Online Task-Free Continual Learning via Discrepancy Mechanism

Fei Ye[a], and Adrian G. Bors[b]

[a]School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, China,
[b]Derpartment of Computer Science, University of York, York YO10 5GH, York, UK,

Abstract

Task Free Continual Learning (TFCL) involves training a deep neural network in a dynamic changing environment defined by unpredictable probabilistic data representation changes. Catastrophic forgetting, which occurs when the network's weights are replaced following training, is the main factor of performance degeneration in the TFCL. We develop a theoretical framework that accounts for the forgetting process in a continual learning model by deriving the generalization bounds when learning new data while preserving the previously learnt data representations. The theoretical analysis indicates that by dynamically creating new trainable submodels when new information becomes available, can address the challenges of catastrophic forgetting. We then propose the Online Discrepancy Distance Learning (ODDL) model, which expands model's architecture by evaluating the difference between what was learned by the components of a mixture model and a memory buffer storing the newly available data for training. We then develop a sample selection approach based on a proposed discrepancy distance, which stores only those samples deemed critical to the learning of the model, ensuring the learning of diverse information. The proposed methodology outperforms other static and dynamic expansion models in various TFCL applications.

Keywords: Lifelong Learning; Expanding Mixture Models; Task-Free Continual Learning; Representation Learning.

1. Introduction

An essential function in machine learning is represented by the ability to learn and acquire knowledge from a changing environment without forgetting what was learnt before. Many deep-learning models can achieve significant performance on individual tasks but fail when learning a succession of different datasets. A challenge to such systems is catastrophic forgetting [42], which is caused by the rewriting of the model's parameters when adapting to new tasks during learning. Continual Learning (CL), also known as lifelong learning, emerged lately as a research direction addressing catastrophic forgetting [42].

Continual learning scenarios can be roughly divided into two branches : general CL [62] and Task-Free Continual Learning (TFCL) [3]. The former assumption considers that task information is accessible during both training and testing phases, while the latter does not access any task information. Many studies have proposed several technologies to address network forgetting in continual learning, which can be roughly divided into three branches : memory-replay-based methods that store some past data in a memory buffer, in order to be used later for retraining the model [4, 12], regularization-based methods that would introduce a penalty item in the primary objective function in order to prevent changes in some important network parameters and the Dynamic Expansion Models (DEMs) [53] that preserve the entire previously learnt knowledge in the parameters of frozen modules or even entire networks, by progressively building new sub-networks when learning new tasks. Among these approaches, memory-based methods represent a simple but efficient approach to deal with network forgetting while their primary challenge is represented by the fact that they are constrained by the available memory. Memory-based methods can also be empowered by regularization-based strategies to further improve the model's performance. In addition, compared to the static network architecture, dynamic expansion mechanisms can ensure the best performance for all tasks, theoretically demonstrated in [53], but at a cost of substantial storage requirements due to the architecture expansion. Furthermore, the DEM expansion mechanisms relies highly on the task information which is not available in the TFCL scenario.

Recently, certain studies have analyzed continual learning from different perspectives, including from the point of view of NP-hard challenges [27], Teacher-Student frameworks [32, 56], evaluation of risk bounds [52, 53] or from the game theory perspective [43]. However, all these studies rely on knowing the task information, thus considering that task boundaries are available. However, such assumptions cannot be considered in realistic learning scenarios, when data distributions continuously change over time, as in considered in the TFCL scenario. Most CL methods can be considered in the context of TFCL after dropping the task labels associated with the data. Memory-driven CL approaches would store some data which are then used during the subsequent training to address forgetting [16, 23]. However, such approaches have to design an appropriate sample selection mechanism that selectively stores only those training samples deemed important, aiming to avoid memory overload [3]. Another challenge for the memory-buffer based CL methods is represented by the negative backward transfer, which leads to the model's performance degeneration [12] due to the interference between the statistics of memorized samples with the model's continuous updating when learning from the incoming data samples [12]. Various DEM's mixture expansion criteria have been considered relying on either estimating the data density [33, 45] or by controlling loss function changes [54]. However, such approaches do not provide theoretical guarantees for their dynamic expansion mechanisms and cannot guarantee optimal network architectures for TFCL problems.

This paper provides a new theoretical framework for TFCL, by extending the theoretical results from the domain adaptation theory [40] and deriving generalization risk bounds for defining the dynamic forgetting process. The theoretical analysis shows that a dynamic expansion model improves its performance over a static, classical model, while its network architecture is optimized by promoting the knowledge diversity among its components during the model's expansion process. Inspired by this analysis we develop a new approach for TFCL, called Online Discrepancy Distance Learning (ODDL). ODDL evaluates the differences between the new information stored in the memory buffer and that learnt previously by the model. This is then considered as an expansion signal for a dynamic expansion model, aiming to ensure a compact DEM architecture. Furthermore, the proposed dynamic expansion mech-

anism encourages each component to capture different underlying data probabilistic representations. We also propose the Online Discrepancy Distance Learning with sample selection (ODDL-S) model which selects new data for the memory buffer, ensuring the data diversity in the memory buffer which is different from the already accumulated knowledge. ODDL-S encourages the novelty of the knowledge acquired by the newly trained DEM components.

This paper provides advancements in four different directions when compared to [57] : (1) This paper introduces a novel expert initialization strategy that automatically finds the appropriate expert, thus accelerating its training; (2) The theoretical analysis framework from [57] does not describe the effects of the transfer learning for TFCL, while this paper provides a new theoretical analysis for transfer learning, which explains the benefits of choosing an appropriate expert; (3) It proposes a novel discrepancy distance measure approach that enables detecting novel data representations and provides appropriate expansion signals without any supervised signals, unlike [57] which only considers supervised learning; (4) This paper adds many new experimental results about the unsupervised learning setting, which demonstrates that the proposed ODDL can achieve the best performance in both supervised and unsupervised learning.

The paper brings the following contributions :

- In the previous studies, the absence of a thorough theoretical analysis has left their forgetting behaviour unexplored in the more challenging Task Free Continual Learning (TFCL) scenario. In this study, we address this gap by introducing an innovative theoretical framework that conceptualizes memory and evolving data streams as the source and target distributions, respectively, while deriving a lower bound to characterize the dynamic shifts in model performance. The proposed framework offers novel insights into the forgetting behaviour of continual learning models.
- In contrast to the majority continual learning techniques developed so far, that overlook data distribution shifts in TFCL, we introduce an innovative method named the Online Discrepancy Distance Learning (ODDL) which incorporates a novel discrepancy-based expansion mechanism to identify data distribution shifts as an expansion signal for the model, eventually resulting in an efficient network architecture.

- Conventional sample selection techniques predominantly focus on retaining a diverse array of data samples across all encountered categories, which proves inadequate for our methodology. Consequently, we introduce an innovative sample selection strategy that prioritizes the retention of only those samples deemed distinct from previously acquired knowledge, thereby enhancing the diversity of knowledge among the model's components while improving its overall performance.
- The process of creating and training new experts with randomly initialized parameters results in a slow acquisition of new knowledge. To mitigate this limitation, we introduce an innovative expert initialization strategy. This mechanism evaluates the relevance between the knowledge already learned by each expert and the characteristics of incoming samples, utilizing the parameters of the most suitable expert to initialize a new one withing the DEM. This facilitates the rapid assimilation of novel information by means of the newly created expert.
- Experimental results show that ODDL provides better results than other baselines under several TFCL benchmarks.

The remainder of the paper is structured as follows. The literature review is discussed in Section 2. The general theoretical framework is introduced in Section 3, while this is expanded for formulating mixture systems for learning knowledge diversity representations in Section 4. The proposed methodology is provided in Section 5 and the experimental results are presented in Section 6. Section 7 draws the conclusions of this study.

2. Related Work

Despite achieving remarkable performances in individual tasks, most deep learning models are not able to learn a successions of different tasks due to catastrophic forgetting, [13, 64]. Continual learning is a specific approach in machine learning which aims at training a model to continually acquire and process knowledge from new tasks without forgetting previously learnt information. One of the most popular approaches for relieving forgetting is to introduce an extra regularization term into the main objective function, penalizing changes in certain parameters in order to preserve the most

important information from past tasks [15, 55]. Memory systems, which store data in order to be reused whenever necessary, have also been used as data representations of the past learnt tasks, [4, 6]. The critical idea for the memory-based approaches is to use a sample selection mechanism that selects appropriate training samples, characteristic for each task [11, 38]. Memory-based approaches can be combined with knowledge distillation [9, 10, 35] or regularization-based methods [19, 24, 47, 49] to further improve their performance. Memory-based approaches perform well in continual learning, however, storing real samples could lead to data privacy and security concerns [39].

A distinct approach consists of training a generative replay network (GRN), usually considered as a Variational Autoencoder (VAE) [26], or a Generative Adversarial Net (GAN) [20]. The GRN is used as a generator reproducing samples statistically consistent to those learnt from past tasks, in order to relieve forgetting when learning new tasks [5, 36, 45]. The pioneering work from [50] trains a GAN as a generative replay network together with a classifier to be used for classification. A twin GAN network framework was employed in [61], where the two networks learn alternatively from each other in tandem together with assimilating new information. Meanwhile, GRNs have been implemented into teacher-student frameworks in [44], where two VAEs are considered as teacher and student, respectively. After each task switch, the teacher and student would exchange their roles to enable knowledge accumulation during continual learning. However, this approach suffers from poor image-generation performance because VAEs tend to generate rather blurred images failing to provide high-quality knowledge transfer for training the student. This weakness was addressed by introducing a hybrid VAEGAN approach [52] or a GAN-based teacher module [56, 58], which considers the high-quality generation ability of a GAN as a teacher. Nevertheless, all these models are limited in the amount of tasks they can learn [53].

Recently, ensemble models [2, 18], or a dynamic expansion model [53], have been considered for learning longer streams of different datasets. The former models assume that the model knows the number of tasks and then associates sub-network components for integrating the learning of new tasks [2, 18]. However, such fixed network architectures can not handle the learning of an increasing number of tasks. The latter model

does not require knowing how many tasks are in advance, and can dynamically add new modules to a mixture model when learning novel tasks within a continual learning framework, [53].

Task-free continual learning (TFCL) represents a challenging and more realistic learning scenario [65], where a model can not access the task information and does not know task boundaries. The primary challenge for TFCL is that the model does not know exactly when and how the underlying data distribution changes over time and thus can not properly capture the data stream. The first work addressing catastrophic forgetting in TFCL was by Aljundi, Kelchtermans, and Tuytelaars [3]. This method was then extended in the Maximal Interfered Retrieval (MIR) [1] for learning a VAE model along with a classifier to provide generative replay samples. The Gradient Sample Selection (GSS) [4] formulates the replay buffer generation as a constrained optimization problem solved using quadratic programming. More recently, a *learner-evaluator* framework, called the Continual Prototype Evolution (CoPE) [16], was proposed to implement data selection in TFCL aiming at storing statistically balanced samples from all data categories achieving very good performances on imbalanced data streams. Modifying the memorized samples was investigated in the Gradient-based Memory EDiting (GMED) [23], and was shown that it can further improve the model's performance in TFCL. However, these approaches cannot learn long or infinite data streams.

More recent research studies have focused on the Dynamic Expansion Models (DEMs) for addressing catastrophic forgetting in TFCL. Compared to the fixed model, DEMs are scalable by adding new component modules for learning a dynamically changing data stream and can achieve better generalization results. Training a DEM under the TFCL paradigm was first investigated in the Continual Unsupervised Representation Learning (CURL) [45]. During the training, CURL dynamically adds new inference models to capture multiple underlying data distributions over time while generating past data samples to relieve forgetting. The main downside of CURL is that the generator is always updated with incoming and generative replay samples and would gradually forget previously learnt information over time. This issue was addressed in [33] which introduces a new VAE-based expansion framework called the Continual

Neural Dirichlet Process Mixture (CNDPM) that dynamically adds a new VAE component through a Dirichlet process-based expansion mechanism. Specifically, CNDPM updates a single expert and freezes all those previously trained in order to preserve past information. The selection of data samples was also used for enabling the training of dynamic expansion models with selected data, resulting in better results for TFCL [54]. However, these models result in non-optimal architectures because they do not ensure the learning of a diverse knowledge representation information when expanding their structures.

Discussion. In the TFCL learning scenario, memory-based and dynamic expansion methodologies represent the main strategies to address the issue of catastrophic forgetting in neural networks. The primary challenge associated with memory-based approaches lies in data privacy concerns and the degradation of model performance when processing extended sequences of data streams. Conversely, dynamic expansion techniques are hindered by the exponential increase in parameters, rendering them unsuitable for deployment on devices with limited resources, such as the Field-Programmable Gate Array (FPGA) based devices. This paper focuses on dynamic expansion strategies, proposing an innovative framework designed to enhance model efficacy while preserving a streamlined network structure. Unlike existing dynamic expansion models, this study introduces a novel mechanism that detects data distribution shifts as triggers for expansion, ensuring an optimal network configuration. Additionally, we introduce a new theoretical framework to analyze continual learning models, which is lacking in other continual learning studies. In addition, most existing dynamic expansion models rely on random sampling for memory buffer management [45], while our proposed method employs a novelty discrepancy-based measure to optimize the memory buffer, thereby facilitating the learning and assimilation of novel information by each newly instantiated expert.

3. Theoretical framework for TFCL

It is well known that catastrophic forgetting significantly impacts the performance of models in continual learning. However, lacking a theoretical analysis in contin-

Table 1: The description of some important notations.

Notations	Descriptions
$\mathcal{T} = \{\mathcal{T}_1, \cdots, \mathcal{T}_n\}$	Tasks, where n is the number of training times/steps.
$\mathcal{D}_t^S$	The training dataset.
$\mathcal{D}_t^T$	The testing dataset.
$\mathrm{P}_{\mathrm{t,j}}^{\mathrm{T},\mathcal{X}}$	The distribution of the training dataset.
$\mathcal{M}_i$	The memory buffer updated at the i-th training time.
$\mathcal{G} = \{G_1, \cdots, G_c\}$	The dynamic expansion model.
$\mathcal{A}$	The data stream.
$\mathcal{L}_{class}(G_j, \mathcal{M}_i)$	The classification loss.
$\mathcal{L}_{VAE}(G_j, \mathcal{M}_i)$	The VAE loss.
$\mathcal{L}_D(\mathrm{P}_{t,j}^{T,\mathcal{X}}, \mathrm{P}_{t,j}^{S,\mathcal{X}})$	The discrepancy distance.
$\{\mathbf{X}_i^b, \mathbf{Y}_i^b\}$	The i-th data batch.

ual learning limits our understanding of the forgetting behaviour exhibited by various models. A robust theoretical framework for continual learning is essential to enable researchers and readers to gain a deeper comprehension of the forgetting effects associated with different models. In order to address this, we propose an innovative theoretical analysis framework for continual learning, which conceptualizes forgetting as a specific domain adaptation challenge. Furthermore, we provide a theoretical examination of the proposed model expansion mechanism, which demonstrates that our approach can achieve an optimal network architecture while ensuring superior generalization performance. Specifically, we adopt and expand the findings from [40] in order to formulate a novel theoretical framework for TFCL. Through this framework we define network forgetting as a generalization error of the risk bounds. We provide a short description of many important notations used in this paper in Table 1.

