This is accomplished through the implementation of the Dynamic Memory Organizer (DMO), which adaptively adds new memory clusters when incoming data samples are distinct from existing memorized representations.

Unlike Theorem 1, the purpose of Theorem 2 is not to establish a tighter analytical bound than Eq. (25), but to provide a probabilistic interpretation of why dynamically organizing memory into multiple clusters can better approximate the underlying data distribution. The KDM-based expansion mechanism increases the representational coverage of the memory system, while the graph-based pruning mechanism removes redundant clusters, jointly maintaining a compact yet diverse empirical memory distribution. These algorithmic properties are empirically validated in the experiments from Section 5.

Although memory-augmented techniques can represent well complex data distributions, they still may encounter two main constraints : 1) The memory architecture typically possesses a predetermined storage capacity, which limits the system's ability to accommodate new information during the training on continuous data streams; 2) The static model architecture constrains its capacity and effectiveness in capturing a diversity of data distributions; Considering these constraints, we employ the proposed memory augmentation strategy to the DCMS model in the context of continual learning, enabling the model's incremental scalability when assimilating novel information.

4.3. Analysis of Dynamic Expansion Models

Unlike the Theorems 1 and 2, which provide the theoretical backing of the standard DCMS architecture consisting of a single generative model and an external memory system made up of several clusters, this section considers the dynamic expansion extension DCMS-EPM introduced in Sections 3.6 and 3.7. Therefore, the following analysis applies only to this architecture expansion approach for continual learning and not to the standard memory management approach DCMS, evaluated in the main experiments.

Definition 4. Let us define $\mathbf{V}_i$ as a dynamic expansion model updated at t_i, which has already created K experts, where each expert $\mathcal{E}_j$ is implemented by a VAE model V_{θ_j}. Let $P_{\mathcal{M}^1} \otimes P_{\mathcal{M}^{k_j}}$ be a memory distribution that was used to train the expert $\mathcal{E}_j$, where k_j denotes the number of memory clusters used for training the expert $\mathcal{E}_j$.

For completeness, we further analyze the optional dynamic expansion version of DCMS, namely DCMS-EPM, where multiple expert modules are incrementally created. The following theorem is applicable only to this extension.

Theorem 3. Let us assume that the dynamic expansion model $\mathbf{V}_j$ has created K experts at t_j. Let $\{\mathbf{X}_1, \cdots, \mathbf{X}_i\}$ be a set of all data batches previously learned until t_i. Let $P_{i,c}$ denote the distribution of the data batch $\mathbf{X}_c$. We derive a new ELBO to evaluate the performance of the dynamic expansion model $\mathbf{V}_i$ at t_i, as :

$$\begin{aligned}\sum_{c=1}^{i}\left\{\mathbb{E}_{P_{i,c}}\left[\log p_{\theta^i}(\mathbf{x})\right]\right\} \geq \sum_{c=1}^{i}\{&\mathbb{E}_{F_e(P_{i,c})}\left[\mathcal{L}_{ELBO}(\mathbf{x};\theta_{F'_e(P_{i,c})},\omega_{F'_e(P_{i,c})})\right] \\ &- D_f\big(P_{i,c}||F_e(P_{i,c})\big) - \mathcal{F}_{\mathrm{L}}\big(F_e(P_{i,c}), P_{i,c}, P_{\theta_{F_e(P_{i,c})}}\big) \\ &+ \mathcal{F}'_{\mathrm{L}}\big(P_{i,c}, P_{\mathcal{M}_i}\big)\}\,,\end{aligned} \tag{34}$$

where $F'_e(P_{i,c})$ is a function that returns the index of a selected expert, defined by :

$$F'_e(P_{i,c}) = \underset{j=1,\cdots,K}{\operatorname{argmax}}\Big\{\mathbb{E}_{\mathbf{x}\sim P_{i,c}}[\mathcal{L}_{ELBO}(\mathbf{x};\theta_j,\omega_j)]\Big\}\,, \tag{35}$$

where $j^\star = F'_e(P_{i,c})$ is the index of the selected expert that achieves the maximum ELBO on $P_{i,c}$. $F_e(P_{i,c})$ is an expert selection function that returns the memory distribution of the selected expert according to the given distribution $P_{i,c}$, which is defined by :

$$F_e(P_{i,c}) = P_{\mathcal{M}^1} \otimes P_{\mathcal{M}^{k_{j^\star}}}, j^\star = F_e(P_{i,c})\,. \tag{36}$$

Remark. We have several observations from Theorem 3 :

- Utilizing the expert selection mechanism outlined in Eq. (35), the proposed dynamic expansion architecture can optimally assign the most suitable expert to each data batch, thereby ensuring a better model performance.
- Ensuring a suitable discrepancy among experts can facilitate the proposed framework's incremental modeling of diverse underlying data distributions through a compact neural network architecture. This is accomplished via the dynamic expansion mechanism outlined in Section 3.6.

It is worth emphasizing that the theoretical analysis does not assume a particular generative architecture. The ELBO formulation is adopted as a standard probabilistic analysis tool because of its mathematical tractability. Although we consider the DDPM generative model in the experiments, other diffusion probabilistic models can also be used within the variational inference framework, where

Table 1: The hyperparameter configuration for our approach and baselines under the image generation tasks.

Hyperparameter	Values
Learning rate	0.001
Optimization Algorithm	Adam
Maximum memory size	2000
Batch size	64
The number of training epochs	10

the denoising objective corresponds to optimizing a variational lower bound on the data likelihood.

5. Experiments

5.1. Experiment Setting

Baselines and hyperparameters. According to the setting of the unsupervised learning framework from [11], we consider several baselines for comparing the performance of the proposed approach, with CGKD-WAE and CGKD-GAN [11], where'WAE' denotes that we implement each component of the CGKD framework using a Wasserstein auto-encoder [57], Lifelong Teacher-Student (LTS) [34], Continual Neural Dirichlet Process Mixture (CNDPM) [28], Incremental Classifier and Representation Learning (iCaRL) [58], Online Recursive Variational Autoencoder (ORVAE) [29], Reservoir sampling [16], Lifelong Generative Modelling (LGM) [30] and Memory Replay GANs (MeRGANs) [35]. Specifically, the Reservoir sampling method [16] is applied to optimize the VAE and DDPM, respectively, resulting in Reservoir-VAE (R-VAE) and Reservoir-DDPM (R-DDPM). The batch size is set to $b = 64$ while we set the maximum memory capacity (the number of samples) as 2,000, used for all approaches. The maximum number of samples for each memory cluster is $\pi = 128$ and we set the number of diffusion steps $t^\star = 100$ in Eq. (10).

