Task-Free Continual Learning via Online Discrepancy Distance Learning

Fei Ye and Adrian G. Bors
Department of Computer Science
University of York
York, YO10 5GH, UK
{fy689,adrian.bors}@york.ac.uk

Abstract

Learning from non-stationary data streams, also called Task-Free Continual Learning (TFCL) remains challenging due to the absence of explicit task information. Although recently some methods have been proposed for TFCL, they lack theoretical guarantees. Moreover, forgetting analysis during TFCL was not studied theoretically before. This paper develops a new theoretical analysis framework which provides generalization bounds based on the discrepancy distance between the visited samples and the entire information made available for training the model. This analysis gives new insights into the forgetting behaviour in classification tasks. Inspired by this theoretical model, we propose a new approach enabled by the dynamic component expansion mechanism for a mixture model, namely the Online Discrepancy Distance Learning (ODDL). ODDL estimates the discrepancy between the probabilistic representation of the current memory buffer and the already accumulated knowledge and uses it as the expansion signal to ensure a compact network architecture with optimal performance. We then propose a new sample selection approach that selectively stores the most relevant samples into the memory buffer through the discrepancy-based measure, further improving the performance. We perform several TFCL experiments with the proposed methodology, which demonstrate that the proposed approach achieves the state of the art performance.

1 Introduction

Continual learning (CL) and its extension to lifelong learning, represents one of the most desired functions in an artificial intelligence system, representing the capability of learning new concepts while preserving the knowledge of past experiences [32]. Such an ability can be used in many real-time applications such as robotics, health investigative systems, autonomous vehicles [20] or for guiding agents exploring artificial (meta) universes, requiring adapting to a changing environment. Unfortunately, modern deep learning models suffer from a degenerated performance on past data after learning novel knowledge, a phenomenon called catastrophic forgetting [13].

A popular attempt to relieve forgetting in CL is by employing a small memory buffer to preserve a few past samples and replay them when training on a new task [1, 6]. However, when there are restrictions on the available memory capacity, memory-based approaches would suffer from degenerated performance on past tasks, especially when aiming to learn an infinite number of tasks. Recently, the Dynamic Expansion Model (DEM) [51] has shown promising results in CL, aiming to guarantee optimal performance by preserving the previously learnt knowledge through the parameters of frozen components trained on past data, while adding a new component when learning a novel task. However, such approaches require knowing where and when the knowledge associated with a given task is changed, which is not always applicable in a real environment.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

In this paper, we address a more realistic scenario, called Task-Free Continual Learning (TFCL) [3], where task identities are not available while the model can only access a small batch of samples at a given time. Most existing CL methods requiring the task label can be adapted to TFCL by removing the task information dependency. For instance, memory-based approaches can store a few past samples from the data stream at each training time and replay them during later training steps [8, 12]. However, such an approach requires to carefully design the sample selection criterion to avoid memory overload. The key challenge for the memory-based approach is the negative backward transfer caused by the stored samples that interfere with the model's updating with incoming samples [6]. This issue can be relieved by DEM in which previously learnt samples are preserved into frozen components and do not interfere with the learning of probabilistic representations of new data [24, 38]. However, these approaches do not provide any theoretical guarantees and there are no studies analysing the trade-off between the model's generalization and its complexity under TFCL.

Recent attempts have provided the theoretical analysis for CL from different perspectives including the risk bound [46, 51], NP-hard problem [17], Teacher-Student framework [23, 58] and game theory [37]. However all these approaches require strong assumptions, such as defining the task identities, which is not available in TFCL. This inspires us to bridge the gaps between the underlying theory and the algorithm implementation for TFCL. We propose a theoretical classification framework, which provides new insights in the forgetting behaviour analysis and guidance for algorithm design addressing catastrophic forgetting. The primary motivation behind the proposed theoretical framework is that we can formulate forgetting as a generalization error in the domain adaptation theory. Based on this analysis we extend the domain adaptation theory [29] to derive time-dependent generalization risk bounds, explicitly explaining the forgetting process at each training step.

Inspired by the theory, we devise the Online Discrepancy Distance Learning (ODDL) method which introduces a new expansion mechanism based on the discrepancy distance estimation for implementing TFCL. The proposed expansion mechanism detects the data distribution shift by evaluating the variance of the discrepancy distance during the training. This model enables a trade-off mechanism between the model's generalization and complexity. We also propose a new sample selection approach based on the discrepancy-based criterion, which guides storing diverse samples with respect to the already learnt knowledge, further improving performance. Our contributions are :

- This paper is the first research study to propose a new theoretical framework for TFCL, which provides new insights into the forgetting behaviour of the model in classification tasks.
- Inspired by the theoretical analysis, we develop a novel dynamic expansion approach, which ensures a compact model architecture enabled by optimal performance.
- We propose a new sample selection approach that selects appropriate data samples for the memory buffer, further improving performance.
- The proposed method achieves state of the art results on TFCL benchmarks,

2 Related works

Continual learning defines a learning paradigm which aims to learn a sequence of tasks without forgetting. Catastrophic forgetting is a major challenge in continual learning. One of the most popular approaches to relieve forgetting is by imposing a regularization loss within the optimization procedure [7, 11, 13, 16, 19, 25, 26, 31, 34, 35, 40, 41, 57], where the network's parameters which are important to the past tasks are penalized when updating. Another kind of approaches for continual learning focuses on the memory system, which usually employs a small memory buffer [1, 5, 6, 28, 36, 44, 59] to store a few past data or trains a generator to provide the replay samples when learning new tasks [38, 43, 46, 47, 52, 58, 53]. However, these approaches usually rely on knowing the task information, which is not applicable in TFCL.