3.1. Preliminaries

Let us consider the input space as $\mathcal{X}$, while the output space is $\mathcal{Y}$, representing class information. We consider the training dataset as $\mathcal{D}_i^S = \{\mathbf{x}_j^S, y_j^S\}_{j=1}^{N_i^S}$, while the testing dataset is $\mathcal{D}_i^T = \{\mathbf{x}_j^T, y_j^T\}_{j=1}^{N_i^T}$, where the data is denoted as $\mathbf{x}_j^T \in \mathcal{X}$ and $y_j^T \in \mathcal{Y}$ is

its corresponding label; N_i^S and N_i^T represent the number of samples for $\mathcal{D}_i^S$ and $\mathcal{D}_i^T$.

Definition 1. ***(The Task-Freee Continual Learning (TFCL) setting).*** *The most popular continual learning paradigm is the class-incremental setting, in which each new task contains data samples from a certain set of data categories [3]. Under this setting, a training dataset $\mathcal{D}_t^S$ is divided into c_t^S parts $\{\mathcal{D}_{t,1}^S, \cdots, \mathcal{D}_{t,c_t^S}^S\}$, according to the category information, where each part $\mathcal{D}_{t,j}^S$ contains samples from one or several successive classes. The distributions for $\mathcal{D}_{t,j}^T$ and $\mathcal{D}_{t,j}^S$, are denoted as $\mathrm{P}_{\mathrm{t,j}}^{\mathrm{T}}$ and $\mathrm{P}_{\mathrm{t,j}}^{\mathrm{S}}$, respectively. We consider $\mathrm{P}_{\mathrm{t,j}}^{\mathrm{T},\mathcal{X}}$ as the marginal j-th training distribution over $\mathcal{X}$. In the class-incremental setting, a data stream is formed by $\mathcal{A} = \bigcup_{j=1}^{c_t^S} \mathcal{D}_{t,j}^S$. We also consider that learning $\mathcal{A}$ requires n training steps/times, $\mathcal{T} = \{\mathcal{T}_1, \cdots, \mathcal{T}_n\}$. At each training time $\mathcal{T}_i$, we use only a small set of samples $\{\mathbf{X}_i^b, \mathbf{Y}_i^b\} = \{\mathbf{x}_j^S, y_j^S\}_{j=1}^b \in \mathcal{A}$, where b is the batch size, drawn from a specific set of data categories, for training, while all previous training data sets $\{\mathbf{X}_1^b, \mathbf{Y}_1^b, \cdots, \mathbf{X}_{i-1}^b, \mathbf{Y}_{i-1}^b\}$ are no longer available.*

Unlike TFCL, task-aware continual learning assumes that the model can access task information during both training and testing phases. We illustrate the differences between TFCL and task-aware continual learning in Fig. 1. We can observe that the model can access the task information and boundaries in the task-aware continual learning. In addition, the task information is also available during the inference phase under the task-aware continual learning. In contrast, a model optimized under the task-free continual learning does not have any task information, and this scenario presents a significant learning challenge.

Definition 2. ***(Model and memory).*** *Let h be a classifier implemented by either a deep convolution or a fully connected network, while $\mathcal{H} = \{h \,|\, h \colon \mathcal{X} \to \mathcal{Y}\}$ represents the classifiers' space. We also consider $\mathcal{M}_i$ as a memory of maximum size $|\mathcal{M}_i|^{Max}$, where i denotes that $\mathcal{M}_i$ is updated at $\mathcal{T}_i$. The probability distribution of $\mathcal{M}_i$ is $\mathrm{P}_{\mathcal{M}_\mathrm{i}}$.*

3.2. Measuring Changes in Probabilistic Representations

A discrepancy distance can be used to evaluate distances between probabilistic representations. Such probabilistic distances have been successfully used for deriving generalization bounds (GB) for domain adaptation [7, 14, 40], or for deriving GAN-based

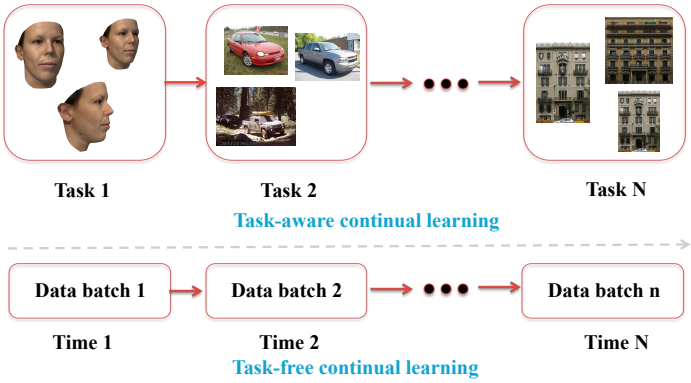


Figure 1: The differences between task-aware and task-free continual learning, respectively.

objective functions to compare real and generated (fake) data distributions during training [17, 34, 37]. In our study, such discrepancy distances are used for analyzing the forgetting behaviour. The key idea is to measure the difference between probabilistic representation between the source dataset, consisting of all previously seen data samples (during the learning of different tasks), and the target dataset from the memory buffer, and investigate how this measure affects the result when considering the target dataset for training. We start by introducing the model risk.

Definition 3. ***(The model risk [40])*** *in the machine learning field evaluates the errors made by the model for the given data samples and represents an assessment measure of the model's performance. A high risk indicates that the model suffers from significant performance loss. In contrast, a low risk indicates that the model has a good generalization performance on the data samples. For a given distribution* $\mathrm{P}^{S}_{t,j}$*, the model h risk is defined as :*

$$\mathcal{L}_{\mathrm{P}^{S}_{t,j}}(h, f_{\mathrm{P}^{S}_{t,j}}) := \mathbb{E}_{\{\mathbf{x}\}\sim\mathrm{P}^{S}_{t,j}}\left[\mathcal{L}(f_{\mathrm{P}^{S}_{t,j}}(\mathbf{x}), h(\mathbf{x}))\right], \tag{1}$$

where $\mathcal{L}\colon \mathcal{Y}\times\mathcal{Y}\to[0,1]$ *is the loss function, and* $f_{\mathrm{P}^{S}_{t,j}}(\cdot)$ *returns the true class label for a data sample drawn from* $\mathrm{P}^{S}_{t,j}$*.* $\mathbb{E}$ *denotes the statistical expectation.*

Definition 4. ***(Discrepancy distance [40].)*** *The discrepancy distance* $\mathcal{L}_D$ *is defined on two marginals, corresponding to two probability densities* $\mathrm{P}^{T,\mathcal{X}}_{t,j}$ *and* $\mathrm{P}^{S,\mathcal{X}}_{t,j}$*, as :*

$$\mathcal{L}_D(\mathrm{P}^{T,\mathcal{X}}_{t,j}, \mathrm{P}^{S,\mathcal{X}}_{t,j}) := \max_{(h,h')\in\mathcal{H}^2}\left|\mathbb{E}_{\mathbf{x}\sim\mathrm{P}^{T,\mathcal{X}}_{t,j}}[\mathcal{L}(h(\mathbf{x}), h'(\mathbf{x}))] - \mathbb{E}_{\mathbf{x}\sim\mathrm{P}^{S,\mathcal{X}}_{t,j}}[\mathcal{L}(h(\mathbf{x}), h'(\mathbf{x}))]\right|, \tag{2}$$

where $\{h, h'\} \in \mathcal{H}$ are evaluated for specific data samples, in order to estimate a statistical distance measure. Specifically, we search for h and h' that can maximize the distance measure defined in Eq. (2)*. In practice, the discrepancy distance $\mathcal{L}_D(\cdot,\cdot)$ is evaluated as the upper bound by employing the Rademacher complexity, used in the domain adaptation theory as a measure of the hypothesis space richness [41, 63].*

Corollary 1. *For two given domains $\mathrm{P}_{t,j}^{T,\mathcal{X}}$ and $\mathrm{P}_{t,j}^{S,\mathcal{X}}$, let U_{P} and $U_{\mathbb{P}}$ represent sample sets of sizes $m_{\mathcal{P}}$ and $m_{\mathbb{P}}$, drawn independently from $\mathrm{P}_{t,j}^{T,\mathcal{X}}$ and $\mathrm{P}_{t,j}^{S,\mathcal{X}}$. Let as consider $\widehat{\mathrm{P}}_{t,j}^{T,\mathcal{X}}$ and $\widehat{\mathrm{P}}_{t,j}^{S,\mathcal{X}}$ as the empirical distributions for $U_{\mathcal{P}}$ and $U_{\mathbb{P}}$, respectively. Let $\mathcal{L}(\mathbf{x}', \mathbf{x}) = |\mathbf{x}' - \mathbf{x}|^q$ be a loss function where $q = 1$, defined as the L1-Norm, satisfying $\forall(\mathbf{x}, \mathbf{x}') \in \mathcal{X}, \mathcal{L}(\mathbf{x}, \mathbf{x}') > M$, where $M > 0$. Then, with the probability of $1 - \delta$, where δ is a small quantity, we have [40] :*

$$\mathcal{L}_D(\mathrm{P}_{t,j}^{T,\mathcal{X}}, \mathrm{P}_{t,j}^{S,\mathcal{X}}) \leq \mathcal{L}_D(\widehat{\mathrm{P}}_{t,j}^{T,\mathcal{X}}, \widehat{\mathrm{P}}_{t,j}^{S,\mathcal{X}}) + B^{\star} + 3M\left(\sqrt{\frac{\log\left(\frac{4}{\delta}\right)}{2m_{\mathrm{P}}}} + \sqrt{\frac{\log\left(\frac{4}{\delta}\right)}{2m_{\mathcal{P}}}}\right), \tag{3}$$

where $B^{\star} = 4q\left(\mathrm{Re}_{U_{\mathcal{P}}}(\mathcal{H}) + \mathrm{Re}_{U_{\mathbb{P}}}(\mathcal{H})\right)$ and $\mathrm{Re}_{U_{\mathcal{P}}}(\mathcal{H})$ is the Rademacher complexity [40]. We denote the expression from the Right-Hand Side (RHS) of Eq. (3) *by $\widehat{\mathcal{L}}(\mathrm{P}_{t,j}^{T,\mathcal{X}}, \mathrm{P}_{t,j}^{S,\mathcal{X}})$. We consider $\mathcal{L}\colon \mathcal{Y} \times \mathcal{Y} \to [0, 1]$ as a bounded and symmetric loss function $\forall(y, y') \in \mathcal{Y}^2, \mathcal{L}(y, y') \leq U'$, $U' > 0$, where $\mathcal{L}(\cdot,\cdot)$ obeys the triangle inequality.*

3.3. Forgetting Analysis for A Fixed Model

In the following we derive the generalization bound for the forgetting analysis of a fixed model by using the definitions from the previous section.

Theorem 1. *Let $\mathcal{P}_i$ be the distribution of all previously seen data samples drawn from the data stream $\mathcal{A}$ at $\mathcal{T}_i$. Let $h_{\mathcal{M}_i} = \arg\min_{h \in \mathcal{H}} \mathcal{L}_{\mathrm{P}_{\mathcal{M}_i}}(h, f_{\mathrm{P}_{\mathcal{M}_i}})$ and $h_{\mathcal{P}_i} = \arg\min_{h \in \mathcal{H}} \mathcal{L}_{\mathcal{P}_i}(h, f_{\mathcal{P}_i})$ be the optimal classifiers for $\mathrm{P}_{\mathcal{M}_i}$ and $\mathcal{P}_i$, respectively. The Generalized Bound (GB) between $\mathcal{P}_i$ and $\mathrm{P}_{\mathcal{M}_i}$ is derived as :*

$$\mathcal{L}_{\mathcal{P}_i}(h, f_{\mathcal{P}_i}) \leq \mathcal{L}_{\mathrm{P}_{\mathcal{M}_i}}(h, h_{\mathcal{M}_i}) + \widehat{\mathcal{L}}(\mathcal{P}_i^{\mathcal{X}}, \mathrm{P}_{\mathcal{M}_i}^{\mathcal{X}}) + \eta(\mathcal{P}_i, \mathrm{P}_{\mathcal{M}_i}), \tag{4}$$

with $\eta(\mathcal{P}_i, \mathrm{P}_{\mathcal{M}_i})$ defined as :

$$\eta(\mathcal{P}_i, \mathrm{P}_{\mathcal{M}_i}) = \mathcal{L}_{\mathcal{P}_i}(h_{\mathcal{P}_i}, h_{\mathcal{M}_i}) + \mathcal{L}_{\mathcal{P}_i}(h_{\mathcal{P}_i}, f_{\mathcal{P}_i}), \tag{5}$$

where $f_{\mathcal{P}_i}$ is the target function for $\mathcal{P}_i$, which returns the true class label for each sample drawn from $\mathcal{P}_i$.