In addition, DCMS-SE denotes the standard DCMS framework using the squared-error-based KDM. DCMS-JS denotes the standard DCMS framework using the JS-divergence-based KDM. Dynamic-JS refers to the dynamic expansion version that uses the DCMS-JS and the proposed model expansion mechanism. DCMS-EPM denotes the dynamic expert expansion variant (Dynamic-JS) equipped with the Expert Pruning Mechanism. We clarify that these are not all separate methods with unrelated designs, but different configurations of the proposed framework. We employ the same hyperparameter configuration for our approach and

Table 2: Image generation performance evaluated using the Fréchet Inception Distance (FID) score for class-incremental learning. Lower FID indicates better performance.

Datasets	DCMS-SE	DCMS-JS	LTS	LGM	R-VAE	R-DDPM	CGKD-GAN	CNDPM	CGKD-WAE	MeRGANs
Split CIFAR10	82.74	76.58	124.22	177.15	155.72	106.18	115.38	233.02	162.12	121.74
Split Fashion	46.65	43.38	128.84	109.20	103.25	82.23	85.23	172.23	88.16	127.55
Split MNIST	28.57	30.63	71.67	66.31	55.67	63.26	54.34	65.19	47.98	49.96
Split SVHN	61.52	62.61	87.25	72.60	65.18	87.22	101.26	150.18	100.15	81.35
Average	54.87	**53.30**	102.99	106.31	94.95	84.72	89.05	155.15	99.54	95.15

all baselines. The primary hyperparameter configurations of our approach and all baselines are provided in Table 1.

Network architectures. In all experiments, we utilize the widely adopted Denoising Diffusion Probabilistic Model (DDPM) [12] network architecture, specifically the U-net structure, to implement the DDPM framework as outlined in [45]. For alternative deep generative models, we employ distinct network architectures tailored to each framework. In the VAE framework, the inference network is constructed as a five-layer convolution neural network with 32, 64, 128, 256, and 512 feature maps, respectively. To parameterize the Gaussian distribution, we introduce two parallel fully connected layers atop the CNN backbone, each with 128 hidden units. The VAE decoding network is designed as a six-layer convolution network with 512, 256, 128, 64, 32, and 3 feature maps. For GAN-based methods, the generator is implemented using a ResNet architecture [2] with four residual blocks, each containing three convolutional layers with 256 feature maps per layer. The discriminator is similarly structured with a ResNet backbone consisting of four residual blocks. To enable the discriminator to output a scalar discrepancy score, we append an additional fully connected layer to the top of the ResNet architecture.

Datasets. In this work, we evaluate OTFCL [11] on four widely used benchmark datasets, namely SVHN [59], FashionMNIST [60], MNIST [61], and CIFAR10 [62]. Since each dataset contains ten classes, we divide them into five disjoint subsets according to the protocol in [10], yielding Split SVHN, Split FashionMNIST, Split MNIST, and Split CIFAR10. To maintain consistency across datasets, all images are resized to $32 \times 32 \times 3$. In addition to these relatively simple benchmarks, we further assess the proposed method on several more challenging large-scale datasets, including CACD [63], CelebA [64], 3DChairs [65], MINIImageNet [66], and ImageNet [4], to comprehensively examine its generality of learning various



Figure 3: Image generation results for the proposed DCMS-SE trained on CelebA-3DChair data stream.

data distributions.

Performance criterion. Following the setting from [11], we employ the Fréchet Inception Distance (FID) score to evaluate the generation performance while for the classification task, we use the average accuracy [10]. A low FID score indicates better image generation results.

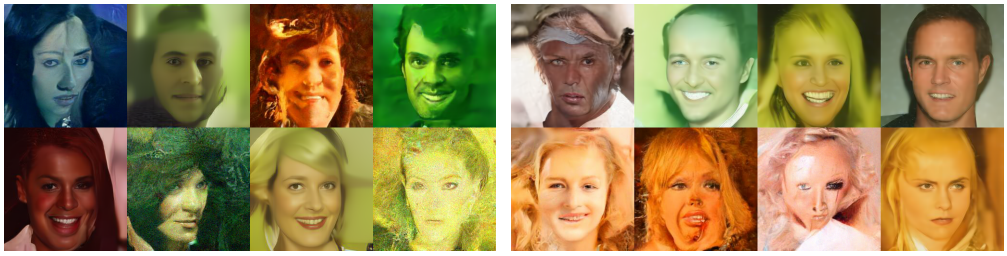
5.2. Class-Incremental Generation

In this section, we assess the efficacy of the proposed methodology in image synthesis tasks. Four benchmark datasets are utilized: Split CIFAR10, Split SVHN, Split MNIST, and Split Fashion. The quantitative results of image generation are summarized in Table 2. The findings reveal that dynamic expansion-based approaches, such as CGKD-VAE and CGKD-GAN, consistently outperform static architectures. Notably, despite leveraging a more advanced generative model, R-DDPM fails to achieve competitive FID scores. This shortcoming is attributed to the limited diversity of samples retained in the memory buffer, which constrains the DDPM's generative capacity to a narrow subset of categories. Conversely, the proposed methods, DCMS-JS and DCMS-SE, yield superior FID scores relative to baseline models, underscoring the effectiveness of the memory optimization framework in preserving sample diversity and fully harnessing the generative potential of DDPM under OTFCL settings. Additionally, Table 6 details the number of stored samples required by each method across the four datasets, demonstrating that the proposed framework matches or reduces memory requirements compared to existing baselines. The storage demands for various methods on more complex datasets are further delineated in Table 7.



(a) DCMS-SE. (b) DCMS-JS.

Figure 4: Image generation results of various models on CACD with $256 \times 256 \times 3$.



(a) DCMS-SE. (b) DCMS-JS.