Task-free continual learning is a special scenario in CL where a model can only see one or very few samples in each training step/time without having any task labels. Using a small memory buffer to store past samples has shown benefits for TFCL and was firstly investigated in [3, 54, 56]. This memory replay approach was then extended by employing Generative Replay Mechanisms (GRMs) for training both a Variational Autoencoder (VAEs) [15] and a classifier, where a new retrieving mechanism is used to select specific data samples, called the Maximal Interfered Retrieval (MIR), [2]. The Gradient Sample Selection (GSS) [1] is another sample selection approach that treats sample selection as a constrained optimization reduction. More recently, a Learner-Evaluator framework is proposed for TFCL, called the Continual Prototype Evolution (CoPE) [8] which stores the same number of samples for each class in the memory in order to ensure the balance replay. Another direction for the memory-based approaches is to edit the stored samples which would increase the loss in the upcoming model updates, called the Gradient based Memory EDiting (GMED) [12], which can be employed in the existing CL models to further enhance their performance.

Dynamic expansion models aim to automatically increase the model's capacity to adapt to new tasks by adding new hidden layers and units. Such an approach, called the Continual Unsupervised Representation Learning (CURL) [38], dynamically builds new inference models when meeting the expansion criterion. However, since CURL still requires a GRM to relieve forgetting, it would lead to a negative knowledge transfer when updating the network's parameters to adapt to a new task. This issue can be addressed by using Dirichlet processes by adding new components while freezing all previously learnt members, in a model called the Continual Neural Dirichlet Process Mixture (CN-DPM), [24]. However, the expansion criterion used by these approaches relies on the change of the loss when training each time, which does not have any theoretical guarantees.

3 Theoretical analysis of TFCL

In this section, we firstly introduce learning settings and notations, and then we analyze the forgetting behaviour for a single as well as for a dynamic expansion model by deriving their Generalization Bounds (GBs).

3.1 Preliminary

Let $\mathcal{X}$ be the input space and $\mathcal{Y}$ represent the output space which is $\{-1,1\}$ for binary classification and $\{1,2,\dots,n'\}, n' > 2$ for multi-class classification. Let $\mathcal{D}_i^T = \{\mathbf{x}_j^T, y_j^T\}_{j=1}^{N_i^T}$ and $\mathcal{D}_i^S = \{\mathbf{x}_j^S, y_j^S\}_{j=1}^{N_i^S}$ represent the training and testing sets for the i-th dataset where $\mathbf{x}_j^T \in \mathcal{X}$ and $y_j^T \in \mathcal{Y}$ are the image and the associated ground truth label. N_i^T and N_i^S are the total number of samples for $\mathcal{D}_i^T$ and $\mathcal{D}_i^S$, respectively. In the paper, we mainly focus on the task-free class-incremental learning, described as follows.

Definition 1. (Data stream.) For a given t-th training dataset $\mathcal{D}_t^S$ with C_t^S data categories, let us consider a data stream $\mathcal{S}$ which consists of samples $\mathcal{D}_{t,j}^S$ from each category, expressed by $S = \bigcup_{j=1}^{C_t^S} \mathcal{D}_{t,j}^S$. Let $\mathcal{D}_{t,j}^T$ represent the set of samples drawn from the j-th category of $\mathcal{D}_t^T$. Let $\mathbb{P}_{t,j}^T$ and $\mathbb{P}_{t,j}^S$ represent the distributions for $\mathcal{D}_{t,j}^T$ and $\mathcal{D}_{t,j}^S$, respectively. Let $\mathbb{P}_{t,j}^{T,\mathcal{X}}$ represent the marginal distribution over $\mathcal{X}$.

Definition 2. (Learning setting.) Let $\mathcal{T}_i$ represent the i-th training step. For a given data stream $\mathcal{S}$, we assume that there are a total of n training steps for $\mathcal{S}$, where each training step $\mathcal{T}_i$ is associated with a small batch of paired samples $\{\mathbf{X}_i^b, \mathbf{Y}_i^b\}$ drawn from $\mathcal{S}$, expressed by $\mathcal{S} = \bigcup_{i=1}^n \{\mathbf{X}_i^b, \mathbf{Y}_i^b\}, \quad \{\mathbf{X}_i^b, \mathbf{Y}_i^b\} \cap \{\mathbf{X}_j^b, \mathbf{Y}_j^b\} = \varnothing$, where $i \neq j$ and a model (classifier) can only access $\{\mathbf{X}_i^b, \mathbf{Y}_i^b\}$ at $\mathcal{T}_i$ while all previous batches are not available. After finishing all training steps, we evaluate the classification accuracy of the model on the testing set $\mathcal{D}_t^T$. In the following, we define the model and the memory buffer.