Proof. Firstly, we assume that the classifier $h \in \mathcal{H}$ is fixed and $\mathcal{L}(\cdot,\cdot)$ has the triangle inequality property according to *Definition 4*. we have:

$$\begin{aligned}\mathcal{L}_{\mathcal{P}_i}(h, f_{\mathcal{P}_i}) &\leq \mathcal{L}_{\mathcal{P}_i}(h, h_{\mathcal{M}_i}) + \mathcal{L}_{\mathcal{P}_i}(h_{\mathcal{P}_i}, h_{\mathcal{M}_i}) + \mathcal{L}_{\mathcal{P}_i}(h_{\mathcal{P}_i}, f_{\mathcal{M}_i}) \\ &\leq \mathcal{L}_{\mathcal{M}_i}(h, h_{\mathcal{M}_i}) + \mathcal{L}_d(\mathcal{P}_i, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}) + \mathcal{L}_{\mathcal{P}_i}(h_{\mathcal{P}_i}, h_{\mathcal{M}_i}) + \mathcal{L}_{\mathcal{P}_i}(h_{\mathcal{P}_i}, f_{\mathcal{P}_i})\,.\end{aligned} \tag{6}$$

According to the results from Corollary 1, we have :

$$\begin{aligned}\mathcal{L}(\mathcal{P}^{\mathcal{X}}_i, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}) \leq \mathcal{L}(\widehat{\mathcal{P}}^{\mathcal{X}}_i, \widehat{\mathrm{P}}^{\mathcal{X}}_{\mathcal{M}_i}) + 4q\left(\mathrm{Re}_{U_{\mathcal{P}^{\mathcal{X}}_i}}(\mathcal{H}) + \mathrm{Re}_{U_{\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}}}(\mathcal{H})\right) \\ + 3M\left(\sqrt{\frac{\log\left(\frac{4}{\delta}\right)}{2m_{\mathcal{P}^{\mathcal{X}}_i}}} + \sqrt{\frac{\log\left(\frac{4}{\delta}\right)}{2m_{\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}}}}\right),\end{aligned} \tag{7}$$

where $U_{\mathcal{P}^{\mathcal{X}}_i}$ and $U_{\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}}$ represent samples of sizes $m_{\mathcal{P}^{\mathcal{X}}_i}$ and $m_{\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}}$, drawn independently from $\mathcal{P}^{\mathcal{X}}_i$ and $\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}$. $\widehat{\mathcal{P}}^{\mathcal{X}}_i$ and $\widehat{\mathrm{P}}^{\mathcal{X}}_{\mathcal{M}_i}$ represent the empirical distributions of $U_{\mathcal{P}^{\mathcal{X}}_i}$ and $U_{\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}}$. We consider $\mathcal{L}_{\widehat{d}}(\mathcal{P}^{\mathcal{X}}_i, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i})$ to represent the right hand side of Eq. (7). Then we rewrite Eq. (6) as :

$$\mathcal{L}_{\mathcal{P}_i}(h, f_{\mathcal{P}_i}) \leq \mathcal{L}_{\mathcal{M}_i}(h, h_{\mathcal{M}_i}) + \widehat{\mathcal{L}}(\mathcal{P}_i, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}) + \mathcal{L}_{\mathcal{P}_i}(h_{\mathcal{P}_i}, h_{\mathcal{M}_i}) + \mathcal{L}_{\mathcal{P}_i}(h_{\mathcal{P}_i}, f_{\mathcal{P}_i})\,. \tag{8}$$

Theorem 1 explicitly measures the gap between the predictions of the model h at each task $\mathcal{T}_i$ and the corresponding labels of the data. The memory buffer $\mathcal{M}_i$ is initially able to store all previously seen data samples and thus the model can achieve a small target risk on $\mathcal{P}_i$. While the model learns additional data batches over time (i increases), the discrepancy distance term $\widehat{\mathcal{L}}(\mathcal{P}^{\mathcal{X}}_i, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i})$ in Eq. (4) gradually increases leading to a larger target risk on $\mathcal{P}_i$. This is caused by the memory buffer that ignores some past data samples due to its limited memory capacity. Such performance degeneration eventually leads to the model's forgetting. Next we assess the forgetting of a model on the testing dataset:

Theorem 2. *Let $\mathrm{P}^{\mathcal{T}}_{t,j}$ be a target domain, we can introduce a GB for analyzing the forgetting process of a static model at each training time $\mathcal{T}_i$:*

$$\mathcal{L}_{\mathrm{P}^{\mathcal{T}}_{t,j}}(h, f_{\mathrm{P}^{\mathcal{T}}_{t,j}}) \leq \mathcal{L}_{\mathrm{P}_{\mathcal{M}_i}}(h, h_{\mathcal{M}_i}) + \widehat{\mathcal{L}}(\mathrm{P}^{\mathcal{T},\mathcal{X}}_{t,j}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}) + \eta(\mathrm{P}^{\mathcal{T}}_{t,j}, \mathrm{P}_{\mathcal{M}_i})\,. \tag{9}$$

Theorem 2's proof is similar to that for Theorem 1. The results from Eq. (9) show that the model's performance on the target distribution $\mathrm{P}^{\mathrm{T}}_{\mathrm{t,j}}$ relies mainly on the discrepancy distance between the memory distribution $\mathrm{P}_{\mathcal{M}_i}$ and the target distribution $\mathrm{P}^{T,\mathcal{X}}_{t,j}$. Next, we derive a new GB for evaluating multiple target distributions based on the results from Theorem 2.

Lemma 1. *Let $\mathcal{A} = \bigcup_{j=1}^{c_t^S} \mathcal{D}^S_{t,j}$ be a data stream constructed by collecting samples from the training dataset $\mathcal{D}^S_t$. For a given corresponding testing set $\mathcal{D}^T_t$, let us define the testing data distribution for the j-th class as $\mathrm{P}^{\mathrm{T}}_{\mathrm{t,j}}$. Then we have the following GB for multiple target domains :*

$$\sum\nolimits_{j=1}^{c_t^T} \left\{ \mathcal{L}_{\mathrm{P}^T_{t,j}}(h, f_{\mathrm{P}^T_{t,j}}) \right\} \leq \sum\nolimits_{j=1}^{c_t^T} \left\{ \mathcal{L}_{\mathrm{P}_{\mathcal{M}_i}}(h, h_{\mathcal{M}_i}) + \widehat{\mathcal{L}}(\mathrm{P}^{T,\mathcal{X}}_{t,j}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}) + \eta(\mathrm{P}^T_{t,j}, \mathrm{P}_{\mathcal{M}_i}) \right\}, \quad (10)$$

where c_t^T is the number of underlying data distributions corresponding to the testing dataset.

Remarks. Lemma 1 has the following observations :

- For achieving the best results on the testing dataset, one should minimize the discrepancy distance between each $\mathrm{P}^{T,\mathcal{X}}_{t,j}$, representing the target distribution, and the memory distribution $\mathrm{P}_{\mathcal{M}_i}$ for each $\mathcal{T}_i$.
- Having an optimal memory buffer is essential for combating catastrophic forgetting in continual learning, as shown in [27]. However, that theoretical study can only be used in continual learning assuming that the task information is provided. Meanwhile, Eq. (10) is used to evaluate the memory quality in a more realistic learning scenario, such as the Task Free Continual Learning (TFCL). The connection between the memory buffer and catastrophic forgetting in various approaches including [12, 16], without knowing task boundaries or labels, can be explained through Eq. (10).

3.4. Forgetting Analysis for the Dynamic Expansion Model

According to Theorem 1, a memory-based learning system suffers from decreasing performance when learning successive data streams, characterized by multiple different

underlying data distributions. Another issue is the negative backward transfer [38], where the model's performance gradually decreases due to the training on samples drawn from entirely different underlying distributions [25]. A Dynamic Expansion Model (DEM) is not affected by such limitations and can address the learning of infinite data streams by adding new components for representing new underlying data sets over time. A DEM has the following advantages over a fixed model : 1) It relieves the negative transfer by acquiring previously learnt information in the components, added to its network structure, which is progressively frozen after training; 2) It can achieve better generalization performance while learning a long-term data stream under TFCL.

Definition 5. ***(Dynamic expansion mechanism (DEM).)*** *Let us consider a DEM as* $\mathcal{G} = \{G_1, \cdots, G_c\}$*, where* G_j *is the* j*-th component, in the mixture/ensemble model* $\mathcal{G}$*, which is implemented by a classifier. Initially, the mixture model* $\mathcal{G}$ *trains its first component* G_1*, and during the following learning stages,* $\mathcal{G}$ *adds new components while freezing those components that have been previously trained, to avoid forgetting.*

Theorem 3. *Let* $\mathcal{A} = \{\mathbf{X}_1^b, \cdots, \mathbf{X}_n^b\}$ *be a data stream in which the data samples are obtained from the m-th training dataset and* $\mathcal{P}_{(m,j)}$ *be the probability density function of the* j*-th training data batch* $\mathbf{X}_j^b \in \mathcal{A}$*, visited at the* j*-th training time (*$\mathcal{T}_j$*). At* $\mathcal{T}_j$ *we consider that we have already trained c components for the DEM,* $\mathcal{G} = \{G_1, \cdots, G_c\}$*. Let us define* $\mathcal{T} = \{\mathcal{T}_{k_1}, \cdots, \mathcal{T}_{k_c}\}$ *as the times of training for DEM, with the component* G_v *finishing its training and afterwards being frozen at* $\mathcal{T}_{k_v}$*. The GB for assessing the changes in* $\mathcal{G}$ *at* $\mathcal{T}_i$ *is given by :*

$$\frac{1}{n}\sum\nolimits_{j=1}^{n}\left\{\mathcal{L}_{\mathcal{P}_{(m,j)}}(h, f_{\mathcal{P}_{(m,j)}})\right\} \leq \frac{1}{n}\sum\nolimits_{j=1}^{n}\left\{\mathrm{F}_S(\mathcal{P}_{(m,j)}, \mathcal{G})\right\}, \tag{11}$$

where $\mathrm{F}_S(\cdot,\cdot)$ *is the selection function that returns the estimated risk of the selected component :*

$$\mathrm{F}_S(\mathcal{P}_{(m,j)}, \mathcal{G}) \triangleq \min_{v=1,\cdots,c}\left\{\mathcal{L}_{\mathrm{P}_{\mathcal{M}_{k_v}}}(h, h_{\mathcal{M}_{k_v}}) + \widehat{\mathcal{L}}(\mathcal{P}^{\mathcal{X}}_{(m,j)}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_{k_v}}) + \eta(\mathcal{P}_{(m,j)}, \mathrm{P}_{\mathcal{M}_{k_v}})\right\}, \tag{12}$$

where $\mathrm{P}_{\mathcal{M}_{k_v}}$ *represents the memory distribution at* $\mathcal{T}_{k_v}$*.* $\mathcal{P}^{\mathcal{X}}_{(m,j)}$ *and* $\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_{k_v}}$ *denote the marginal distribution of* $\mathcal{P}_{(m,j)}$ *and* $\mathrm{P}_{\mathcal{M}_{k_v}}$*, respectively.* $\mathrm{F}_S(\cdot,\cdot)$ *represents an ideal model selection for the minimal risk component. The dynamic expansion model achieves a*

lower upper bound for the risk, given by the Left Hand Side (LHS) of Eq. (11)*, than the fixed model, defined by Theorem 1, resulting in better generalization performance.*

Proof. We consider $\mathcal{P}_i$ to be a mixture distribution :

$$p_{(1:m)}(\mathbf{x}) = \frac{1}{m}\sum_{j=1}^{m} p_j(\mathbf{x}), \tag{13}$$

where each $p_j(\mathbf{x})$ represents the density function for a batch of samples :

$$\mathcal{L}_{\mathcal{P}_j}(h, f_{\mathcal{P}_i}) = \int p_j(\mathbf{x})\mathcal{L}(h, f_{\mathcal{P}_j})\,\mathrm{d}\mathbf{x}, \tag{14}$$

and when we replace this into Eq. (13), we have :

$$p_{(1:m)}(\mathbf{x}) = \frac{1}{m}\sum_{j=1}^{m}\int p_j(\mathbf{x})\mathcal{L}(h, f_{\mathcal{P}_m})\,\mathrm{d}\mathbf{x}\,. \tag{15}$$

Let us define $\mathcal{P}_{(j,m)}$ as the distribution of $p_j(\mathbf{x})$. Since $f_{\mathcal{P}_m}$ is the true labeling function for $\mathcal{P}_m$, we decompose $f_{\mathcal{P}_m}$ into j components $\{f_{\mathcal{P}_{(m,1)}}, \cdots, f_{\mathcal{P}_{(m,j)}}\}$, where each $f_{\mathcal{P}_{(m,j)}}$ is the true labeling function for $\mathcal{P}_{(m,j)}$, then we can rewrite Eq. (15) as :

$$\frac{1}{m}\sum_{j=1}^{m}\int p_j(\mathbf{x})\mathcal{L}(h, f_{\mathcal{P}_{(m,j)}})\,\mathrm{d}\mathbf{x}\,. \tag{16}$$

Then we rewrite Eq. (16) as the expectation form $\frac{1}{m}\sum_{j=1}^{m}\mathcal{L}_{\mathcal{P}_{(i,j)}}(h, f_{\mathcal{P}_{(i,j)}})$. Then we derive GB for this expectation from as :

$$\frac{1}{m}\sum_{j=1}^{m}\mathcal{L}_{\mathcal{P}_{(m,j)}}(h, f_{\mathcal{P}_{(m,j)}}) \leq \frac{1}{m}\sum_{j=1}^{m}\mathrm{F}_S(\mathcal{P}_{(m,j)}, \mathcal{G})\,. \tag{17}$$

In the following section we link the dynamic expansion of the proposed model to the efficiency of its knowledge diversity representation.