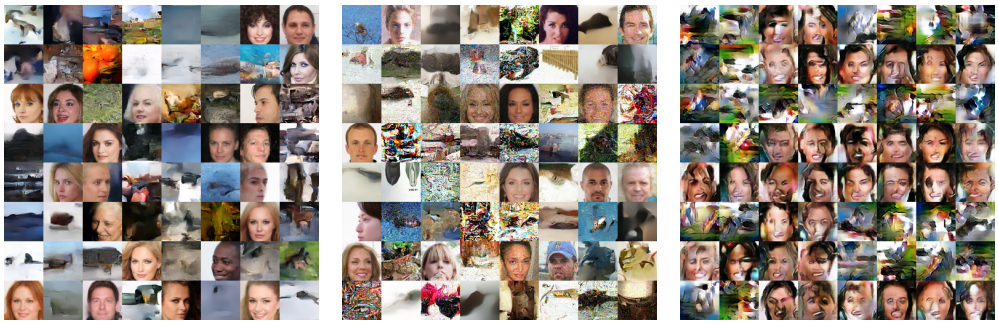
Figure 5: Image generation results of various models on CelebA-HQ with $256 \times 256 \times 3$.

5.3. Learning Multiple and Complex Image Domains

In the previous experiments, we evaluated the effectiveness of various deep learning architectures on standard image datasets. However, real-world images are typically high-resolution, and consists of more complex visual data. This section aims to analyze model performance on such intricate visual domains by training the models on challenging datasets. Specifically, we construct data streams by aggregating the data from the CelebA and CACD datasets, forming the CelebA-CACD dataset. Similarly, we generate a data stream named Celeba-3DChair and CelebA-ImageNet by combining samples from the respective datasets. All images in CelebA-CACD and CelebA-ImageNet are resized to $64 \times 64 \times 3$ pixels. Model performance is quantitatively evaluated using the Fréchet Inception Distance (FID) score computed on 5,000 test samples uniformly collected from the respective test sets, with results presented in Table 3. The results indicate that the proposed method achieves superior performance on data streams comprising complex, high-resolution images. In Table 4 we provide the FID scores calculated for high-resolution datasets such as CACD, FFHQ, and CelebA-HQ [64]. We also provide image generation results after training on Celeba-ImageNet and

Table 3: Results of the FID evaluation on datasets with complex images. Lower FID indicates better performance.

Datasets	DCMS-SE	DCMS-JS	LTS	LGM	R-VAE	R-DDPM	CGKD-GAN	CNDPM	CGKD-WAE	MeRGANs
CelebA-3DChair	**40.45**	82.18	186.25	241.14	210.18	183.72	132.12	340.25	154.45	166.99
CelebA-CACD	67.30	**48.38**	124.87	117.76	121.52	103.52	78.00	336.34	142.52	101.97
CelebA-ImageNet	**110.60**	115.25	255.94	265.23	225.12	217.98	178.72	186.76	170.07	236.81
Split MINIImageNet	**146.98**	154.83	179.78	216.06	205.12	181.15	176.18	302.58	241.11	169.26



(a) DCMS-SE. (b) DCMS-JS. (c) CGKD-GAN.

Figure 6: The generation results of various models on CelebA to ImageNet.

CelebA-3DChair in Figures 6 and 3. Additionally, representative high-resolution images generated by our method are displayed in Figures 4, 5, and 7. These empirical findings demonstrate that our approach consistently delivers state-of-the-art results, including on challenging, high-resolution image datasets.

5.4. Few-Shot Generation

This section presents the evaluation of the proposed methodology for few-shot image generation, conducted on the MiniImagenet benchmark [66]. This dataset represents a standard evaluation protocol for few-shot learning tasks [67] and comprises 100 distinct classes. Adhering to established conventions, the dataset is partitioned into subsets of 64, 16, and 20 classes designated for meta-training, meta-validation, and meta-testing phases, respectively.

In the present configuration, a data stream denoted as Split MINIImageNet is generated through the amalgamation of the meta-training and meta-validation sets. This stream is subsequently divided into 16 distinct subsets, with each comprising samples sourced from five consecutive classes. Several models undergo sequential training on this stream, and their performance is benchmarked against the meta-

Table 4: FID scores for assessing the image generation performance for datasets containing high resolution images. Lower FID indicates better performance.

Methods	Resolution	CelebA-HQ	CACD	FFHQ
DCMS-SE	$128 \times 128 \times 3$	**89.23**	69.11	95.02
DCMS-JS	$128 \times 128 \times 3$	96.03	**57.19**	**90.80**
CGKD-GAN	$128 \times 128 \times 3$	132.65	142.66	157.03
CGKD-WVAE	$128 \times 128 \times 3$	139.96	158.32	179.59
DCMS-SE	$256 \times 256 \times 3$	87.39	**110.21**	**123.95**
DCMS-JS	$256 \times 256 \times 3$	**75.18**	123.96	129.38
CGKD-GAN	$256 \times 256 \times 3$	168.52	236.98	254.32
CGKD-WVAE	$256 \times 256 \times 3$	176.63	240.12	261.37



(a) DCMS-SE. (b) DCMS-JS.

Figure 7: Image results generated by the DCMS-SE and DCMS-JS on FFHQ with $256 \times 256 \times 3$.

testing set. As documented in Table 3, the results demonstrate that the proposed approach consistently outperforms competing methodologies in few-shot generative tasks.

5.5. Classification Tasks

This section presents the evaluation of the proposed DCMS on classification tasks under the task-free continual learning (TFCL) regime. Both the DCMS-SE and DCMS-JS variants are employed for classifier optimization via supervised learning. Adhering to the TFCL protocol established by Zhu et al. (2021), memory capacities are configured at 1,000, 2,000, and 5,000 exemplars for Split CIFAR-10, Split MNIST, and Split CIFAR-100, respectively.

During training, a limited batch size of ten samples per iteration is adopted, and previously observed data are not revisited. For Split MNIST, a fully connected network with two hidden layers of 400 units each is used, while a ResNet-18 architecture is applied for Split CIFAR10 and Split CIFAR100. The results are reported in Table 5, where the proposed DCMS consistently achieves superior

Table 5: Classification results following five independent runs for various models on three datasets. * and † denote the results cited from [25] and [26], respectively.