Definition 3. (Model and memory.) Let us consider h a model implemented by a classifier, and $\mathcal{H} = \{h \,|\, h\colon \mathcal{X} \to \mathcal{Y}\}$ the space of classifiers. Let $\mathcal{M}_i$ be a memory buffer at $\mathcal{T}_i$. We assume that $\mathcal{M}_i$ randomly removes samples from the memory buffer while continually adding new data to its memory at each training step. Let $\mathbb{P}_{\mathcal{M}_i}$ represent the probabilistic representation of $\mathcal{M}_i$ and $|\mathcal{M}_i|$ its cardinality.

3.2 Measuring the distribution shift

In TFCL, the distance between the target domain (testing set) and the source domain (memory) would be dynamically changed during each training step. We can use the discrepancy distance [29] to measure this gap through the analysis of the model's risk.

Definition 4. (The risk.) For a given distribution $\mathbb{P}^S_{t,j}$, the risk of a model h is defined as $\mathcal{R}\big(h, \mathbb{P}^S_{t,j}\big) \triangleq \mathbb{E}_{\{\mathbf{x},y\}\sim\mathbb{P}^S_{t,j}}\big[\mathcal{L}\big(y, h(\mathbf{x})\big)\big]$ where $\mathcal{L}\colon \mathcal{Y}\times\mathcal{Y}\to[0,1]$ is the loss function.

Definition 5. (Discrepancy distance.) For two given distributions $\mathbb{P}^T_{t,j}$ and $\mathbb{P}^S_{t,j}$, the discrepancy distance $\mathcal{L}_d$ is defined on two marginals as :

$$\mathcal{L}_d\big(\mathbb{P}^{T,\mathcal{X}}_{t,j}, \mathbb{P}^{S,\mathcal{X}}_{t,j}\big) \triangleq \sup_{(h,h')\in\mathcal{H}^2} \Big|\mathbb{E}_{\mathbf{x}\sim\mathbb{P}^{T,\mathcal{X}}_{t,j}}\big[\mathcal{L}\big(h(\mathbf{x}), h'(\mathbf{x})\big)\big] - \mathbb{E}_{\mathbf{x}\sim\mathbb{P}^{S,\mathcal{X}}_{t,j}}\big[\mathcal{L}\big(h(\mathbf{x}), h'(\mathbf{x})\big)\big]\Big|\,, \tag{1}$$

where $\{h, h'\}\in\mathcal{H}$. In practice, the discrepancy distance $\mathcal{L}_d(\cdot,\cdot)$ can be estimated as the upper bound based on the Rademacher complexity which is used in the domain adaptation theory as a measure of richness for a particular hypothesis space [30, 60].

Corollary 1. [29] For two given domains $\mathbb{P}^{T,\mathcal{X}}_{t,j}$ and $\mathbb{P}^{S,\mathcal{X}}_{t,j}$, let $U_{\mathcal{P}}$ and $U_{\mathbb{P}}$ represent sample sets of sizes $m_{\mathcal{P}}$ and $m_{\mathbb{P}}$, drawn independently from $\mathbb{P}^{T,\mathcal{X}}_{t,j}$ and $\mathbb{P}^{S,\mathcal{X}}_{t,j}$. Let $\widehat{\mathbb{P}}^{T,\mathcal{X}}_{t,j}$ and $\widehat{\mathbb{P}}^{S,\mathcal{X}}_{t,j}$ represent the empirical distributions for $U_{\mathcal{P}}$ and $U_{\mathbb{P}}$. Let $\mathcal{L}(\mathbf{x}',\mathbf{x}) = |\mathbf{x}'-\mathbf{x}|^1$ be a loss function (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 $1-\delta$, we have :

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

where $C^\star = 4q\left(\mathrm{Re}_{U_{\mathcal{P}}}(\mathcal{H}) + \mathrm{Re}_{U_{\mathbb{P}}}(\mathcal{H})\right)$ and $\mathrm{Re}_{U_{\mathcal{P}}}(\mathcal{H})$ is the Rademacher complexity (Appendix-B from Supplemental Material (SM)). Let $\mathcal{L}_{\widehat{d}}(\mathbb{P}^{T,\mathcal{X}}_{t,j}, \mathbb{P}^{S,\mathcal{X}}_{t,j})$ represent the Right-Hand Side (RHS) of Eq. (2). We also assume that $\mathcal{L}\colon\mathcal{Y}\times\mathcal{Y}\to[0,1]$ is a symmetric and bounded loss function $\forall(y,y')\in\mathcal{Y}^2, \mathcal{L}(y,y')\le U'$, and $\mathcal{L}(\cdot,\cdot)$ obeys the triangle inequality, where U' is a positive number.

3.3 GB for a single model

Based on the definitions from Section 3.2, we firstly derive the GB that can describe the learning process of a single model under TFCL.