4. Theoretical analysis of the knowledge diversity representation

A mixture system is expected to contain components representing diverse knowledge. This requirement together with defining its complexity (model size and number of parameters) plays a critical role in the model's performance and its overall efficiency for the TFCL. By ensuring the learning of diverse information by each component leads to a well-defined network architecture which is compact. Maintaining an optimal model size together with capturing well the representations of all past learnt data should go hand-in-hand, as given by the following lemma:

Lemma 2. *Let $\mathcal{A} = \bigcup_{j}^{c_t^S} \mathcal{D}_{t,j}^S$ be a data stream consisting of samples from the training dataset $\mathcal{D}_t^S$. We also have a set of testing datasets $\{\mathcal{D}_{t,1}^T, \cdots, \mathcal{D}_{t,C_t^T}^T\}$ which are used for the evaluation, with each $\mathcal{D}_{t,j}^T$ containing $c_{t,j}^b$ data batches. Let $\mathrm{P}_{t,j}^T(k)$ be the probability density of the k-th data batch in $\mathcal{D}_{t,j}^T$. We consider that we have already learnt c components for $\mathcal{G}$ at $\mathcal{T}_i$. Then the GB for multiple target domains is derived as :*

$$\sum_{j=1}^{c_t^T} \Big\{ \sum_{k=1}^{c_{t,j}^b} \mathcal{L}_{\mathrm{P}_{t,j}^T(k)}(h, f_{\mathrm{P}_{t,j}^T(k)}) \Big\} \leq \sum_{j=1}^{c_t^T} \Big\{ \sum_{k=1}^{c_{t,j}^b} \mathrm{F}_S(\mathrm{P}_{t,j}^T(k), \mathcal{G}) \Big\}, \tag{18}$$

Remark. *We have the following observations from Lemma 2 :*

- *The mixture model $\mathcal{G}$ generalization evaluated on the target distributions relies on the discrepancy distance between the memory distribution $\mathrm{P}_{\mathcal{M}_{k_i}}$ of the selected component and each target distribution $\mathrm{P}_{t,j}^T$.*
- *If the mixture model $\mathcal{G}$ adds more components, then it can learn more information about each target's probabilistic representation while also improving its generalization. Conversely, by training very few components in the mixture model $\mathcal{G}$ may not allow for capturing the entire underlying data distribution.*
- *A similar theoretical framework for analysing forgetting in continual learning, but which requires accessing the task boundaries, was provided in [56]. However, this theoretical framework provides a more realistic GB evaluation for the forgetting process of a dynamic continual learning model.*

At the testing stage, we employ the following criterion $\widehat{\mathrm{F}}(\cdot,\cdot)$ for comparing the sample log-likelihoods of two distributions, in order to enable model selection without requiring the task information :

$$\widehat{\mathrm{F}}(\mathrm{P}_{t,j}^T(k), \mathcal{G}) \triangleq \arg\max_{v=1,\cdots,c} \{\mathbb{E}_{\mathbf{x}\sim\mathrm{P}_{t,j}^T(k)}[\hat{f}(\mathbf{x}, G_{k_v})]\}, \tag{19}$$

where $\hat{f}(\cdot,\cdot)$ is the log-likelihood data function. Then Eq. (19) is used for model selection :

$$\widehat{\mathrm{F}}_S(\mathrm{P}_{t,j}^T(k), \mathcal{G}) = \mathcal{L}_{\mathrm{P}_{\mathcal{M}_s}}(h, h_{\mathcal{M}_s}) + \widehat{\mathcal{L}}(\mathrm{P}_{t,j}^{T,\mathcal{X}}(k), \mathrm{P}_{\mathcal{M}_s}^{\mathcal{X}}) + \eta(\mathrm{P}_{t,j}^T(k), \mathrm{P}_{\mathcal{M}_s}) \tag{20}$$

By considering Eq. (18) under real circumstances, we have :

$$\sum_{j=1}^{c_t^T} \Big\{ \sum_{k=1}^{c_{t,j}^b} \mathcal{L}_{\mathrm{P}_{t,j}^T(k)}(h, f_{\mathrm{P}_{t,j}^T(k)}) \Big\} \leq \sum_{j=1}^{c_t^T} \Big\{ \sum_{k=1}^{c_{t,j}^b} \{\widehat{\mathrm{F}}_S(\mathrm{P}_{t,j}^T(k), \mathcal{G})\} \Big\}. \tag{21}$$

When comparing Eq. (21) to the ideal solution from Eq. (18) we observe that the additional error terms are due to the model selection process, defined by Eq. (20). These error terms are given by :

$$\sum_{j=1}^{C_i^T} \Big\{ \sum_{k=1}^{c_{t,j}^b} \{\widehat{\mathrm{F}}_S(\mathrm{P}_{t,j}^T(k), \mathcal{G}) - \mathrm{F}_S(\mathrm{P}_{t,j}^T(k), \mathcal{G})\}\Big\}. \tag{22}$$

In the following we investigate the trade-off between the model's performance and its complexity. According to the results from Lemma 2, the GB for a dynamic expansion model $\mathcal{G}$ is defined by Eq. (18), where we assume that we have already learnt c components for the mixture model $\mathcal{G} = \{G_1, \cdots, G_c\}$ at $\mathcal{T}_i$. An optimal mixture model ensures that each component represents a probabilistic representation corresponding to a specific target data category according to Eq. (18), ensuring an appropriate balance between the model's complexity, given by the number of components and its generalization performance. While the mixture model $\mathcal{G}$ adds and learns additional distinct components it accumulates more knowledge about each target set, improving its performance. However, if some of these components share similar information with each other, then $\mathcal{G}$ is not learning as efficiently as expected, resulting in degenerated performance. Learning an optimal mixture model can be achieved in two ways : (1) One can measure the data distribution shift to guide the model expansion; (2) Maintaining the discrepancy distance between different components leads to better modeling of the knowledge diversity while resulting in a compact mixture model.

In the following, we analyse how encouraging the diversity of the trained mixture components leads to better performance.

Assumption 1. Let $\mathcal{G} = \{G_1, \cdots, G_c\}$ be a mixture model that has already learnt c components at the i-th training time ($\mathcal{T}_i$). Let us define a training timed set $\mathcal{K} = \{\mathcal{T}_{k_1}, \cdots, \mathcal{T}_{k_c}\}$, where each $\mathcal{T}_{k_j}$ represents the time when G_j was trained. After considering that Eq. (12) implements the ideal selection process, we observe that the mixture system $\mathcal{G}$, considered as a single model h, has learnt the information from the current memory $\mathcal{M}_i$ and all prior memory buffers, formed during the continual learning $\{\mathcal{M}_{k_1}, \cdots, \mathcal{M}_{k_{c-1}}\}$, where $\mathcal{M}_{k_c} \equiv \mathcal{M}_i$. This assumption is reasonable, given that Eq. (12) always selects the best component characterized by a minimal risk. Let us define $\mathrm{P}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}$ as the probabilistic representation of all learnt memory buffers

$\{\mathcal{M}_{k_1}, \cdots, \mathcal{M}_{k_{c-1}}\}$ and $\mathcal{M}_i$, at $\mathcal{T}_i$.

Theorem 4. *Let $\mathcal{A}$ be a data stream and $\mathcal{G} = \{h_1, \cdots, h_c\}$ be a mixture model that has learnt c components at the step $\mathcal{T}_i$. Let us define $\mathcal{P}_i$ as the distribution of all previously seen samples at $\mathcal{T}_i$. After considering Assumption 1 we have a GB as :*

$$\mathcal{L}_{\mathcal{P}_i}(h, f_{\mathcal{P}_i}) \leq \mathcal{L}_{\mathrm{P}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}}(h, h_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}) + \widehat{\mathcal{L}}(\mathcal{P}_i^{\mathcal{X}}, \mathrm{P}^{\mathcal{X}}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}) + \eta(\mathcal{P}_i, \mathrm{P}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}) \,. \tag{23}$$

Remark. We have several observations from Theorem 4 :

- The performance of the proposed mixture model on $\mathcal{P}_i$ is based on the discrepancy distance between $\mathrm{P}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}$ and $\mathcal{P}_i$. The diversity of information from $\mathrm{P}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}$, representing the entire information from all learnt data memory buffers including the current one, defines the efficient training of the model. We would have significant redundancy, if all memory buffers would capture similar information. On the other hand, when the memory buffers correspond to non-overlapping underlying probability density functions, leads to capturing diverse modes of the overall underlying information representation $\mathcal{P}_i$ with a minimal number of c components.
- Theorem 4 demonstrates that the probabilistic diversity among the learnt components is crucial for relieving forgetting in the mixture model.

The next theorem shows that by ensuring the probabilistic diversity of $\mathcal{G}$'s components leads to a tighter generalization bound.

Theorem 5. *Let us consider $\mathcal{A} = \bigcup_{C_i^S}^{j} \mathcal{D}_{t,j}^S$, a data stream consisting of samples from $\mathcal{D}_t^S$. We also have the target sets $\{\mathcal{D}_{t,1}^T, \cdots, \mathcal{D}_{t,C_t^T}^T\}$, of probabilistic representation $\mathrm{P_t^T}$. According to Assumption 1 we have a GB as :*

$$\mathcal{L}_{\mathrm{P}_t^T}(h, f_{\mathrm{P}_t^T}) \leq \mathcal{L}_{\mathrm{P}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}}(h, h_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}) + \widehat{\mathcal{L}}(\mathrm{P}_t^{T,\mathcal{X}}, \mathrm{P}^{\mathcal{X}}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}) + \eta(\mathrm{P}_t^T, \mathrm{P}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}) \,. \tag{24}$$

From Eq. (24)*, we deduce that while more data becomes available when increasing $\mathcal{T}_i$, the mixture model $\mathcal{G}$ gradually gains more knowledge improving its generalization*

performance. By training an optimally diverse number of components in $\mathcal{G}$, for the continual representation of $\mathrm{P}_{\bigcup_{t=k_1}^{k_{c-1}}\{\mathcal{M}_t\}\otimes\mathcal{M}_i}$ *we capture a diversity of information.*

5. Methodology

This section describes the proposed methodology. We firstly introduce the training of the proposed Online Discrepancy Distance Learning (ODDL) model in Section 5.1. Then, we present the discrepancy-based mixture expansion model in Section 5.2. A more efficient expansion strategy is presented in Section 5.3. In Section 5.4 we provide an approach for selecting data for ensuring that a diverse information is stored in the memory buffer. In Section 5.5 we outline the underlying algorithm.

5.1. The ODDL's Training

We consider that each component $G_j \in \mathcal{G}$ in the proposed mixture model ODDL has associated a classifier h_j, implemented by a network $f_{\varsigma_j}(\mathbf{x})$ parameterized by ς_j as well as a VAE for model selection. When the j-th component is trained on $\mathcal{M}_i$ at $\mathcal{T}_i$, we define two loss functions for training the classifier h and VAE as :

$$\mathcal{L}_{class}(G_j, \mathcal{M}_i) := \frac{1}{|\mathcal{M}_i|}\sum_{t=1}^{|\mathcal{M}_i|}\left\{\mathcal{L}_{ce}(h_j(\mathbf{x}_t), \mathbf{y}_t)\right\}, \tag{25}$$

$$\mathcal{L}_{VAE}(G_j, \mathcal{M}_i) := -\mathbb{E}_{q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x})}\left[\log p_{\theta_j}(\mathbf{x}\,|\,\mathbf{z})\right] + D_{KL}\left[q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x})\,\|\,p(\mathbf{z})\right], \tag{26}$$

where $\{\mathbf{x}_t, \mathbf{y}_t\}$ are paired data and its class label from the memory buffer $\mathcal{M}_j$, while $|\mathcal{M}_i|$ is the memory buffer size at $\mathcal{T}_i$. $\mathcal{L}_{VAE}(\cdot,\cdot)$ is the VAE loss [26], and $\mathcal{L}_{ce}(\cdot)$ is the cross-entropy loss, defined as :

$$\mathcal{L}_{ce}(\mathbf{y}, \hat{\mathbf{y}}) := -\sum_{i=1}^{K} y_i \log \hat{y}_i\,, \tag{27}$$

where y_i is the i-th entry and K is the dimension of $\mathbf{y}$. $\hat{\mathbf{y}}$ is the classification probability vector provided by the classifier. We implement the encoding distribution $p_{\theta_j}(\mathbf{x}\,|\,\mathbf{z})$ and the decoding one $q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x})$ using the neural networks enc_{ω_j} and dec_{θ_j}, of parameters ω_j and θ_j, respectively. The latent variable $\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\tau}$ is drawn from $q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}\mathbf{I})$ using the reparameterization trick to enable end-to-end training [26], where $\boldsymbol{\tau} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and where $\odot$ represents the dot product. $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are the

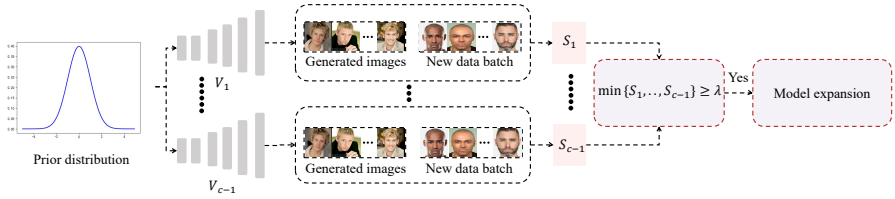


Figure 2: The process of the proposed dynamic model expansion procedure. For a new given data batch, we employ each expert to generate past examples. The evaluator is used to estimate the discrepancy distance between the generated images and real samples, which is used as the expansion signal. If the expansion criterion defined in Eq. (30) is satisfied, then we dynamically create a new expert.

mean and variance hyperparameters of the Gaussian distribution, given by the encoder enc_{ω_j}. A component is selected at the testing phase using Eq. (19), where $\hat{f}(\cdot,\cdot)$ is implemented by $-\mathcal{L}_{VAE}(\cdot,\cdot)$.

5.2. Mixture Model Expansion by means of the discrepancy distance

From Lemma 2, we find that encouraging data representation diversity in the components' representation abilities can lead to a good generalization performance while ensuring a compact architecture. The dynamic expansion mechanism adds new components to $\mathcal{G}$ during the training, by evaluating the following criterion :

$$\min_{t=1,\cdots,c} \widehat{\mathcal{L}}(\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_{k_t}}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}) \geq \lambda\,, \tag{28}$$

where λ is a mixture expansion threshold. If the distribution of the current memory buffer $\mathrm{P}_{\mathcal{M}_i}$ is sufficiently different from the memory distribution modeled by each other component, satisfying Eq. (28), then $\mathcal{G}$ is augmented with a new component for learning the information associated with the current memory $\mathcal{M}_i$. This expansion mechanism, illustrated in Fig. 2, encourages the learning diversity among the mixture model components.

Training the proposed ODDL involves three stages. In the beginning we build the first component h_1 in the mixture model $\mathcal{G}$, consisting of a VAE model V_1 and a classifier h_1. We then train G_1 on the memory buffer when $\mathcal{M}_j$ becomes full at the j-th training time ($\mathcal{T}_j$). Afterwards, the first component is frozen to preserve the previous knowledge and we employ a second component G_2 which is used for learning from

the data samples from $\mathcal{M}_i$ during the subsequent training steps. We also consider a component as an evaluator, $G_e = \{h_e, v_e\}$, initially considering $G_e \equiv G_2$, for evaluating the incoming data. The model expansion is decided using Eq. (28) at the i-th training time ($\mathcal{T}_i$).