Methods	**Split MNIST**	**Split CIFAR10**	**Split CIFAR100**
finetune*	19.75 ± 0.05	18.55 ± 0.34	3.53 ± 0.04
GEM* [17]	93.25 ± 0.36	24.13 ± 2.46	11.12 ± 2.48
iCARL* [58]	83.95 ± 0.21	37.32 ± 2.66	10.80 ± 0.37
reservoir*	92.16 ± 0.75	42.48 ± 3.04	19.57 ± 1.79
MIR* [23]	93.20 ± 0.36	42.80 ± 2.22	20.00 ± 0.57
GSS* [24]	92.47 ± 0.92	38.45 ± 1.41	13.10 ± 0.94
CoPE-CE* [25]	91.77 ± 0.87	39.73 ± 2.26	18.33 ± 1.52
CoPE* [25]	93.94 ± 0.20	48.92 ± 1.32	21.62 ± 0.69
ER + GMED† [26]	82.67 ± 1.90	34.84 ± 2.20	20.93 ± 1.60
ER_a + GMED† [26]	82.21 ± 2.90	47.47 ± 3.20	19.60 ± 1.50
WGF-SVGD [48]	-	47.90 ± 2.50	19.90 ± 2.30
CURL* [27]	92.59 ± 0.66	-	-
CNDPM* [28]	93.23 ± 0.09	45.21 ± 0.18	20.10 ± 0.12
Dynamic-OCM [32]	94.02 ± 0.23	49.16 ± 1.52	21.79 ± 0.68
ORVAE [29]	94.07 ± 0.13	50.43 ± 0.15	22.83 ± 0.25
DCMS-SE	95.12 ± 0.32	50.02 ± 1.12	22.03 ± 0.79
DCMS-JS	94.76 ± 0.29	49.89 ± 1.23	22.98 ± 0.74
Dynamic-JS	**98.52** ± 0.32	**55.94** ± 0.58	**26.27** ± 0.63

classification performance compared to existing baseline methods. The baseline methods include the Continual Prototype Evolution with prototypical approach (CoPE) [25] and balanced replay with standard cross-entropy (CoPE-CE) [25]. Also the Gradient Memory EDiting (GMED) with Experience Replay (ER) [68] as ER+GMED [26], and after averaging using data augmentation such as random rotations, scaling, and horizontal flipping applied to memory examples drawn for replay in ER, denoted as ER_a + GMED [26]. dynamic-OCM (Online Cooperative Memorization) was proposed in [32]. Wasserstein Gradient Flow (WGF) with Stein Variational Gradient Descent (SVGD), denoted as WGF-SVGD, was proposed in [48].

Furthermore, we present the classification outcomes obtained by the proposed Dynamic-JS framework, which integrates the JS-divergence-based Knowledge Distillation Module (KDM) with a dynamic expansion mechanism. as explained in Section 3.6. Empirical evidence shows that incorporating the dynamic expansion mechanism yields additional improvements in model performance across classification tasks.

Table 6: The number of stored samples of various models used for class-incremental learning.

Datasets	DCMS-SE	DCMS-JS	LTS	LGM	R-VAE	R-DDPM	CGKD-GAN	CNDPM	CGKD-WAE	CGKD-VAE
Split CIFAR10	1792	1792	2000	2000	2000	2000	2000	2000	2000	2000
Split SVHN	1536	1664	2000	2000	2000	2000	2000	2000	2000	2000
Split Fashion	1664	1664	2000	2000	2000	2000	2000	2000	2000	2000
Split MNIST	1408	1536	2000	2000	2000	2000	2000	2000	2000	2000

Table 7: The number of stored samples of various models used for the domain-incremental learning.

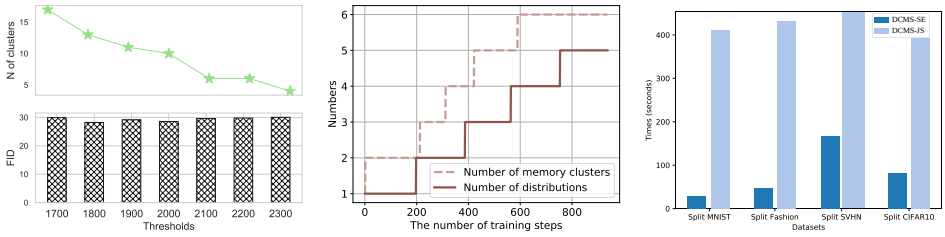
Datasets	DCMS-SE	DCMS-JS	LTS	LGM	R-VAE	R-DDPM	CGKD-GAN	CNDPM	CGKD-WAE	CGKD-VAE
CelebA-CACD	1920	1920	2000	2000	2000	2000	2000	2000	2000	2000
CelebA-3DChair	1920	1920	2000	2000	2000	2000	2000	2000	2000	2000
Split MINIImageNet	1920	1920	2000	2000	2000	2000	2000	2000	2000	2000
CelebA-ImageNet	1920	1920	2000	2000	2000	2000	2000	2000	2000	2000

5.6. Ablation Study

In this section, we perform a series of experiments to evaluate the performance of the proposed approach under different model configurations.

The effect when changing λ. The threshold parameter λ, specified in Eq. (12), modulates the rate of memory expansion within the system. We systematically examine the impact and efficacy of the proposed methodology by varying the threshold λ in the TFCL framework. In particular, we optimize the DCMS-SE model on the Split MNIST dataset under different λ settings, with the empirical findings illustrated in Fig. 8-a. The results indicate that a lower expansion threshold λ makes the memory expansion criterion easier to satisfy, thereby facilitating the frequent generation and integration of additional memory clusters. In contrast, a higher threshold λ imposes a stricter novelty requirement and results in fewer memory clusters. Notably, the DCMS framework achieves optimal performance when configured with six memory clusters, as empirically observed in the experiments.

The dynamic expansion process. In order to examine the capability of the proposed dynamic memory mechanism in tracking temporal data distribution shifts, we systematically record both the number of memory clusters and the corresponding data distributions at each training interval. We optimize the DCMS-SE framework on the Split MNIST benchmark, with the outcomes illustrated in Fig. 8-b. The experimental findings indicate that DCMS-SE effectively generates new



(a) Changing λ for DCMS-SE.

(b) The dynamic expansion of DCMS-SE.