Theorem 1. Let $\mathcal{P}_i$ represent the distribution of all visited training samples (including all previous batches) drawn from $\mathcal{S}$ at $\mathcal{T}_i$. Let $h_{\mathcal{P}_i} = \arg\min_{h\in\mathcal{H}}\mathcal{R}(h,\mathcal{P}_i)$ and $h_{\mathcal{M}_i} = \arg\min_{h\in\mathcal{H}}\mathcal{R}(h,\mathbb{P}_{\mathcal{M}_i})$ represent the ideal classifiers for $\mathcal{P}_i$ and $\mathbb{P}_{\mathcal{M}_i}$, respectively. We derive the GB between $\mathcal{P}_i$ and $\mathbb{P}_{\mathcal{M}_i}$, based on the results from Corollary 1 :

$$\mathcal{R}\big(h,\mathcal{P}_i\big) \le \mathcal{R}\big(h, h_{\mathcal{M}_i}, \mathbb{P}_{\mathcal{M}_i}\big) + \mathcal{L}_{\widehat{d}}\big(\mathcal{P}^{\mathcal{X}}_i, \mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}\big) + \eta\big(\mathcal{P}_i, \mathbb{P}_{\mathcal{M}_i}\big)\,, \tag{3}$$

where $\eta\big(\mathcal{P}_i,\mathbb{P}_{\mathcal{M}_i}\big)$ is the optimal combined error $\mathcal{R}(h_{\mathcal{P}_i}, h_{\mathcal{M}_i}, \mathcal{P}_i) + \mathcal{R}(h_{\mathcal{P}_i}, h^\star_{\mathcal{P}_i}, \mathcal{P}_i)$ where $\mathcal{R}(h_{\mathcal{P}_i}, h_{\mathcal{M}_i}, \mathbb{P}_{\mathcal{M}_i})$ is the risk, expressed by $\mathbb{E}_{\mathbf{x}\sim\mathbb{P}_{\mathcal{M}_i}}[\mathcal{L}(h_{\mathcal{P}_i}(\mathbf{x}), h_{\mathcal{M}_i}(\mathbf{x}))]$ and $h^\star_{\mathcal{P}_i}$ is the true labeling function for $\mathcal{P}_i$.

The proof is provided in Appendix-A from the SM. Compared to the GB used in the domain adaptation [29], Theorem 1 provides an explicit way to measure the gap between the model's prediction and the true labels in each training step ($\mathcal{T}_i$). During the initial training stages (when i is very small), the memory $\mathcal{M}_i$ can store all previous samples and GB is tight. However, as the number of training steps increases, the discrepancy distance $\mathcal{L}_{\widehat{d}}\big(\mathcal{P}^{\mathcal{X}}_i, \mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}\big)$ would increase because $\mathcal{M}_i$ would lose the knowledge about previously learnt samples. This can lead to a degenerated performance on $\mathcal{P}_i$, corresponding to the forgetting process. Next we extend Theorem 1 to analyze the generalization performance on testing sets.

Theorem 2. For a given target domain $\mathbb{P}^T_{t,j}$, we derive the GB for a model at the training step $\mathcal{T}_i$:

$$\mathcal{R}\big(h,\mathbb{P}^T_{t,j}\big) \le \mathcal{R}\big(h, h_{\mathcal{M}_i}, \mathbb{P}_{\mathcal{M}_i}\big) + \mathcal{L}_{\widehat{d}}\big(\mathbb{P}^{T,\mathcal{X}}_{t,j}, \mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}\big) + \eta\big(\mathbb{P}^T_{t,j}, \mathbb{P}_{\mathcal{M}_i}\big)\,, \tag{4}$$

The proof is similar to that for Theorem 1. We can observe that the generalization performance on a target domain $\mathbb{P}^T_{t,j}$, by a model h is relying mainly on the discrepancy distance between $\mathbb{P}^{T,\mathcal{X}}_{t,j}$ and $\mathbb{P}_{\mathcal{M}_i}$. In practice, we usually measure the generalization performance of h on several data categories where each category is represented by a different underlying distribution. In the following, we extend Theorem 2 for multiple target distributions.

Lemma 1. For a given data stream $S = \bigcup_{j=1}^{C^S_t}\mathcal{D}^S_{t,j}$ consisting of samples from $\mathcal{D}^S_t$, let $\mathcal{D}^T_t$ be the corresponding testing set and $\mathbb{P}^T_{t,j}$ represent the distribution of samples for the j-th category from

$\mathcal{D}_t^T$, we derive the GB for multiple target domains as:

$$\sum\nolimits_{j=1}^{C_t^T} \Big\{ \mathcal{R}\big(h, \mathbb{P}_{t,j}^T\big) \Big\} \leq \sum\nolimits_{j=1}^{C_t^T} \Big\{ \mathcal{R}\big(h, h_{\mathcal{M}_i}, \mathbb{P}_{\mathcal{M}_i}\big) + \mathcal{L}_{\widehat{d}}\big(\mathbb{P}_{t,j}^{T,\mathcal{X}}, \mathbb{P}_{\mathcal{M}_i}^{\mathcal{X}}\big) + \eta\big(\mathbb{P}_{t,j}^T, \mathbb{P}_{\mathcal{M}_i}\big) \Big\}, \quad (5)$$

where C_t^T represents the number of testing data streams.

Remarks. Lemma 1 had the following observations : 1) The optimal performance of the model h on the testing set can be achieved by minimizing the discrepancy distance between each target domain $\mathbb{P}_{t,j}^{T,\mathcal{X}}$ and the distribution $\mathbb{P}_{\mathcal{M}_i}$ at the training step $\mathcal{T}_i$. 2) [17] employs the set theory to theoretically demonstrate that a perfect memory is crucial for CL. In contrast, we evaluate the memory quality using the discrepancy distance in Eq. (5), which provides a practical way to investigate the relationship between the memory and forgetting behaviour of existing approaches [6, 8] at each training step without requiring any task information (See more details in Appendix-D from SM). 3) [51] introduces a similar risk bound for forgetting analysis, which still requires the task information. In contrast, the proposed GB can be used in a more realistic CL scenario. Moreover, we provide the theoretical analysis for component diversity (Appendix-C from SM), which is missing in [51].