However, data corresponding to $\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_{k_1}}$ cannot be accessed because the content of the memory buffer $\mathcal{M}_i$ was changed. We then form $\mathrm{P}^{\mathcal{X}}_{\mathcal{M}_{k_1}}$ by employing the auxiliary distributions $\mathrm{P}^{\mathcal{X}}_{v_1^{k_1=j}}$ consisting of samples drawn from G_1^j, using its VAE V_1^j, where the subscript $j = k_1$ indicates the component ending its training at $\mathcal{T}_j$. Consequently, we estimate the discrepancy distance $\mathcal{L}^\star(\cdot,\cdot)$, between the current evaluator h_e^i and the classifier of the first component h_1^j :

$$\mathcal{L}^\star(\mathrm{P}^{\mathcal{X}}_{v_1^j}, \mathrm{P}_{\mathcal{M}_i}) := \left|\mathbb{E}_{\mathbf{x}\sim\mathrm{P}^{\mathcal{X}}_{v_1^j}}[\mathcal{L}(h_1^j(\mathbf{x}), h_e^i(\mathbf{x}))] - \mathbb{E}_{\mathbf{x}\sim\mathrm{P}_{\mathcal{M}_i}}[\mathcal{L}(h_1^j(\mathbf{x}), h_e^i(\mathbf{x}))]\right|, \tag{29}$$

where $h_e^i(\cdot)$ is a classifier applied on the memory buffer at i-th training time.

When the condition from Eq. (28) is fulfilled, we freeze G_e^i and use it for expanding $\mathcal{G}$, while considering a new evaluator at $\mathcal{T}_{i+1}$; otherwise, we continually learn $G_e^i \rightarrow G_e^{i+1}$ on the memory buffer $\mathcal{M}_{i+1}$ during next training step, $\mathcal{T}_{i+1}$. In addition, after using the memory buffer $\mathcal{M}_i$ for training the new component, we replace its content with new data. This encourages the newly added component to learn novel information, ensuring the learning of diverse information by the model. Furthermore, when the model $\mathcal{G} = \{G_1, \cdots, G_{c-1}\}$ has already frozen more than one component, we extend the criterion from Eq. (28) to use the discrepancy distance from Eq. (29), at $\mathcal{T}_i$, as :

$$\min_{t=1,\cdots,c-1} \mathcal{L}^\star(\mathrm{P}^{\mathcal{X}}_{v_t^{k_t}}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}) \geq \lambda\,, \tag{30}$$

where $\mathrm{P}^{\mathcal{X}}_{v_t^{k_t}}$ is the distribution of the generated samples by all existing components, $G_t^{k_t}$. Unlike Eq. (28) which considers only one component, Eq. (30) enables the knowledge evaluation from all previously learnt components, leading to better expansion signals for the mixture model $\mathcal{G}$.

5.3. Model expansion by initializing the evaluator with an existing component

The negative backward transfer means that the model suffers from performance degeneration in previous data when learning new data samples. Such a negative effect happens when the new data samples share significantly different information with

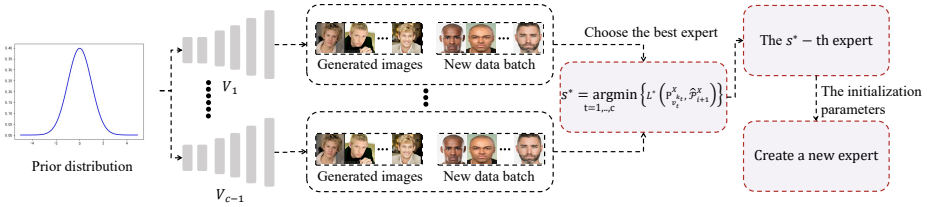


Figure 3: The process of the proposed expert initialization mechanism. The samples drawn from the prior distribution are used as the inputs of the decoder of each expert, which can then generate images. We choose the best expert $G_{s^\star}$ using Eq. (32) and employ the parameters of $G_{s^\star}$ to create a new expert.

respect to previous data samples. The positive knowledge transfer indicates that learning new data samples can improve the model's performance on previous data. Such a positive knowledge transfer occurs when the new data samples share similar semantic information with respect to previously learnt data.

In Section 5.2, we have introduced a novel dynamic expansion model to avoid the negative backward transfer. In this section, we aim to improve the positive backward transfer of the proposed framework, which can promote new task learning. Creating a new evaluator with randomly initialized parameters would slow the convergence during training. In the following we introduce a new expansion approach by initializing a new evaluator using the parameters from an existing component. The criterion for choosing an appropriate component consists of assessing the similarity between the information contained in the new data batch and that learnt by existing components. Then, the new evaluator initialized by the selected component's parameters undergoes a positive knowledge transfer when learning novel data. We define the knowledge transfer by means of the following Lemma.

Lemma 3. *Let $\widehat{\mathcal{P}}_{i+1}$ be the distribution of a new data batch $\{\mathbf{X}^b_{i+1}, \mathbf{Y}^b_{i+1}\}$ drawn from $\mathcal{A}$ at $\mathcal{T}_{i+1}$. Let $h_{\widetilde{\mathcal{P}}_{i+1}} = \arg\min_{h\in\mathcal{H}} \mathcal{L}_{\widehat{\mathcal{P}}_{i+1}}(h, f_{\widetilde{\mathcal{P}}_{i+1}})$ and $h_{\mathcal{M}_i} = \arg\min_{h\in\mathcal{H}} \mathcal{L}_{\mathrm{P}_{\mathcal{M}_i}}(h, f_{\mathrm{P}_{\mathcal{M}_i}})$ be the optimal classifiers for $\widehat{\mathcal{P}}_i$ and $\mathrm{P}_{\mathcal{M}_i}$, respectively. We derive the GB between $\widehat{\mathcal{P}}_{i+1}$ and $\mathrm{P}_{\mathcal{M}_i}$, according to Theorem 1, as :*

$$\mathcal{L}_{\widehat{\mathcal{P}}_{i+1}}(h, f_{\widetilde{\mathcal{P}}_{i+1}}) \leq \mathcal{L}_{\mathrm{P}_{\mathcal{M}_i}}(h, h_{\mathcal{M}_i}) + \widehat{\mathcal{L}}(\widehat{\mathcal{P}}^{\mathcal{X}}_{i+1}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}) + \eta(\widehat{\mathcal{P}}_{i+1}, \mathrm{P}_{\mathcal{M}_i}) \,. \tag{31}$$

However, different from Theorem 1, which mainly provides the forgetting analysis,

Eq. (31) is used to evaluate the knowledge transferability on a new data batch, achieved by a model trained at a previous training time ($\mathcal{T}_i$). We observe that a small discrepancy distance component from the RHS of Eq. (31) can improve the knowledge transferability of the model. Inspired by this analysis, we propose a new expansion strategy that chooses an appropriate component to initialize the evaluator according to the following criterion at a new training step ($\mathcal{T}_{i+1}$) :

$$s^\star = \arg\min_{t=1,\cdots,c} \left\{ \mathcal{L}^\star(\mathrm{P}^{\mathcal{X}}_{\nu_t^{k_t}}, \widehat{\mathcal{P}}^{\mathcal{X}}_{i+1}) \right\}, \tag{32}$$

where c represents the number of components from $\mathcal{G}$ and $s^\star$ is the index of the selected component used to initialize a new evaluator when the model $\mathcal{G}$ checks whether to expand its architecture. We also provide the detailed process of the proposed expert initialization mechanism in Fig. 3. Eq. (32) enables a newly created evaluator to reuse existing knowledge, aiming to accelerate future learning and this approach is called ODDL-S2.

5.4. Ensuring sample diversity in the memory buffer $\mathcal{M}_i$

According to the results from Theorem 4, the probabilistic diversity of the knowledge learnt by the model plays an important role in its performance. Thus we propose a criterion for selecting data for the memory buffer $\mathcal{M}_i$, further ensuring the learning of diverse information. We consider the following discrepancy distance for selecting data for the memory buffer $\mathcal{M}_i$, such that it is different from the data currently known to the model's components :

$$\mathcal{L}(\mathbf{x}, \mathcal{G}) := \frac{1}{c-1} \sum\nolimits_{t=1}^{c-1} \left| \mathcal{L}(h_t^{k_t}(\mathbf{x}), h_e^i(\mathbf{x})) - \mathcal{L}(h_t^{k_t}(\mathbf{x}_t^{k_t}), h_e^i(\mathbf{x}_t^{k_t})) \right|, \tag{33}$$

where $\mathbf{x}$ is a sample obtained from $\mathcal{M}_i$, $h_e^i(\cdot)$ is a classifier, while $\mathbf{x}_t^{k_t}$ is a generated sample drawn from $G_t^{k_t}$, using its VAE V_{k_t}, c represents the number of components for $\mathcal{G}$. $h_t^{k_t}(\cdot)$ is the classifier associated to the component $G_t^{k_t}$ from $\mathcal{G}$. Eq. (33) calculates the average discrepancy distance between each memorized sample and the data generated by each trained component, which is used to guide the sample selection at the time ($\mathcal{T}_i$) based on its novelty to the model. Specifically, we employ Eq. (33) to estimate the selection score for each memory buffer, resulting in :

$$\left\{ s_1, s_2, \cdots, s_{|\mathcal{M}^i|} \right\} = F_{\text{sort}}\Big(\Big\{ \mathcal{L}(\mathcal{M}_i[1], \mathcal{G}), \mathcal{L}(\mathcal{M}_i[2], \mathcal{G}), \cdots, \mathcal{L}(\mathcal{M}_i[|\mathcal{M}_i|], \mathcal{G}) \Big\} \Big), \tag{34}$$

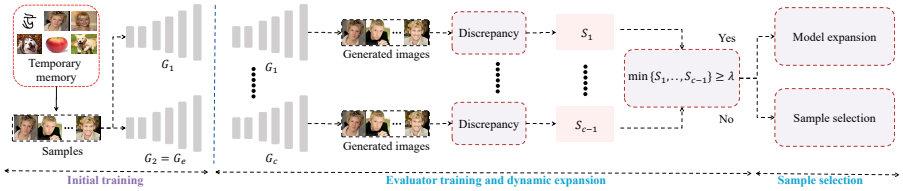


Figure 4: The learning process of the proposed Online Discrepancy Distance Learning with sample selection (ODDL-S), consisting of three phases. In the first learning stage, we update the components G_1 and G_e to learn initial knowledge. Then we only consider updating the current component G_e if the memory is full. The proposed expansion criterion is performed to check the model expansion using Eq. (28). If the criterion is met, we add a new component to the mixture system and clear up the current memory buffer, otherwise, we select data samples for the memory buffer using Eq. (33).

where $\mathcal{M}_i[j]$ denotes the j-th sample from $\mathcal{M}_i$ and F_{sort} is a function that sorts incoming samples by the selection score, based on the consistency of the new sample with the existing knowledge learnt by the j-th component, from the most to the least knowledgeable component module of the sample, and returns the sample index of the sorted samples. $s_1, s_2, \cdots, s_{|\mathcal{M}_i|}$ denotes a set of indexes of all sorted samples By using Eq. (34), we update the memory buffer $\mathcal{M}_i$ by :

$$\mathcal{M}_i = \left\{\mathcal{M}_i[s_1], \cdots, \mathcal{M}_i[s_{\mathcal{M}^{max}}]\right\}, \tag{35}$$

where $\mathcal{M}^{max}$ denotes the maximum number of samples for the memory buffer. We call this approach as ODDL with sample selection (ODDL-S).

5.5. Algorithm Implementation

The implementation for OODL is provided in Algorithm 1 and the learning process is presented in Fig. 4. This consists of three stages at each training time $\mathcal{T}_i$:

- **(Initial training).** At the start of the training, we build two components $\{G_1, G_2\}$, with one of them used as the evaluator $G_e = G_2$. We train G_1 on the memory buffer using Eq. (25) and Eq. (26). When the memory buffer $\mathcal{M}_j$ size becomes equal to $|\mathcal{M}|^{max}$, we freeze G_1 to preserve the previously learnt knowledge and consider another identical component $G_2 \equiv G_1$. The dynamic expansion mechanism is performed during subsequent learning according to Eq. (28), which requires the knowledge from at least an existing frozen component.

Algorithm 1: The supervised learning of ODDL-S

Input: The data stream $\mathcal{A}$, total number of tasks n, number of components k;
for $i < n$ **do**
  At task $\mathcal{T}_i$, get the data batch $\{\mathbf{X}_i^b, \mathbf{Y}_i^b\}$ from $\mathcal{A}$;
  Initial training stage;
  if (k == 2) **then**
    if ($|\mathcal{M}_i| < \mathcal{M}^{Max}$) **then**
      Update the parameters of the classifiers of G_1 and G_2 on $\mathcal{M}_i$ using Eq. (25) ;
      Update the parameters of the VAEs of G_1 and G_2 on $\mathcal{M}_i$ using Eq. (26);
    end if
  else
    Evaluator training stage;
    if ($|\mathcal{M}_i| < \mathcal{M}^{Max}$) **then**
      Update the parameters of the classifier of G_e on $\mathcal{M}_i$ using Eq. (25);
      Update the parameters of the VAE of G_e on $\mathcal{M}_i$ using Eq. (26);
    else
      Estimate the discrepancy distance using Eq. (29));
      if $\min_{t=1,\cdots,c-1} \left\{\mathcal{L}^{\star}(\mathrm{P}^{\mathcal{X}}_{\mathcal{V}_t^k}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i})\right\} \geq \lambda$ **then**
        Add the component $\mathcal{G} = G_e \cup \mathcal{G}$;
        Build a new component $G_e = G_{k+1}$;
        Update the number of components $k = k + 1$;
      else
        Sample selection stage;
        $\left\{s_1, s_2, \cdots, s_{|\mathcal{M}_i|}\right\} = F_{\text{sort}}\Big(\Big\{\mathcal{L}(\mathcal{M}_i[1], \mathcal{G}), \mathcal{L}(\mathcal{M}_i[2], \mathcal{G}), \cdots, \mathcal{L}(\mathcal{M}_i[|\mathcal{M}_i|], \mathcal{G})\Big\}\Big)$;
        $\mathcal{M}_i = \left\{\mathcal{M}_i[s_1], \cdots, \mathcal{M}_i[s_{\mathcal{M}^{max}}]\right\}$;
      end if
    end if
  end if
end for

- **(Evaluator training).** At a certain training time $\mathcal{T}_{i+1}$, we use Eq. (25) and Eq. (26) for training the Evaluator G_e on the memory buffer. If this is full $|\mathcal{M}_{i+1}| \geq |\mathcal{M}|^{max}$, then we assess the need for expansion through Eq. (30), by evaluating the discrepancy distance using Eq. (29). If model expansion is indicated then we freeze G_e and add it to $\mathcal{G}$ while building a new evaluator G_e, initialized according to Eq. (32), while clearing up the memory $\mathcal{M}_{i+1}$ to avoid learning similar information during subsequent training; otherwise, the selection of new samples is undertaken.
- **(Sample selection).** We assess the discrepancy distance between the data generated by the VAEs of previously learnt components and the memorized samples from the

memory buffer $\mathcal{M}_j$ using Eq. (33). Sample selection is then undertaken for $\mathcal{M}_{i+1}$. The algorithm then proceeds with the second stage, Evaluator training, from above.