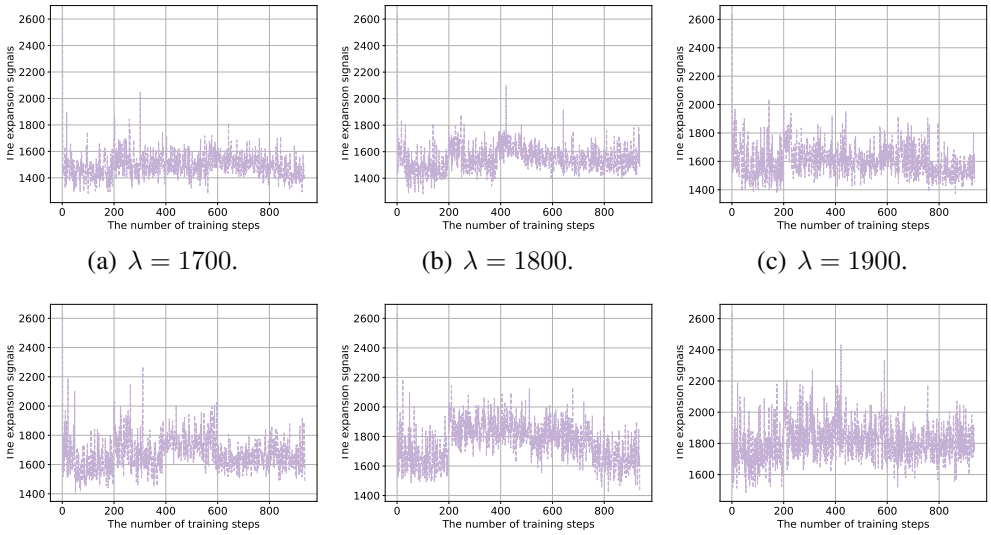
(c) The time required for the memory optimization.

Figure 8: Ablation study results. (a) The memory cluster count and the results about the model expansion of DCMS-SE when we change λ in Eq. (12). (b) The memory cluster count and data distribution changes at each time. (c) The computational time needed for the memory operations.

memory clusters in response to distributional changes, thereby validating that the dynamic memory expansion strategy provides reliable cues for the adaptive memory allocation.

Memory management processing time. In many real-world scenarios, minimizing computational overhead is essential for enabling real-time operations. We analyze the computational time required for the optimization phase of the proposed DCMS framework under the TFCL setting. Fig. 8-c illustrates the optimization times for DCMS-SE and DCMS-JS when training on the Split MNIST dataset, where only the memory framework optimization is performed, excluding the DDPM training phase. Empirical findings indicate that the DCMS-SE method typically completes the optimization in under one minute across most datasets. Conversely, DCMS-JS incurs higher optimization times than DCMS-SE, attributable to the increased computational demands of the JS-divergence calculation.

Dynamic memory expansion. The proposed dynamic memory expansion framework is designed to incrementally generate activation signals that facilitate the memory expansion. We examine how these activation signals, produced by the memory expansion mechanism, can inform the development of a more efficient memory architecture. In particular, we optimize both DCMS-JS and DCMS-SE models varying λ on the Split MNIST benchmark. The activation signals computed at each iteration (as indicated on the left-hand side of Eq. (10)) are shown for DCMS-JS and DCMS-SE in Fig. 9 and Fig. 10, respectively. Empirical analysis reveals the emergence of distinct signal peaks at multiple training intervals corresponding to shifts in the data distribution. Consequently, the use of square loss and JS divergence within the proposed dynamic memory expansion mechanism provides robust activation signals, thereby supporting the construction of an



(a) $\lambda = 1700$. (b) $\lambda = 1800$. (c) $\lambda = 1900$.

(d) $\lambda = 2000$. (e) $\lambda = 2100$. (f) $\lambda = 2200$.

Figure 9: The memory expansion signals when changing λ from Eq. (12) in DCMS-SE.

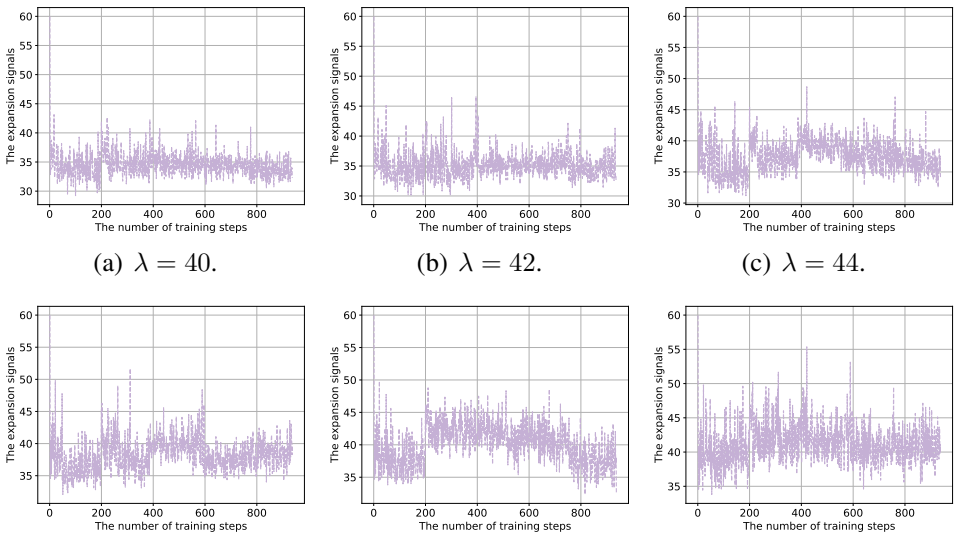
Table 8: The performance metric evaluated by the FID score under class-incremental learning. Lower FID indicates better performance.

Datasets	DCMS-SE	DCMS-JS	DCMS-EPM
Split CIFAR10	82.74	76.58	**75.13**
Split Fashion	46.65	43.38	**41.52**
Split MNIST	28.57	30.63	**30.22**
Split SVHN	61.52	62.61	**60.52**
Average	54.87	53.30	**51.84**

adaptive and efficient memory architecture.

The maximum size π of the memory cluster. Since each memory cluster has a fixed capacity (the number of memorized samples) of π, we investigate the performance changes of the proposed memory approach when varying the size of the memory cluster. Specifically, we choose the hyperparameters $\lambda = 2000$ and $\lambda = 42$ in Eq. (12) for DCMS-SE and DCMS-JS, respectively. The empirical results are provided in Fig. 11, showing that considering memory clusters of different sizes does not significantly change model performance.