3.4 GB for the dynamic expansion mechanism

As discussed in Lemma 1, a memory of fixed capacity would lead to degenerated performance on all target domains. The other problem for a single memory system is the negative backward transfer [27] in which the performance of the model is decreased due to samples being drawn from entirely different distributions [14]. A Dynamic Expansion Model (DEM) can address these limitations from two aspects :1) DEM relieves the negative transfer by preserving the previously learnt knowledge into a frozen component from a mixture system; 2) DEM would achieve better generalization performance under TFCL by allowing each component to model one or only a few similar underlying data distributions. We derive GB for DEM and show the advantages of DEM for TFCL.

Definition 7. (Dynamic expansion mechanism.) Let $\mathcal{G}$ represent a dynamic expansion model and G_j represent the j-th component in $\mathcal{G}$, implemented by a single classifier. $\mathcal{G}$ starts with training its first component during the initial training phase and would add new components during the following training steps. In order to overcome forgetting, only the newly created component is updated each time, while all previously trained components have their parameters frozen.

Theorem 3. For a given data stream $\mathcal{S} = \{\mathbf{X}_1^b, \cdots, \mathbf{X}_n^b\}$, let $\mathcal{P}_{(i,j)}$ represent the distribution of the j-th training batch $\mathbf{X}_j^b$ (visited) drawn from $\mathcal{S}$ at $\mathcal{T}_i$. We assume that $\mathcal{G} = \{G_1, \cdots, G_c\}$ trained c components at $\mathcal{T}_i$. Let $\mathcal{T} = \{\mathcal{T}_{k_1}, \cdots, \mathcal{T}_{k_c}\}$ be a set of training steps, where G_j was frozen at $\mathcal{T}_{k_j}$. We derive the GB for $\mathcal{G}$ at $\mathcal{T}_i$ as:

$$\frac{1}{i}\sum\nolimits_{j=1}^{i} \big\{\mathcal{R}\big(h, \mathcal{P}_{(i,j)}\big)\big\} \leq \frac{1}{i}\sum\nolimits_{j=1}^{i} \big\{\mathrm{F}_S\big(\mathcal{P}_{(i,j)}, \mathcal{G}\big)\big\}, \quad (6)$$

where $\mathrm{F}_S(\cdot,\cdot)$ is the selection function, defined as :

$$\mathrm{F}_S\big(\mathcal{P}_{(i,j)}, \mathcal{G}\big) \triangleq \min_{k_1,\cdots,k_c} \Big\{ \mathcal{R}\big(h, h_{\mathcal{M}_{k_i}}, \mathbb{P}_{\mathcal{M}_{k_i}}\big) + \mathcal{L}_{\widehat{d}}\big(\mathcal{P}_{(i,j)}^{\mathcal{X}}, \mathbb{P}_{\mathcal{M}_{k_i}}^{\mathcal{X}}\big) + \eta\big(\mathcal{P}_{(i,j)}, \mathbb{P}_{\mathcal{M}_{k_i}}\big) \Big\}, \quad (7)$$

where $\mathbb{P}_{\mathcal{M}_{k_i}}$ represents the memory distribution at $\mathcal{T}_{k_i}$. The proof is provided in Appendix-B from SM. It notes that $\mathrm{F}_S(\cdot,\cdot)$ can be used for arbitrary distributions. We assume an ideal model selection in Eq. (7), where always the component with the minimal risk is chosen. DEM can achieve the minimal upper bound to the risk (Left Hand Side (LHS) of Eq. (6)) when comparing with a single model (Theorem 3),. Then, we derive the GB for analyzing the generalization performance of $\mathcal{G}$ on multiple target distributions.

Lemma 2. For a given data stream $S = \bigcup_j^{C_t^S} \mathcal{D}_{t,j}^S$ consisting of samples from $\mathcal{D}_t^S$, we have a set of target sets $\{\mathcal{D}_{t,1}^T, \cdots, \mathcal{D}_{t,C_t^T}^T\}$, where each $\mathcal{D}_{t,j}^T$ contains $C_{t,j}^b$ batches of samples. Let $\mathbb{P}_{t,j}^T(d)$ represent the distribution of the d-th batch of samples in $\mathcal{D}_{t,j}^T$. We assume that $\mathcal{G}$ consists of c components trained on samples from $\mathcal{S}$, at $\mathcal{T}_i$. We derive the GB for multiple target domains as :

$$\sum\nolimits_{j=1}^{C_t^T} \Big\{ \sum\nolimits_{d=1}^{C_{t,j}^b} \mathcal{R}\big(h, \mathbb{P}_{t,j}^T(d)\big) \Big\} \leq \sum\nolimits_{j=1}^{C_t^T} \Big\{ \sum\nolimits_{d=1}^{C_{t,j}^b} \big\{ \mathrm{F}_S\big(\mathbb{P}_{t,j}^T(d), \mathcal{G}\big) \big\} \Big\}. \quad (8)$$

Remark. We have several observations from Lemma 2 : 1) The generalization performance of $\mathcal{G}$ is relying on the discrepancy distance between each target distribution $\mathbb{P}_{t,j}^T$ and the memory distribution

$\mathbb{P}_{\mathcal{M}_{k_i}}$ of the selected component (Also see details in Appendix-C from SM). 2) Eq. (8) provides the analysis of the trade-off between the model's complexity and generalization for DEM [24, 38]. By adding new components, $\mathcal{G}$ would capture additional information of each target distribution and thus improve its performance. On the other hand, the selection process ensures a probabilistic diversity of stored information, aiming to capture more knowledge with a minimal number of components.