5.6. Extending ODDL to the Unsupervised Learning

In this section, we extend the proposed ODDL to unsupervised learning by introducing a novel dynamic expansion mechanism, which employs the VAE model to estimate the discrepancy, expressed as :

$$\mathcal{L}_U^{\star}(\mathrm{P}^{\mathcal{X}}_{v_t^{k_t}}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}) := \Big|\mathbb{E}_{\mathbf{x}\sim \mathrm{P}^{\mathcal{X}}_{v_1^j}}[\mathcal{L}_R(\widehat{\mathbf{x}}_t, \widehat{\mathbf{x}}_e))] - \mathbb{E}_{\mathbf{x}\sim \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i}}[\mathcal{L}_R(\widehat{\mathbf{x}}_t, \widehat{\mathbf{x}}_e)]\Big|, \tag{36}$$

where $\widehat{\mathbf{x}}_t$ and $\widehat{\mathbf{x}}_e$ denote the image reconstruction of $\mathbf{x}$ given by the VAE of the t-th component and by that of the evaluator, respectively. Unlike Eq. (29), which requires to train a classifier for each component, Eq. (36) can estimate the discrepancy distance using the VAE component and thus can be used for unsupervised learning. Based on the discrepancy distance, defined in Eq. (36), we introduce a new unsupervised model expansion criterion as :

$$\min_{t=1,\cdots,c-1}\left\{\mathcal{L}_U^{\star}(\mathrm{P}^{\mathcal{X}}_{v_t^{k_t}}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i})\right\} \geq \lambda_U\,, \tag{37}$$

where λ_U is an expansion threshold controlling the unsupervised model expansion process. We call this unsupervised learning model variant as ODDL unsupervised (ODDL-U).

Unlike in the supervised learning, we design each component of the proposed ODDL-U framework adapted for the unsupervised learning using the Variational Autoencoder (VAE). Each component G_j has an inference model $q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x})$ as well as a decoder $p_{\theta_j}(\mathbf{x}\,|\,\mathbf{z})$, where ω_j and θ_j denote the corresponding parameter sets. If G_j is the current component, we only update the parameters of G_j on the memory buffer using Eq. (26). We provide the pseudocode of the unsupervised learning algorithm of the proposed ODDL-U framework in **Algorithm 2**, which consists of two steps at each training time :

- **(Initial training).** At the initial stage of the training process, we build two components $\{G_1, G_2\}$ and take one of them as the evaluator $G_e = G_2$. We update G_1 on

Algorithm 2: The unsupervised learning of ODDL-U

Input: The data stream $\mathcal{A}$, The total number of iterations n, The number of components k;
for $i < n$ **do**
  $\{\mathbf{X}_i^b\}$ Get the data batch from A ;
  Initial training stage;
  if (k == 2) **then**
    if ($|\mathcal{M}_i| < \mathcal{M}^{Max}$) **then**
      Update the parameters of the VAEs of G_1 and G_2 on $\mathcal{M}_i$ using Eq. (26);
    end if
  else
    Evaluator training stage;
    if ($|\mathcal{M}_i| < \mathcal{M}^{Max}$) **then**
      Update the parameters of the VAE of G_e on $\mathcal{M}_i$ using Eq. (26);
    else
      Estimate the discrepancy distance using Eq. (36));
      if $\min_{t=1,\cdots,c-1} \left\{\mathcal{L}^{\star}(\mathrm{P}^{\mathcal{X}}_{\substack{k_t \\ v_t}}, \mathrm{P}^{\mathcal{X}}_{\mathcal{M}_i})\right\} \geq \lambda$ **then**
        Add the component $\mathcal{G} = G_e \cup \mathcal{G}$;
        Build a new component $G_e = G_{k+1}$;
        Update the number of components $k = k + 1$;
      end if
    end if
  end if
end for

the memory buffer using Eq. (25) and Eq. (26). When the memory buffer is full $|\mathcal{M}_j| = |\mathcal{M}|^{max}$, we freeze G_1 to preserve the previously learnt knowledge and continually update another component G_2 in the subsequent learning. The dynamic expansion mechanism in the unsupervised learning is performed by using the expansion criterion from Eq. (37), which requires the knowledge from at least an existing frozen component.

- **(Evaluator training).** At a certain training/learning time $\mathcal{T}_{i+1}$, we employ Eq. (26) for updating the Evaluator on the memory buffer. If the memory buffer is full $|\mathcal{M}_{i+1}| = |\mathcal{M}|^{max}$, then we assess the need for expansion through Eq. (37), by evaluating the discrepancy distance using Eq. (36). If model expansion is indicated then we freeze G_e and add it to $\mathcal{G}$ while building a new evaluator G_e, initialized according to Eq. (32), while clearing up the memory $\mathcal{M}_{i+1}$ to avoid learning similar information during subsequent training; otherwise, the selection of new samples is undertaken.

6. Experiments

After introducing the experimental setting, we provide the results when applying the proposed model in various Task Free Continual Learning (TFCL) applications. Results analyzing the forgetting process of the model are assessed and discussed.

6.1. Experimental Setting

Datasets. We consider adapting the standard TFCL benchmark from [16], which employs several datasets for evaluating the model's performance, including Split CIFAR100 [28], Split MNIST [31, 54] and Split CIFAR10 [28, 54]. For Split MNIST, we divide 60k training data samples into 5 parts based on the category information [16]. We do similar splits for CIFAR10 and CIFAR100, with the resulting datasets named Split CIFAR10 and Split CIFAR100, respectively. Split CIFAR10 and Split CIFAR100 contain 5 and 20 subsets, respectively.

Hyperparameters and GPU configuration. The Adam algorithm is used for all models where we consider the learning rate of 0.0001. We use a GeForce GTX 1080 for the experiments while the operating system is Ubuntu 18.04.5.

The network architecture. According to the experiment setting from [16], we use ResNet-18 [22] as the backbone of the classifier for Split CIFAR100 and Split CIFAR10. For the Split MNIST, we consider an MLP network with two hidden layers of 400 units [16], which is used as the backbone of the classifier. For Split CIFAR100, Split MNIST and Split CIFAR10, the maximum memory buffer size $|\mathcal{M}|^{max}$ is set as 5000, 2000 and 1000, respectively. When the model is updated at each training session, we can only access a small data batch ($b = 10$ data samples). We empirically search $\lambda \in [0.1, 4.0]$. The final λ values for Split CIFAR100, Split CIFAR10 and Split MNIST are of 3.0, 0.45 and 0.2, respectively.

Performance criterion. We adopt the average classification accuracy, calculated on the testing dataset, as the performance criterion, which has also been widely used in other TFCL benchmarks, [3]. Specifically, let $\{\mathcal{T}_1, \cdots, \mathcal{T}_n\}$ denote a series of n tasks. When the model finishes learning all tasks, it can then be evaluated on all testing datasets

Table 2: Classification accuracy of various continual learning models, when considering the average for five independent runs.

Methods	**Split MNIST**	**Split CIFAR10**	**Split CIFAR100**
GEM* [38]	93.25 ± 0.36	24.13 ± 2.46	11.12 ± 2.48
iCARL* [46]	83.95 ± 0.21	37.32 ± 2.66	10.80 ± 0.37
finetune*	19.75 ± 0.05	18.55 ± 0.34	3.53 ± 0.04
MIR* [1]	93.20 ± 0.36	42.80 ± 2.22	20.00 ± 0.57
reservoir* [51]	92.16 ± 0.75	42.48 ± 3.04	19.57 ± 1.79
CoPE-CE* [16]	91.77 ± 0.87	39.73 ± 2.26	18.33 ± 1.52
GSS* [4]	92.47 ± 0.92	38.45 ± 1.41	13.10 ± 0.94
CoPE* [16]	93.94 ± 0.20	48.92 ± 1.32	21.62 ± 0.69
ER + GMED†[23]	82.67 ± 1.90	34.84 ± 2.20	20.93 ± 1.60
ER_a + GMED†[23]	82.21 ± 2.90	47.47 ± 3.20	19.60 ± 1.50
CURL* [45]	92.59 ± 0.66	-	-
Dynamic-OCM [54]	94.02 ± 0.23	49.16 ± 1.52	21.79 ± 0.68
CNDPM* [33]	93.23 ± 0.09	45.21 ± 0.18	20.10 ± 0.12
ODDL-S2	**96.15** ± 0.06	**53.68** ± 0.13	**29.62** ± 0.64
ODDL-S	95.75 ± 0.05	52.69 ± 0.11	27.21 ± 0.87
ODDL	94.85 ± 0.02	51.48 ± 0.12	26.20 ± 0.72

$\{\mathcal{D}_1^T, \cdots, \mathcal{D}_n^T\}$ while the average accuracy is calculated by :

$$a = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|\mathcal{D}_i^T|}\sum_{j=1}^{|\mathcal{D}_i^T|} F_{\text{acc}}(\mathbf{y}'_{i,j}, \mathbf{y}_{i,j}), \tag{38}$$

where $|\mathcal{D}_i^T|$ denotes the total number of data samples for $\mathcal{D}_i^T$. $\mathbf{y}'_{i,j}$ and $\mathbf{y}_{i,j}$ denote the j-th class label from $\mathcal{D}_i^T$ and the prediction on the data sample $\mathbf{x}_{i,j}$. $F_{\text{acc}}(\cdot,\cdot)$ denotes a function that returns 1 if $\mathbf{y}'_{i,j} = \mathbf{y}_{i,j}$, otherwise, returns 0.

6.2. Results on the TFCL Benchmark

In this section, we perform several TFCL experiments to evaluate the performance of the proposed Online Discrepancy Distance Learning with sample selection (ODDL-S). The results for Split MNIST, Split CIFAR10 and Split CIFAR100 are provided in Table 2, where * and † denote the results cited from [16] and [23], respectively. We compare the proposed ODDL-S with several baselines, including: finetune that directly trains a classifier on the data stream, GSS [4], Dynamic-OCM [54], MIR [1], Gradient Episodic Memory (GEM) [38], Incremental Classifier and Representation Learning

Table 3: Average classification accuracy, calculated for twenty runs, on testing samples from Split MImageNet and Permuted MNIST.

Methods	**Split MImageNet**	**Permuted MNIST**
MIR [1]	25.21 ± 2.2	79.13 ± 0.7
MIR+GMED [23]	26.50 ± 1.3	79.25 ± 0.8
ER + GMED [23]	27.27 ± 1.8	78.86 ± 0.7
CNDPM [33]	27.12 ±1.5	80.68 ± 0.7
ER_a [48]	25.92 ± 1.2	78.11 ± 0.7
ODDL	27.45 ± 0.9	82.33 ± 0.6
ODDL-S	28.68 ± 1.5	83.56 ± 0.5
ODDL-S2	**30.03** ± 1.4	**85.26** ± 0.5

(iCARL) [46], Reservoir [51], CURL [45], CNDPM [33], CoPE [16], ER + GMED and ER_a + GMED [23] where ER is the Experience Replay [48] and ER_a is ER with data augmentation. The final mixture model size created by ODDL-S (and ODDL) for Split MNIST, Split CIFAR10 and Split CIFAR100 consists of 7, 9, 7, components, respectively. From the results of Table 2, we can observe that the proposed ODDL-S achieves the best results when compared to all other models. In addition, we also train ODDL considering Eq. (32) for the model expansion, which creates a new component using the parameters from a selected existing component, and we call this model as ODDL-S2. The results from Table 2 indicate that by using existing parameters to initialize a new component can further improve the performance of the proposed methodology.

In the following, we investigate the effectiveness of the proposed ODDL-S on the large-scale dataset MINI-ImageNet [30], and Permuted MNIST [21]. We divide MINI-ImageNet into 20 separate parts, where each part contains samples from five classes [1], forming Split MImageNet. Permuted MNIST consists of 10 parts, with each consisting of images resulting from applying random permutations in the pixels of all images from the database [21]. We adopt the setting from [1] to train the model, where the maximum memory size is $10,000$, while ResNet-18 [22] is used as the classifier. The hyperparameter from Eq. (30), controlling the model expansion, is set as $\lambda = 1.2$ for Split MINI-ImageNet and $\lambda = 1.5$ for Permuted MNIST. The classification results for Split MINI-ImageNet and Permuted MNIST are provided in Table 3, where we compare the proposed ODDL-S with several state-of-the-art methods, where the results

Table 4: Classification accuracy, representing the average results for five independent runs, when considering data contamination, by mixing data from different classes.

Methods	Split MNIST	Split CIFAR10	Split MImageNet
ER [48]	79.74 ± 4.0	37.15 ± 1.6	26.47 ± 2.3
Vanilla	21.53 ± 0.1	20.69 ± 2.4	3.05 ± 0.6
MIR+GMED [23]	86.17 ± 1.7	41.22 ± 1.1	26.86 ± 0.7
MIR [23]	84.80 ± 1.9	38.70 ± 1.7	25.83 ± 1.5
ER + GMED [23]	82.73 ± 2.6	40.57 ± 1.7	28.20 ± 0.6
ODDL	94.25 ± 0.9	50.07 ± 1.2	27.98 ± 1.3
ODDL-S	**95.55** ± 1.2	**52.27** ± 2.5	**29.76** ± 1.6

of the other baselines are taken from [23]. From these results, ODDL-S2 outperforms the other baselines when learning these complex datasets.