The effects of the proposed Expert Removing Mechanism (ERM). We evaluate the efficacy of the proposed ERM from Section 3.7 by training the DCMS-ERM



(a) $\lambda = 40$. (b) $\lambda = 42$. (c) $\lambda = 44$.

(d) $\lambda = 46$. (e) $\lambda = 48$. (f) $\lambda = 50$.

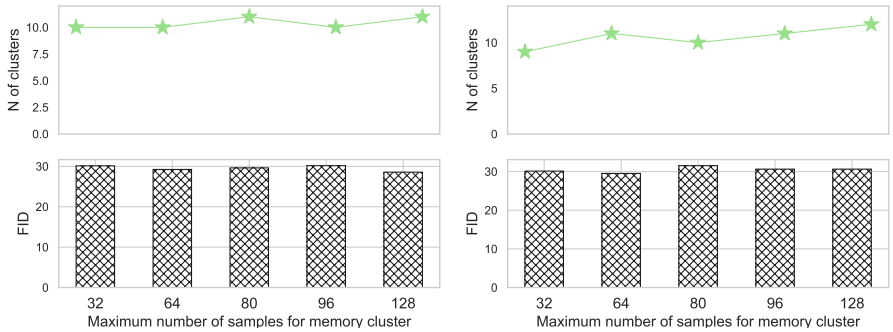
Figure 10: The memory expansion signals when changing λ in DCMS-JS.

across four benchmark datasets, including Split CIFAR-10, Split Fashion-MNIST, Split MNIST, and Split SVHN. Specifically, we set the threshold parameter $M = 6$ uniformly across all datasets, with the experimental outcomes summarized in Table 8. These results show that DCMS-ERM effectively eliminates redundant experts while enhancing the overall classification performance compared to the DCMS-SE and DCMS-JS baselines, demonstrating the effectiveness of the proposed ERM.

The complexity analysis. The prototype of each memory cluster is computed only when a new cluster is created or updated, rather than at every training iteration. Assuming there are K memory clusters and each cluster stores at most n samples, the prototype computation requires $O(n^2)$ operations for a single updated cluster. Since n is fixed and relatively small, this cost remains bounded throughout continual learning.

- **Memory expansion and sample assignment.** For each incoming sample, DCMS only computes its distance to the K cluster prototypes, resulting in a complexity of $O(K)$. This avoids exhaustive comparisons with all stored memory samples, making the expansion criterion computationally efficient.

- **Relation matrix construction for memory pruning.** The relation matrix has a complexity of $O(K^2)$, where K is the number of memory clusters. How-



(a) DCMS-SE. (b) DCMS-JS.

Figure 11: The number of memory clusters and the FID score when using different memory capacity configuration for each memory cluster.

ever, this operation is performed only when the memory reaches its predefined capacity and the pruning process is triggered, rather than for every training iteration. Moreover, the number of clusters is explicitly bounded by the maximum memory budget, ensuring that this overhead remains limited.

In terms of memory complexity, DCMS stores only the memory samples together with one prototype for each cluster, resulting in a storage complexity of $O(KM)$, which is comparable to conventional replay-based memory methods with the same memory budget.

The contribution of each module. To analyze the contribution of each proposed module, we consider several baselines with different configurations. DCMS-EPM denotes a variant of our framework that integrates the Dynamic-JS and EPM modules, as described in Sections 3.6 and 3.7. DCMS-Single refers to a baseline that relies on a single memory buffer. In addition, DCMS-Single randomly selects data samples for the memory buffer. DCMS-NoPruning denotes a variant that removes the memory pruning mechanism while retaining the SE-based KDM. DCMS-Prototype employs the SE-based KDM and incrementally updates prototypes over time. We report the results of different configurations in Table 9. From the empirical results, we have the following observations: (1) incorporating the EPM into the DCMS framework consistently improves performance; (2) using a single memory buffer is insufficient to preserve high-quality historical examples, leading to degraded performance; (3) removing the memory pruning mechanism results in noticeable performance drops, demonstrating its effectiveness; (4) incrementally updating prototypes does not yield significant performance gains, while introducing additional computational overhead.

Table 9: The performance metric evaluated by the FID score when using different configurations. Lower FID indicates better performance.

Datasets	DCMS-SE	DCMS-JS	DCMS-EPM	DCMS-Single	DCMS-NoPruning	DCMS-Prototype
Split CIFAR10	82.74	76.58	**75.13**	104.52	87.39	82.61

6. Conclusion and Limitation

In this research study we present a novel approach, using a dynamic cluster information management system, termed the Dynamic Cluster Memory System (DCMS) to address catastrophic forgetting in Task-Free Continual Learning (TFCL). This architecture incrementally orchestrates and refines a collection of memory clusters, each tailored to encapsulate and retain unique data distributions. We introduce a dynamic memory expansion protocol that designates the centroid sample as the representative prototype for each cluster and quantifies the Euclidean distance between new data points and these prototypes. We also devise an advanced sample selection algorithm that autonomously identifies and assigns pivotal historical exemplars to their corresponding memory clusters, thereby maintaining semantic integrity within each cluster. For deployment on resource-limited platforms, we enforce memory capacity constraints that limit the system's ability to accommodate new data during ongoing learning. Then we propose a memory pruning strategy that systematically removes redundant clusters through graph-based relational analysis. Comprehensive experiments conducted across both supervised and unsupervised learning settings validate that the proposed framework consistently outperforms existing methods in key performance metrics.

While the proposed memory pruning strategy allows for the continual replacement of existing memory clusters with new ones to support the integration of novel information, it inevitably leads to the forfeiture of some historical data when handling an unbounded data stream, owing to the limitations imposed by a fixed memory capacity. The dynamic expansion mechanism presented in this study partially addresses this issue by adaptively scaling the model's capacity; nonetheless, this solution remains suboptimal for deployment on devices with stringent resource constraints. Future investigations will prioritize the development of a sophisticated knowledge distillation framework capable of autonomously identifying and consolidating analogous components into a unified representation, thereby further minimizing model complexity without compromising performance fidelity.

References

[1] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, D. Krishnan, Unsupervised pixel-level domain adaptation with generative adversarial networks, in: Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 3722–3731.

[2] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.