In practice, we usually perform the model selection for $\mathcal{G}$ by using a certain criterion that only accesses the testing samples without task labels. Therefore, we introduce a selection criterion $\widehat{\mathrm{F}}(\cdot,\cdot)$, implemented by comparing the sample log-likelihood :

$$\widehat{\mathrm{F}}\big(\mathbb{P}^T_{t,j}(d),\mathcal{G}\big) \stackrel{\Delta}{=} \mathop{\arg\max}_{k_1,\cdots,k_c} \big\{\mathbb{E}_{\mathbf{x}\sim\mathbb{P}^T_{t,j}(d)}[\hat{f}(\mathbf{x},G_{k_j})]\big\}, \tag{9}$$

where $j=\{1,\cdots,c\}$ and $\hat{f}(\cdot,\cdot)$ is a pre-defined sample-log likelihood function. Then Eq. (9) is used for model selection:

$$\begin{aligned}\widehat{\mathrm{F}}_S\big(\mathbb{P}^T_{t,j}(d),\mathcal{G}\big) = \big\{&\mathcal{R}\big(h,h_{\mathcal{M}_s},\mathbb{P}_{\mathcal{M}_s}\big) + \mathcal{L}_{\widehat{d}}\big(\mathbb{P}^{T,\mathcal{X}}_{t,j}(d),\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_s}\big) + \eta\big(\mathbb{P}^T_{t,j}(d),\mathbb{P}_{\mathcal{M}_s}\big) \mid \\ &s=\widehat{\mathrm{F}}\big(\mathbb{P}^T_{t,j}(d),\mathcal{G}\big)\big\}\,. \end{aligned} \tag{10}$$

We rewrite Eq. (8) using Eq. (10), resulting in :

$$\sum\nolimits_{j=1}^{C^T_t}\Big\{\sum\nolimits_{d=1}^{C^b_{t,j}}\mathcal{R}\big(h,\mathbb{P}^T_{t,j}(d)\big)\Big\} \le \sum\nolimits_{j=1}^{C^T_t}\Big\{\sum\nolimits_{d=1}^{C^b_{t,j}}\big\{\widehat{\mathrm{F}}_S\big(\mathbb{P}^T_{t,j}(d),\mathcal{G}\big)\big\}\Big\}\,. \tag{11}$$

Compared with an ideal solution (Eq. (8)), Eq. (11) would involve the extra error terms caused by the selection process (Eq. (10)), expressed as $\sum_{j=1}^{C^T_t}\Big\{\sum_{d=1}^{C^b_{t,j}}\big\{\widehat{\mathrm{F}}_S\big(\mathbb{P}^T_{t,j}(d),\mathcal{G}\big) - \mathrm{F}_S\big(\mathbb{P}^T_{t,j}(d),\mathcal{G}\big)\big\}\Big\}$. In Section 4, we introduce a new CL framework according to the theoretical analysis.

3.5 Parallels with other studies defining lifelong learning bounds

In this section, we discuss the differences between the results in this paper and those from other studies proposing lifelong learning bounds. The bound (Theorem 1) in [30] assumes that the model is trained on all previous samples, which is not practical under a TFCL setting. In contrast, in Theorem 1 from this paper, the model is trained on a memory buffer, which can be used in the context of TFCL. In addition, the bounds in [30] mainly provide the theoretical guarantees for the performance when learning a new data distribution (Theorem 1 from [30]), which does not provide an analysis of the model's forgetting. In contrast, this paper studies the forgetting analysis of our model and its theoretical developments can be easily extended for analyzing the forgetting behaviour of a variety of continual learning methods, while this cannot be said about the study from [30] (See Appendix-D from supplemental material).

When comparing with [33], our theoretical analysis does not rely on explicit task boundaries, which is a more realistic assumption for TFCL. In addition, [33] employs the KL divergence to measure the distance between two distributions, which would require knowing the explicit distribution form and is thus hard to evaluate in practice during the learning. However, our theory employs the discrepancy distance, which can be reliably estimated. Moreover, the study from [33] does not develop a practical algorithm to be used according to the theoretical analysis. In contrast, our theory provides guidelines for the algorithm design under the TFCL assumption. Finally, inspired by the proposed theoretical analysis, this paper develops a successful continual learning approach for TFCL.