6.3. Robustness to task data contamination

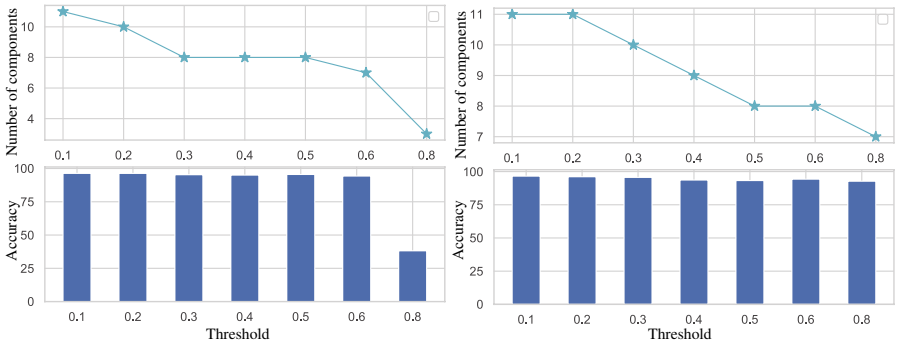
We evaluate the robustness of the proposed approach when considering data contamination as in [33]. Fuzzy task boundaries are created through data contamination by exchanging a certain number of samples between two tasks. These samples become outliers in the given data representations. Under this setting, data samples from the next task are introduced and mixed with samples from the current task after learning half of the current task's data, [33]. Specifically, for Split MNIST, the first task has examples from classes '0', '1', '2', '3' where the number of samples for the classes '0' and '1' is of 5000 while the number of examples for the classes '2' and '3' are of 2500 for each. The second task has examples from the classes '2', '3', '4', '5' in which the number of examples for each category is 2500. The third and fourth tasks use the same sample split procedure and the last task has only samples from the classes '8' and '9'. For Split MImageNet and Split CIFAR10 in the fuzzy task boundary setting, we employ the same split procedure. We report the results of various models under these training circumstances in Table 4. These results demonstrate that the proposed ODDL-S outperforms the other baselines by a large margin on fuzzy task boundaries, showing its robustness to data contamination and outliers.

Table 5: The log-likelihood estimation results, evaluated on the testing data by using the Importance Weighted Variational Autoencoder (IWVAE) [8] bound considering 1000 importance samples. The results for other methods are taken from [59] and N represents the number of components for each dynamic expansion model. 'Memory' denotes the number of memorized samples when the training is finished, while 'NLog' is the negative sample log-likelihood.

	Split MNIST			**Split Fashion**			**Split MNIST-Fashion**			**Cross-domain**		
Methods	NLog ↑	Memory	N	NLog ↑	Memory	N	NLog ↑	Memory	N	NLog ↑	Memory	N
VAE-ELBO-MIR [1]	-143.27	3.0K	1	-274.72	3.0K	1	-238.68	3.0K	1	-237.93	3.0K	1
VAE-reservoir [54]	-144.17	3.0K	1	-276.60	3.0K	1	-240.02	3.0K	1	-239.42	3.0K	1
LIMix [53]	-146.23	2.0K	30	-262.52	2.0K	30	-238.63	2.0K	30	-226.63	2.0K	30
VAE-ELBO-Random	-150.79	3.0K	1	-280.54	3.0K	1	-247.46	3.0K	1	-239.71	3.0K	1
VAE-ELBO-OCM [54]	-132.07	1.6K	1	-250.74	1.6K	1	-215.62	2.0K	1	-201.31	2.0K	1
VAE-IWVAE50-OCM [54]	-127.11	1.6K	1	-247.90	1.6K	1	-224.34	2.0K	1	-204.35	2.0K	1
CNDPM [33]	-120.71	2.0K	30	-257.56	2.0K	30	-236.79	2.0K	30	-218.15	2.0K	30
Dynamic-ELBO-OCM [54]	-115.89	1.6K	5	-237.69	1.8K	10	-187.49	1.9K	10	-177.29	2.0K	11
OAES-ELBO [59]	-103.93	1.5K	5	-231.10	1.5K	10	-171.62	1.9K	8	-165.29	2.0K	11
ODDL	**-102.16**	1.5K	5	**-229.42**	1.5K	10	**-170.13**	1.9K	8	**-164.26**	2.0K	11

6.4. Unsupervised Learning

In this section, we evaluate the performance of various models considering the density estimation evaluation, as it was used in the lifelong generative modeling [60]. We train different models on various datasets, including Split MNIST, Split Fashion, Split MNIST-Fashion and Cross-domain, where Split MNIST-Fashion consists of data samples from Split MNIST and Split Fashion, while Cross-domain includes data samples from MNIST, Fashion and Omniglot [29]. The results for the unsupervised learning are evaluated on the testing data by considering the log-likelihood, evaluated using the Importance Weighted Autoencoder (IWELBO) [8], which provides a better estimation than the original VAE model for density estimation. The density estimation results for various models are reported in Table 5, where the VAE-reservoir is a baseline that is adopted in [54] and trains a VAE model using the reservoir sampling [51]. Single models such as VAE-reservoir and VAE-ELBO-MIR usually achieve small sample log-likelihood results when compared with the dynamic expansion models. The reason for these results is that a single model is unable to capture all categories due its memory restriction. In contrast, the dynamic expansion models can freeze the network



(a) Changing λ in ODDL. (b) Changing λ in ODDL-S.

Figure 5: Classification performance (bottom bar-plots) when varying the threshold λ in Eq, (30) on the Split MNIST dataset for adding new components to the mixture, which are indicated in top plots under TFCL.

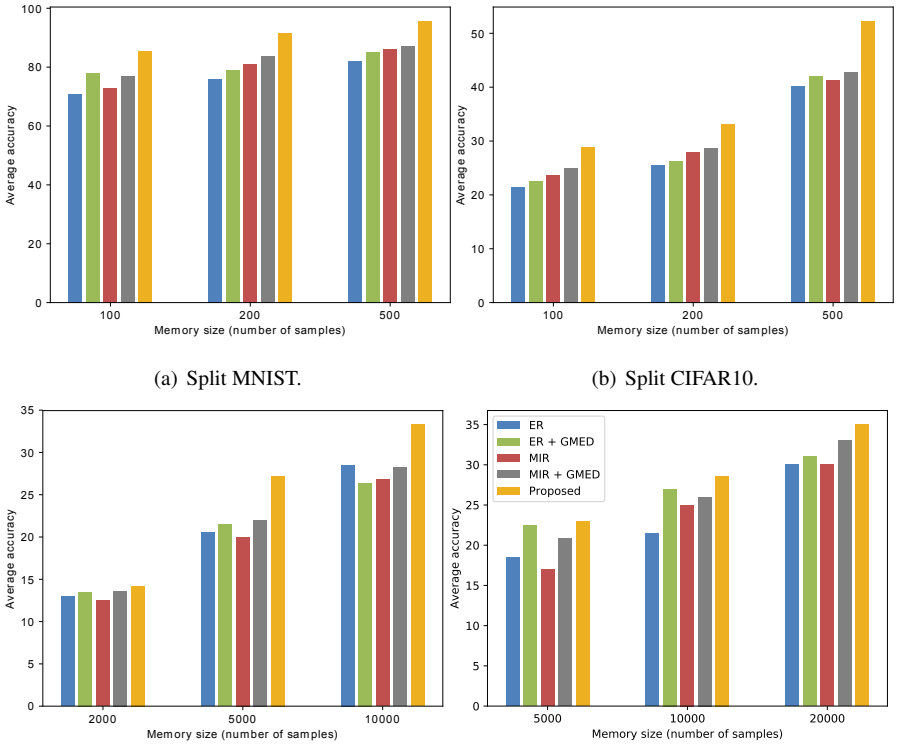
parameters to preserve previously learned information while dynamically creating new experts to learn the new information, thus being able to capture more categories. By using the IWELBO objective function, the model can further improve its performance on unsupervised learning, demonstrated by the results of VAE-IWVAE50-OCM in Table 5. Furthermore, the proposed approach achieves the best performance on all datasets while requiring equal or fewer parameters and memorized samples than other baselines.

6.5. Ablation Study

In the following we perform ablation studies to investigate the selection of the parameters and the importance of each module in the proposed method.

The effect of changing the threshold λ in Eq. (30)**.** The proposed dynamic model expansion mechanism employs an expansion threshold to regulate the model expansion process. Given an appropriate expansion threshold, our dynamic expansion framework adds new experts when the incoming samples contain sufficient novel information with respect to the entire previously learned knowledge.

We investigate the proposed mixture expansion approach when varying λ, and the results on the Split MNIST dataset are reported in Figures 5-a and 5-b for ODDL and ODDL-S, respectively. A small threshold λ leads to creating more components in ODDL or ODDL-S, while also slightly improving the performance. In contrast, a large λ discourages the expansion in the number of mixture components, leading



(a) Split MNIST.

(b) Split CIFAR10.

(c) Split CIFAR100.

(d) Split MINI-ImageNet.

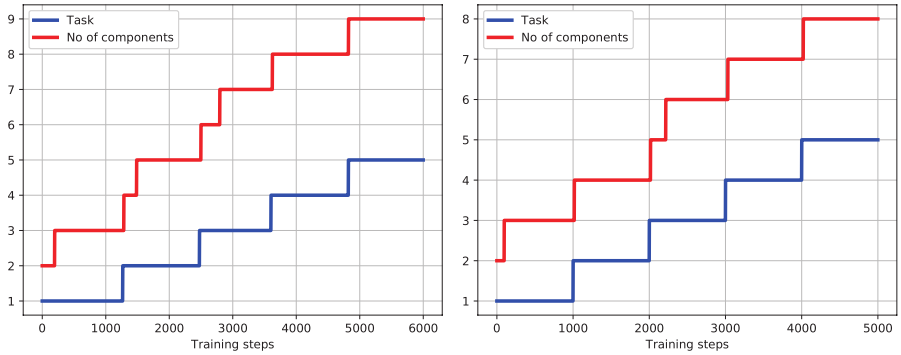
Figure 6: The performance of various models on various continual learning tassk when changing the memory size, which are indicated on the x axes.

to a more compact architecture. We observe that ODDL has for $\lambda = 0.8$ only three components, but suffers enormous degenerated performance, while ODDL-S, which uses a sample selection mechanism from the memory buffer $\mathcal{M}_i$ does not lose that much performance. We empirically search $\lambda \in [0.1, 4.0]$, and select λ as 0.2, 0.45 and 3.0 for the continual learning using ODDL-S of Split MNIST, Split CIFAR10 and Split CIFAR100, respectively.

The effect of the memory size $|\mathcal{M}_i|$. We investigate the performance change of various models when varying the memory size. We consider the number of samples as 100, 200 and 500 in $\mathcal{M}_i$, for Split MNIST and Split CIFAR10, and the results are presented in Fig. 6-a and Fig. 6-b. We also consider memory sizes of 2000, 5000, and 10000 samples for Split CIFAR100, and of 5000, 10000, and 20000 samples for Split MINI-ImageNet. We report the results in Fig. 6-c and Fig. 6-d. The proposed ODDL achieves

Table 6: Classification accuracy when considering the sample selection for the memory buffer $\mathcal{M}_i$ (ODDL-S), as in Eq. (32) compared to not considering it (ODDL), or considering a random sample selection (ODDL-Random). The average results for five different runs are provided.

Methods	Split MNIST	Split CIFAR10	Split CIFAR100
ODDL-Random	95.12 ± 0.13	51.68 ± 0.18	20.23 ± 0.65
ODDL-S	**95.75** ± 0.05	**52.69** ± 0.11	**27.21** ± 0.87
ODDL	94.85 ± 0.02	51.48 ± 0.12	26.20 ± 0.72



(a) Split MNIST. (b) Split CIFAR10.

Figure 7: Detecting data distribution shifts, according to Eq. (30), when deciding whether to increase the number of components in ODDL-S.

the best results in each case and its performance improves when increasing the memory capacity $|\mathcal{M}_i|$. We also observe that even when the memory buffer is smaller, the proposed ODDL still outperforms other baselines by a significant margin.

The effect of the sample selection for the memory buffer $\mathcal{M}_i$. The results from Tables 2, 3, and 4 show the advantage of using the selection mechanism for the memory buffer $\mathcal{M}_i$ in ODDL-S when compared to not using sample selection, as in ODDL. In addition, we also observe that the sample selection preserves the performance even when considering a tiny memory buffer, according to the results from Fig. 5-b. In Table 6, we compare the results on Split MNIST, Split CIFAR10 and Split CIFAR100 for ODDL-S with ODDL and with ODDL-Random, where a random selection of samples is considered for $\mathcal{M}_i$. These results show that the proposed sample selection used in ODDL-S, described in Section 5.4, outperforms ODDL-Random.

Detecting the statistical shift in data. We study the dynamic expansion process of the

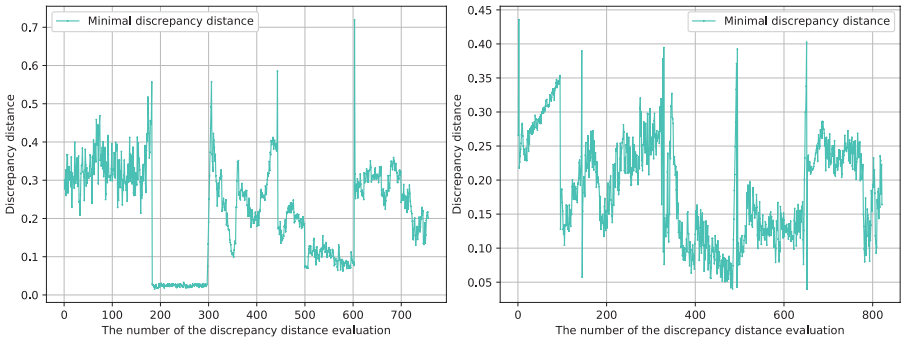
Table 7: The training time (minutes) for ODDL-S and CNDPM.

Methods	Split MNIST	Split CIFAR10	Split CIFAR100
ODDL-S	1.2	22.2	33.68
CNDPM	0.9	18.6	30.23

proposed ODDL-S by employing the discrepancy-based criterion from Eq. (30). At each training time, we record the number of data distributions (tasks) and the number of components used in the ODDL-S. We plot the number of components and tasks at each training step in Fig. 7-a, for the proposed ODDL-S on Split MNIST and in Fig. 7-b for Split CIFAR10, where 'Task' represents the number of given tasks, while the model can not access the task identification information during the training. We observe that the proposed discrepancy-based expansion criterion can accurately detect the data distribution shifts, enabling the proposed ODDL-S to expand the network architecture appropriately. Such an expansion mechanism ensures adding an optimal number of components without sacrificing performance, as discussed in Lemma 1.