[3] A. Sharif Razavian, H. Azizpour, J. Sullivan, S. Carlsson, CNN features off-the-shelf: an astounding baseline for recognition, in: Proc. of the IEEE CVPR-workshops, 2014, pp. 806–813.

[4] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Inf. Proc. Systems (NIPS), 2012, pp. 1097–1105.

[5] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, Image-to-image translation with conditional adversarial networks, in: Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1125–1134.

[6] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, S. Wermter, Continual lifelong learning with neural networks: A review, Neural Networks 113 (2019) 54–71.

[7] J. Bang, H. Kim, Y. Yoo, J.-W. Ha, J. Choi, Rainbow memory: Continual learning with a memory of diverse samples, in: Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 8218–8227.

[8] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, R. Hadsell, Overcoming catastrophic forgetting in neural networks, Proc. of the National Academy of Sciences (PNAS) 114 (13) (2017) 3521–3526.

[9] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, R. Hadsell, Progressive neural networks, arXiv preprint arXiv:1606.04671 (2016).

[10] R. Aljundi, K. Kelchtermans, T. Tuytelaars, Task-free continual learning, in: Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 11254–11263.

[11] F. Ye, A. G. Bors, Continual variational autoencoder via continual generative knowledge distillation, in: Proc. AAAI Conference on Artificial Intelligence, 2023, pp. 10918–10926.

[12] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, Advances in Neural Information Processing Systems (NeurIPS) 33 (2020) 6840–6851.

[13] H. Jiang, X. Feng, H. Yu, PSCD: Synergistic optimization of parameter sharing and chain distillation for plasticity-stability balance in continual learning, Knowledge-Based Systems 330, Part C (2025) 114631.

[14] Z. Fang, B. Wan, S. Guo, J. K. Liu, HMEN: A hybrid modular network with dynamic expansion for continual learning, Knowledge-Based Systems 335 (2026) 115182.

[15] X. Nie, S. Xu, X. Liu, G. Meng, C. Huo, S. Xiang, Bilateral memory consolidation for continual learning, in: Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 16026–16035.

[16] J. S. Vitter, Random sampling with a reservoir, ACM Transactions on Mathematical Software (TOMS) 11 (1) (1985) 37–57.

[17] D. Lopez-Paz, M. Ranzato, Gradient episodic memory for continual learning, in: Advances in Neural Information Processing Systems (NIPS), 2017, pp. 6467–6476.

[18] Y. Bengio, A. Courville, P. Vincent, Representation learning: A review and new perspectives, IEEE Trans. on Pattern Analysis and Machine Intelligence 35 (8) (2013) 1798–1828.

[19] M. Zhai, L. Chen, F. Tung, J. He, M. Nawhal, G. Mori, Lifelong GAN: Continual learning for conditional image generation, in: Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 2759–2768.

[20] D. P. Kingma, M. Welling, Auto-encoding variational Bayes, arXiv preprint arXiv:1312.6114 (2013).

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Proc. Advances in Neural Inf. Proc. Systems (NIPS), 2014, pp. 2672–2680.

[22] F. Ye, A. G. Bors, Online task-free continual learning via discrepancy mechanism, Knowledge-Based Systems 322 (2025) 113688.

[23] R. Aljundi, E. Belilovsky, T. Tuytelaars, L. Charlin, M. Caccia, M. Lin, L. Page-Caccia, Online continual learning with maximal interfered retrieval, in: Advances in Neural Information Processing Systems (NeurIPS), 2019, pp. 11872–11883.

[24] R. Aljundi, M. Lin, B. Goujaud, Y. Bengio, Gradient based sample selection for online continual learning, in: Advances in Neural Information Processing Systems (NeurIPS), 2019, pp. 11817–11826.

[25] M. De Lange, T. Tuytelaars, Continual prototype evolution: Learning online from non-stationary data streams, in: Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 8250–8259.

[26] X. Jin, A. Sadhu, J. Du, X. Ren, Gradient-based editing of memory examples for online task-free continual learning, in: Advances in Neural Information Processing Systems (NeurIPS), Vol. 35, 2021, pp. 29193–29205.

[27] D. Rao, F. Visin, A. A. Rusu, Y. W. Teh, R. Pascanu, R. Hadsell, Continual unsupervised representation learning, in: Proc. Neural Information Processing Systems (NeurIPS), 2019, pp. 7645–7655.

[28] S. Lee, J. Ha, D. Zhang, G. Kim, A neural Dirichlet process mixture model for task-free continual learning, in: Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:2001.00689, 2020.

[29] F. Ye, A. G. Bors, Learning dynamic latent spaces for lifelong generative modelling, in: Proc. of the AAAI Conference on Artificial Intelligence, Vol. 37, 2023, pp. 10891–10899.

[30] J. Ramapuram, M. Gregorova, A. Kalousis, Lifelong generative modeling, in: Proc. Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:1705.09847, 2017.

[31] A. Achille, T. Eccles, L. Matthey, C. Burgess, N. Watters, A. Lerchner, I. Higgins, Life-long disentangled representation learning with cross-domain latent homologies, in: Advances in Neural Information Processing Systems (NeurIPS), 2018, pp. 9873–9883.

[32] F. Ye, A. G. Bors, Continual variational autoencoder learning via online cooperative memorization, in: Proc. European Conference on Computer Vision (ECCV), vol. LNCS 13683, 2022, pp. 531–549.

[33] A. Larsen, S. K. Sønderby, H. Larochelle, O. Winther, Autoencoding beyond pixels using a learned similarity metric, in: Proc. International Conf. on Machine Learning (ICML), vol. PMLR 48, 2015, pp. 1558–1566.

[34] F. Ye, A. G. Bors, Lifelong teacher-student network learning, IEEE Trans. on Pattern Analysis and Machine Intelligence 44 (10) (2022) 6280–6296.

[35] C. Wu, L. Herranz, X. Liu, J. van de Weijer, B. Raducanu, Memory replay GANs: Learning to generate new categories without forgetting, in: Advances In Neural Inf. Proc. Systems (NeurIPS), 2018, pp. 5962–5972.

[36] H. Shin, J. K. Lee, J. Kim, J. Kim, Continual learning with deep generative replay, in: Advances in Neural Inf. Proc. Systems (NIPS), 2017, pp. 2990–2999.