Theorem 1 in [4], similarly to [30], assumes that the model is trained on uncertain data sets over time, which would not be suitable for TFCL since the model can not access previously learnt samples under TFCL. In contrast, Theorem 1 in our theory represents a realistic TFCL scenario in which the model is trained on a fixed-length memory buffer and is evaluated on all previously seen samples. Therefore, the analysis for the forgetting behaviour of the model under TFCL relies on Theorem 1 and its consequences. In addition, the study from [4], similarly to [30, 33], only provides a theoretical guarantee for a single model. In contrast, we extend our theoretical analysis to the dynamic expansion model, which motivates us to develop a novel continual learning approach for TFCL. Moreover, this paper is the first work to provide the theoretical analysis for the diversity of knowledge recorded by different components (See Appendix-C from supplemental material). This analysis indicates that by maintaining the knowledge diversity among different components we ensure a good trade-off between the model's complexity and generalization performance, thus providing invaluable insights into algorithm design for TFCL.

4 Methodology

4.1 Network architecture

Each component $G_j \in \mathcal{G}$ consists of a classifier h_j implemented by a neural network $f_{\varsigma_j}(\mathbf{x})$ with trainable parameters ς_j, and a variational autoencoder (VAE) model implemented by an encoding network enc_{ω_j} as well as the corresponding decoding network dec_{θ_j}, with trainable parameters $\{\omega_j, \theta_j\}$. Due to its robust generation and inference mechanisms, VAE has been widely used in many applications [48, 55, 49, 50, 51, 61]. This paper employs a VAE for discrepancy estimation and component selection. The loss function for the j-th component at $\mathcal{T}_i$ is defined as :

$$\mathcal{L}_{class}(G_j, \mathcal{M}_i) \triangleq \frac{1}{|\mathcal{M}_i|} \sum\nolimits_{t=1}^{|\mathcal{M}_i|} \left\{ \mathcal{L}_{ce}\big(h_j(\mathbf{x}_t), y_t\big) \right\}, \tag{12}$$

$$\mathcal{L}_{VAE}(G_j, \mathcal{M}_i) \triangleq \mathbb{E}_{q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x})}\left[\log p_{\theta_j}(\mathbf{x}_t\,|\,\mathbf{z})\right] - D_{KL}\left[q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x}_t)\,||\,p(\mathbf{z})\right], \tag{13}$$

where $\{\mathbf{x}_t, y_t\} \sim \mathcal{M}_i$, where $|\mathcal{M}_i|$ is the memory buffer size, and $\mathcal{L}_{ce}(\cdot)$ is the cross-entropy loss. $\mathcal{L}_{VAE}(\cdot,\cdot)$ is the VAE loss [15], $p_{\theta_j}(\mathbf{x}_t\,|\,\mathbf{z})$ and $q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x}_t)$ are the encoding and decoding distributions, implemented by enc_{ω_j} and dec_{θ_j}, respectively. We also implement $\hat{f}(\cdot,\cdot)$ in Eq. (9) by $-\mathcal{L}_{VAE}(\cdot,\cdot)$ for the component selection at the testing phase. The training algorithm for the proposed ODDL consists of three stages (See more information in Appendix-E from SM). In the initial training stage, we aim to learn the initial knowledge of the data stream and preserve it into a frozen component G_1, which can provide information for the dynamic expansion and sample selection evaluation in the subsequent learning. During the evaluator training stage we train the current component as the evaluator that estimates the discrepancy distance between each previously learnt component and the memory buffer, providing appropriate signals for the model expansion. In the sample selection stage, we train the current component with new data, while we aim to promote knowledge diversity among components.

In the following, we firstly propose a new dynamic expansion mechanism based on the discrepancy criterion. Then we introduce a new sample selection approach that can further improve the performance of the model. Finally, we provide the detailed algorithm implementation.

4.2 Discrepancy based mixture model expansion

From Lemma 2, we observe that the probabilistic diversity of trained components in $\mathcal{G}$ can ensure a compact network architecture while maintaining a good generalization performance (See Theorem 4 in Appendix-C from SM). In order to achieve this, we maximize the discrepancy between each trained component of $\mathcal{G}$ and the current memory buffer, during the training, by using the discrepancy distance (notations are defined in **Theorem 3**), expressed as :

$$\mathcal{M}^\star = \underset{\mathcal{M}_i}{\arg\max} \sum\nolimits_{j}^{c} \left\{ \mathcal{L}_{\widehat{d}}(\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_{k_j}}, \mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}) \right\}, \tag{14}$$

where $i = k_c + 1, \cdots, n$ represents the index of the training steps and k_c is the c-th component trained at $\mathcal{T}_{k_c}$. $\mathcal{M}^\star$ is the optimal memory. Eq. (14) can be seen as a recursive optimization problem when $\mathcal{G}$ dynamically adds new components (c is increased) during the training. In order to provide a practical way to solve Eq. (14) while balancing the model's complexity and performance, we derive an expansion criterion based on the discrepancy distance :

$$\min\left\{ \mathcal{L}_{\widehat{d}}\big(\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_{k_j}}, \mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}\big) \,|\, j = 1, \cdots, c \right\} \geq \lambda, \tag{15}$$

where $\lambda \in [0, 4]$ is an architecture expansion threshold. If the current memory distribution $\mathbb{P}_{\mathcal{M}_i}$ is sufficiently different from each component (satisfies Eq. (15)), $\mathcal{G}$ will add a new component to preserve the knowledge of the current memory $\mathcal{M}_i$, while also encouraging the probabilistic diversity among the trained components.