Evaluation of the discrepancy distance. We train the proposed ODDL-S under Split MNIST and Split CIFAR10, where we estimate $\mathcal{L}(\mathbf{x}, \mathcal{G})$ for all samples $\mathbf{x} \sim \mathcal{M}_i$ using Eq. (33). We plot the results in Fig 8-a and Fig 8-b, for Split-MNIST and Split CIFAR10, respectively. These plots show that the minimal discrepancy distance is stable at certain training times and then suddenly becomes large, indicating the moment when the incoming samples originate from a different underlying data distribution. Such samples are novel for the probabilistic representation of the model $\mathcal{G}$ at that instance of the continual training process. When also considering the results from Fig. 7, the proposed evaluation of the discrepancy distance has been shown to be an effective approach in detecting data distribution shifts when continuously learning data streams by the proposed ODDL-S model.

The computational cost. When the memory buffer is not full, the computational costs of ODDL consist only from the training of the current expert because we do not perform the sample selection and dynamic expansion evaluation. In contrast, when the memory buffer is full, the sample selection and dynamic expansion evaluation are performed in each training step, leading to additional computational costs. In addition, increasing the



(a) Split MNIST. (b) Split CIFAR10.

Figure 8: The estimation of the minimal discrepancy distance, using Eq. (33), for checking incoming data's novelty.

number of components in the ODDL increases only slightly the computational costs of the sample selection and dynamic expansion evaluation. Therefore, when adding more components over time, the model would only slightly increase its computational costs. However, we can accelerate the proposed sample selection and dynamic expansion evaluation by recording the discrepancy score for the memorized samples and only calculate the discrepancy score once for each incoming sample.

In addition, all methods employ an identical count of training iterations in the experiments. Specifically, for memory-centric strategies such as GEM, iCARL, reservoir, GSS, ER + GMED, and ER$_a$ + GMED, the computational complexity of training iterations is expressed as $O(n\,\mathcal{M}^{max})$, where $\mathcal{M}^{max}$ represents the peak memory capacity and n signifies the maximum number of training iterations. In the context of the proposed framework and various dynamic expansion models, including CURL, Dynamic-OCM, and CNDPM, the upper bound of training iteration complexity aligns with other methods. Nevertheless, as numerous dynamic expansion techniques clear the memory buffer when dynamically building a new expert, the actual maximum training iteration complexity is smaller than $O(n\,\mathcal{M}^{max})$.

We report the training times for ODDL-S and CNDPM in Table 7. These results show that the proposed ODDL-S only requires slightly more computations when compared with CNDPM.

7. Conclusions

This research study provides a new theoretical framework for analysing forgetting behaviour in continual learning. In this framework we formulate the forgetting process as a measure of the generalization error, identified during the continual learning. We extend the findings from the domain adaptation theory to develop generalization bounds which describe the performance loss in the Task-Free Continual Learning (TFCL). The forgetting behaviour of a fixed and a dynamic expansion model is assessed at each training step. We theoretically demonstrate that the dynamic expansion model can achieve better performance than the static model when appropriately expanding its network architecture. Based on this analysis we propose the Online Discrepancy Distance Learning (ODDL) model, which uses a memory buffer for continually storing incoming data samples. ODDL estimates the discrepancy distance between the distribution of the samples temporarily stored in a memory buffer and the knowledge accumulated by each component from the mixture model. This discrepancy measure is then used as an expansion signal for deciding when ODDL is expanded with new components. In a further extension, we propose using a sample selection for the data from the memory buffer, ensuring that it stores a diversity of data samples during the continual learning, resulting in the ODDL with sample selection (ODDL-S) model, which further promotes the knowledge diversity among the components. The proposed models are shown to provide better results than others on several dataset configurations during TFCL experiments.

One limitation for this work is that if the components of the model are not properly decoupled, their learnt knowledge may interfere with each other, leading to a negative information transfer. However, the proposed dynamic model expansion mechanism, described in Section 5.2, learns diverse components if given an appropriate expansion threshold λ. Specifically, the proposed approach compares the discrepancy among components during the model expansion process, which ensures the learning of components representing complementary information to each other. The discrepancy measure from Eq. (30) and the expansion threshold λ ensures this. The only issue with this approach is that if the continual learning model is always presented with novel infor-

mation, it keeps creating new components which requires increasing resources. This may not be practical in some learning systems.

Some components may have learnt similar information and in this case we would aim to reduce the redundancy in the knowledge learnt by different components. This situation can be addressed by employing a component discarding mechanism, which removes the knowledge overlapping components. Specifically, we would first evaluate the knowledge distance between pairs of components. Then we would find a pair of components that have the smallest knowledge distance measure, indicating a strong similarity in their information representation. Then, we remove one of the components from the selected pair and consequently reduce the model size, while preserving the total information learnt by the multi-component model. Future research will also focus on extending the proposed methodology to the unsupervised generation task under TFCL when class information is missing.

References

[1] 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), 11872–11883, 2019.

[2] R. Aljundi, P. Chakravarty, T. Tuytelaars, Expert gate: Lifelong learning with a network of experts, in: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 3366–3375, 2017.

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

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

[5] A. Ashfahani, M. Pratama, Unsupervised Continual Learning in Streaming Environments, IEEE Transactions on Neural Networks and Learning Systems 34 (12) (2023) 9992–10003, doi:10.1109/TNNLS.2022.3163362.

[6] 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 Conf. on Computer Vision and Pattern Recognition (CVPR), 8218–8227, 2021.

[7] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira, Analysis of representations for domain adaptation, in: Advances in Neural Information Processing Systems (NIPS), 137–144, 2007.

[8] Y. Burda, R. Grosse, R. Salakhutdinov, Importance weighted autoencoders, arXiv preprint arXiv:1509.00519 .

[9] 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.

[10] H. Cha, J. Lee, J. Shin, Co2l: Contrastive continual learning, in: Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV), 9516–9525, 2021.

[11] A. Chaudhry, M. Ranzato, M. Rohrbach, M. Elhoseiny, Efficient lifelong learning with A-GEM, in: Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:1812.00420, 2019.

[12] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. Dokania, P. H. S. Torr, M. Ranzato, On Tiny Episodic Memories in Continual Learning, arXiv preprint arXiv:1902.10486 .

[13] R. Chen, X.-Y. Jing, F. Wu, H. Chen, Sharpness-aware gradient guidance for few-shot class-incremental learning, Knowledge-Based Systems (2024) 112030.

[14] C. Cortes, M. Mohri, Domain adaptation and sample bias correction theory and algorithm for regression, Theoretical Computer Science 519 (2014) 103–126.

[15] W. Dai, Q. Yang, G. R. Xue, Y. Yu, Boosting for transfer learning, in: Proc. Int Conf. on Machine Learning (ICML), 193–200, 2007.

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

[17] G. K. Dziugaite, D. M. Roy, Z. Ghahramani, Training generative neural networks via maximum mean discrepancy optimization, in: Proc. of Conf. on Uncertainty in Artificial Intell. (UAI), 258–267, 2015.

[18] S. Ebrahimi, F. Meier, R. Calandra, T. Darrell, M. Rohrbach, Adversarial continual learning, in: Proc. European Conference on Computer Vision (ECCV), vol. LNCS 12356, 386–402, 2020.

[19] Q. Gao, Z. Luo, D. Klabjan, F. Zhang, Efficient Architecture Search for Continual Learning, IEEE Transactions on Neural Networks and Learning Systems 34 (11) (2023) 8555–8565, doi:10.1109/TNNLS.2022.3151511.

[20] 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), 2672–2680, 2014.

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

[22] 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), 770–778, 2016.

[23] 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. 34, 29193–29205, 2021.

[24] H. Jung, J. Ju, M. Jung, J. Kim, Less-forgetting learning in deep neural networks, in: Proc. AAAI Conf. on Artificial Intelligence, vol. 32, 3358–3365, 2018.

[25] Z. Ke, B. Liu, N. Ma, H. Xu, L. Shu, Achieving Forgetting Prevention and Knowledge Transfer in Continual Learning, in: Advances in Neural Information Processing System, arXiv preprint arXiv:2112.02706, vol. 34, 2021.

[26] D. P. Kingma, M. Welling, Auto-encoding variational Bayes, arXiv preprint arXiv:1312.6114 .

[27] J. Knoblauch, H. Husain, T. Diethe, Optimal continual learning has perfect memory and is NP-hard, in: Proc. Int. Conf. on Machine Learning (ICML), vol PMLR 119, 5327–5337, 2020.

[28] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images, Tech. Rep., Univ. of Toronto, 2009.

[29] B. M. Lake, R. Salakhutdinov, J. B. Tenenbaum, Human-level concept learning through probabilistic program induction, Science 350 (6266) (2015) 1332–1338.

[30] Y. Le, X. Yang, Tiny imageNet visual recognition challenge, Tech. Rep., Univ. of Stanford, 2015.

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

[32] S. Lee, S. Goldt, A. Saxe, Continual Learning in the Teacher-Student Setup: Impact of Task Similarity, in: Proc. Int. Conf. on Machine Learning (ICML), vol. PMLR 139, 6109–6119, 2021.

[33] 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.

[34] C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, B. Póczos, MMD GAN: Towards deeper understanding of moment matching network, in: Advances in Neural Inf. Proc. Systems, 2203–2213, 2017.

[35] X. Li, S. Wang, J. Sun, Z. Xu, Variational Data-Free Knowledge Distillation for Continual Learning, IEEE Transactions on Pattern Analysis and Machine Intelligence (2023) 1–17.

[36] X. Li, W. Wang, GopGAN: Gradients Orthogonal Projection Generative Adversarial Network With Continual Learning, IEEE Transactions on Neural Networks and Learning Systems 34 (1) (2023) 215–227, doi: 10.1109/TNNLS.2021.3093319.

[37] Y. Li, K. Swersky, R. Zemel, Generative moment matching networks, in: Proc. Int. Conf. on Machine Learning (ICML), 1718–1727, 2015.

[38] D. Lopez-Paz, M. Ranzato, Gradient episodic memory for continual learning, in: Advances in Neural Information Processing Systems (NIPS), vol. 31, 6467–6476, 2017.

[39] Z. Mai, R. Li, J. Jeong, D. Quispe, H. Kim, S. Sanner, Online continual learning in image classification: An empirical survey, Neurocomputing 469 (2022) 28–51.

[40] Y. Mansour, M. Mohri, A. Rostamizadeh, Domain adaptation: Learning bounds and algorithms, in: Proc. Conference on Learning Theory (COLT), arXiv preprint arXiv:0902.3430, 2009.

[41] M. Mohri, A. M. Medina, New analysis and algorithm for learning with drifting distributions, in: Proc. Int. Conf. on Algorithmic Learning Theory, 124–138, 2012.

[42] 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.

[43] K. Raghavan, P. Balaprakash, Formalizing the Generalization-Forgetting Trade-off in Continual Learning, in: Advances in Neural Information Processing Systems (NeurIPS), vol. 34, 17284–17297, 2021.

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

[45] D. Rao, F. Visin, A. A. Rusu, Y. W. Teh, R. Pascanu, R. Hadsell, Continual Unsupervised Representation Learning, in: Advances in Neural Inf. Proc. Systems (NeurIPS), vol. 32, 7645–7655, 2019.

[46] 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), 2001–2010, 2017.

[47] B. Ren, H. Wang, J. Li, H. Gao, Life-long learning based on dynamic combination model, Applied Soft Computing 56 (2017) 398–404.

[48] D. Rolnick, A. Ahuja, J. Schwarz, T. P. Lillicrap, G. Wayne, Experience Replay for Continual Learning, in: Advances in Neural Information Processing Systems (NeurIPS), 348–358, 2019.

[49] T. G. Rudner, F. B. Smith, Q. Feng, Y. W. Teh, Y. Gal, Continual Learning via Sequential Function-Space Variational Inference, in: Proc. International Conference on Machine Learning (ICML), vol. PMLR 162, 18871–18887, 2022.

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

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

[52] F. Ye, A. G. Bors, Learning Latent Representations Across Multiple Data Domains Using Lifelong VAEGAN, in: Proc. European Conf. on Computer Vision (ECCV), vol. LNCS 12365, 777–795, 2020.

[53] 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), 10695–10704, 2021.

[54] F. Ye, A. G. Bors, Continual Variational Autoencoder Learning via Online Cooperative Memorization, in: Proc. European Conference on Computer Vision (ECCV), vol. LNCS 13681, 531–549, 2022.

[55] F. Ye, A. G. Bors, Lifelong Generative Modelling Using Dynamic Expansion Graph Model, in: Proc. of the AAAI Conference on Artificial Intelligence, vol. 36, 8857–8865, 2022.

[56] F. Ye, A. G. Bors, Lifelong Teacher-Student Network Learning, IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (10) (2022) 6280–6296.

[57] F. Ye, A. G. Bors, Task-free continual learning via online discrepancy distance learning, Advances in Neural Information Processing Systems 35 (2022) 23675–23688.

[58] F. Ye, A. G. Bors, Dynamic Self-Supervised Teacher-Student Network Learning, IEEE Transactions on Pattern Analysis and Machine Intelligence 45 (5) (2023) 5731–5748.

[59] F. Ye, A. G. Bors, Lifelong variational autoencoder via online adversarial expansion strategy, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, 10909–10917, 2023.

[60] F. Ye, A. G. Bors, Wasserstein Expansible Variational Autoencoder for Discriminative and Generative Continual Learning, in: Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV), 18665–18675, 2023.

[61] F. Ye, A. G. Bors, Lifelong Dual Generative Adversarial Nets Learning in Tandem, IEEE Transactions on Cybernetics 54 (3) (2024) 1353–1365.

[62] F. Zenke, B. Poole, S. Ganguli, Continual learning through synaptic intelligence, in: Proc. of Int. Conf. on Machine Learning, vol. PLMR 70, 3987–3995, 2017.

[63] Y. Zhang, T. Liu, M. Long, M. Jordan, Bridging theory and algorithm for domain adaptation, in: Proc. International Conference on Machine Learning (ICML), vol. PMLR 97, 7404–7413, 2019.

[64] B. Zhu, C. Wang, K. Xu, D. Feng, Z. Zhou, X. Zhu, Learning incremental audio-visual representation for continual multimodal understanding, Knowledge-Based Systems (2024) 112513.

[65] X. Zhu, J. Yi, L. Zhang, Continual Learning With Unknown Task Boundary, IEEE Transactions on Neural Networks and Learning Systems (2024) 1–13doi: 10.1109/TNNLS.2024.3412934.