[37] A. Seff, A. Beatson, D. Suo, H. Liu, Continual learning in generative adversarial nets, arXiv preprint arXiv:1705.08395 (2017).

[38] R. Gao, W. Liu, DDGR: continual learning with deep diffusion-based generative replay, in: International Conference on Machine Learning (ICML), PMLR 202, 2023, pp. 10744–10763.

[39] Q. Jodelet, X. Liu, Y. J. Phua, T. Murata, Class-incremental learning using diffusion model for distillation and replay, in: Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV), 2023, pp. 3425–3433.

[40] N. Dilokthanakul, P. A. Mediano, M. Garnelo, M. C. Lee, H. Salimbeni, K. Arulkumaran, M. Shanahan, Deep unsupervised clustering with gaussian mixture variational autoencoders, arXiv preprint arXiv:1611.02648 (2016).

[41] P. Buzzega, M. Boschini, A. Porrello, D. Abati, S. Calderara, Dark experience for general continual learning: a strong, simple baseline, Advances in Neural Information Processing Systems (NeurIPS) 33 (2020) 15920–15930.

[42] Z. Li, Y. Wang, C. Xiong, Y. Xia, X. Hou, Y. Zhao, M. Zhang, K. Xiao, B. Yang, MacVQA: Adaptive memory allocation and global noise filtering for continual visual question answering, in: AAAI Conference on Artificial Intelligence, pp. 31888–31896.

[43] Y. Wang, C. Xiong, Z. Qin, M. Zhang, K. Xiao, Z. Li, HyLoVQA: Dynamic hypernetwork-generated low-rank adaptation for continual visual question answering, arXiv preprint arXiv:2605.22035 (2026).

[44] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, Y. Shao, W. Zhang, B. Cui, M.-H. Yang, Diffusion models: A comprehensive survey of methods and applications, ACM Computing Surveys 56 (4) (2022) 1–39.

[45] A. Q. Nichol, P. Dhariwal, Improved denoising diffusion probabilistic models, in: Proc. of the International Conference on Machine Learning (ICML), vol. PMLR 139, 2021, pp. 8162–8171.

[46] R. Kurle, B. Cseke, A. Klushyn, P. van der Smagt, S. Günnemann, Continual learning with bayesian neural networks for non-stationary data, in: International Conference on Learning Representations (ICLR), https://openreview.net/forum?id=SJlsFpVtDB, 2020.

[47] K. Raghavan, P. Balaprakash, Formalizing the generalization-forgetting trade-off in continual learning, Advances in Neural Information Processing Systems (NeurIPS) 34 (2021) 17284–17297.

[48] Z. Wang, L. Shen, L. Fang, Q. Suo, T. Duan, M. Gao, Improving task-free continual learning by distributionally robust memory evolution, in: Proc. International Conference on Machine Learning (ICML), vol. PMLR 162, 2022, pp. 22985–22998.

[49] J. Astola, P. Haavisto, Y. Neuvo, Vector median filters, Proc. of the IEEE 78 (4) (1990) 678–689.

[50] J. Bang, H. Koh, S. Park, H. Song, J.-W. Ha, J. Choi, Online continual learning on a contaminated data stream with blurry task boundaries, in: Proc. of IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 9275–9284.

[51] Y. Gu, X. Yang, K. Wei, C. Deng, Not just selection, but exploration: Online class-incremental continual learning via dual view consistency, in: Proc.

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 7442–7451.

[52] R. Tiwari, K. Killamsetty, R. Iyer, P. Shenoy, GCR: Gradient coreset based replay buffer selection for continual learning, in: Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 99–108.

[53] Y. Guo, B. Liu, D. Zhao, Online continual learning through mutual information maximization, in: Proc. International Conference on Machine Learning (ICML), vol. PMLR 162, 2022, pp. 8109–8126.

[54] I. O. Tolstikhin, B. K. Sriperumbudur, B. Schölkopf, Minimax estimation of maximum mean discrepancy with radial kernels, Advances in Neural Information Processing Systems (NIPS) 29 (2016) 1930–1938.

[55] F. Ye, A. G. Bors, Lifelong infinite mixture model based on knowledge-driven Dirichlet process, in: Proc. of the IEEE Int. Conf. on Computer Vision (ICCV), 2021, pp. 10695–10704.

[56] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, M. Welling, Improved variational inference with inverse autoregressive flow, in: Advances in Neural Information Processing Systems (NIPS), Vol. 29, 2016, pp. 4743–4751.

[57] I. Tolstikhin, O. Bousquet, S. Gelly, B. Schoelkopf, Wasserstein auto-encoders, in: International Conference on Learning Representations (ICLR), arXiv preprint arXiv:1711.01558, 2018.

[58] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, C. H. Lampert, iCaRL: Incremental classifier and representation learning, in: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2001–2010.

[59] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng, Reading digits in natural images with unsupervised feature learning, in: NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2011.

[60] H. Xiao, K. Rasul, R. Vollgraf, Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms, arXiv preprint arXiv:1708.07747 (2017).

[61] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. of the IEEE 86 (11) (1998) 2278–2324.

[62] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images, Tech. rep., Univ. of Toronto (2009).

[63] B.-C. Chen, C.-S. Chen, W. H. Hsu, Cross-age reference coding for age-invariant face recognition and retrieval, in: Proc. European Conf on Computer Vision (ECCV), vol. LNCS 8694, 2014, pp. 768–783.

[64] Z. Liu, P. Luo, X. Wang, X. Tang, Deep learning face attributes in the wild, in: Proc. of IEEE Int. Conf. on Computer Vision (ICCV), 2015, pp. 3730–3738.

[65] M. Aubry, D. Maturana, A. A. Efros, B. C. Russell, J. Sivic, Seeing 3D chairs: exemplar part-based 2D-3D alignment using a large dataset of CAD models, in: Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 3762–3769.

[66] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, D. Wierstra, Matching networks for one shot learning, Advances in neural information processing systems (NIPS) 29 (2016) 3637–3645.

[67] E. Schönfeld, S. Ebrahimi, S. Sinha, T. Darrell, Z. Akata, Generalized zero- and few-shot learning via aligned variational autoencoders, in: Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 8247–8255.

[68] A. V. Robins, Catastrophic forgetting, rehearsal and pseudorehearsal, Connection Science 7 (1995) 123–146.