In the following, we describe the implementation. We start by training the first component G_1 which consists of a classifier h_1 (classification task) and a VAE model v_1 (model selection at the testing phase). We also train an additional component called the evaluator $G_e = \{h_e, v_e\}$, at the initial training stage, which aims to capture the future information about the data stream. Once the memory $\mathcal{M}_j$ reaches its maximum size at $\mathcal{T}_j$, we freeze the weights of the first component to preserve the previously learnt knowledge while continually training the evaluator during the following training steps. Since the evaluator continually captures the knowledge from the current memory $\mathcal{M}_i$, we check the model expansion using Eq. (15) at $\mathcal{T}_i$:

$$\mathcal{L}_d^\star\big(\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_{k_1}}, \mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}\big) \geq \lambda \,. \tag{16}$$

It notes that we can not access the previously learnt memory distribution $\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_{k_1}}$ and we approximate it by using the auxiliary distribution $\mathbb{P}^{\mathcal{X}}_{v_1^j}$ formed by samples drawn from v_1^j of G_1^j, where the superscript j represents the first component finishing the training at $\mathcal{T}_j$. $\mathcal{L}_d^\star(\cdot,\cdot)$ is the estimator of the discrepancy distance, achieved by h_e^i and h_1^j.

$$\mathcal{L}_d^\star\big(\mathbb{P}^{\mathcal{X}}_{v_1^j}, \mathbb{P}_{\mathcal{M}_i}\big) \triangleq \Big|\mathbb{E}_{\mathbf{x}\sim\mathbb{P}^{\mathcal{X}}_{v_1^j}}\big[\mathcal{L}\big(h_1^j(\mathbf{x}), h_e^i(\mathbf{x})\big)\big] - \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_{\mathcal{M}_i}}\big[\mathcal{L}\big(h_1^j(\mathbf{x}), h_e^i(\mathbf{x})\big)\big]\Big|. \tag{17}$$

If Eq. (16) is fulfilled, then we add G_e^i into the model $\mathcal{G}$ while building a new evaluator $(G_e^{i+1} = G_3^{i+1})$ at $\mathcal{T}_{i+1}$; otherwise, we train $G_e^i \rightarrow G_e^{i+1}$ at the next training step, $\mathcal{T}_{i+1}$. Furthermore, for satisfying the diversity of knowledge in the components from the mixture model (See Lemma 2), we clear the memory $\mathcal{M}_i$ when performing the expansion in order to ensure that we would learn non-overlapping distributions during the following training steps. We can also extend Eq. (16) to the expansion criterion for $\mathcal{G} = \{G_1, \cdots, G_{s-1}\}$ that has already preserved $(s-1)$ components, illustrated in Fig. 1:

$$\min\Big\{\mathcal{L}_d^\star\big(\mathbb{P}^{\mathcal{X}}_{v_j^{k_j}}, \mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}\big) \,|\, j = 1,\cdots,s-1\Big\} \geq \lambda \,, \tag{18}$$

where $\mathbb{P}^{\mathcal{X}}_{v_j^{k_j}}$ is the distribution of the generated samples by $G_j^{k_j}$ and denote $S_j = \mathcal{L}_d^\star\big(\mathbb{P}^{\mathcal{X}}_{v_j^{k_j}}, \mathbb{P}_{\mathcal{M}_i}\big)$.

4.3 Sample selection

According to Lemma 2, the probabilistic diversity of the knowledge accumulated in the components is crucial for the performance. In the following, we also introduce a novel sample selection approach from the memory buffer that further encourages component diversity. The primary motivation behind the proposed sample selection approach is that we desire to store in the memory buffer those samples that are completely different from the data used for training the other components. This mechanism enables the currently created component to capture a different underlying data distribution. To implement this goal, we estimate the discrepancy distance on a pair of samples as the diversity score :

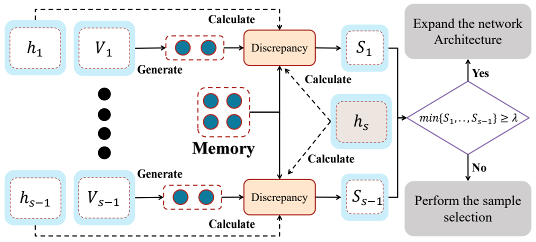


Figure 1: We generate the knowledge by using the VAE (V_j) of each previous component $G_j, j = 1,\cdots,s-1$, which is used to evaluate the discrepancy distance $S_j = \mathcal{L}_d^\star\big(\mathbb{P}^{\mathcal{X}}_{v_j}, \mathbb{P}_{\mathcal{M}_i}\big)$ at $\mathcal{T}_i$ (Eq. (17)) between G_j and the memory buffer. Then we use these discrepancy scores $\{S_1,\cdots,S_{s-1}\}$ to check the model expansion (Eq. (18))

$$\mathcal{L}_d^s(\mathcal{G}, \mathbf{x}_c) \triangleq \frac{1}{s}\sum\nolimits_{t=1}^{s} \big|\mathcal{L}(h_t^{k_t}(\mathbf{x}_c), h_e^i(\mathbf{x}_c)) - \mathcal{L}(h_t^{k_t}(\mathbf{x}_t^{k_t}), h_e^i(\mathbf{x}_t^{k_t}))\big| \,, \tag{19}$$

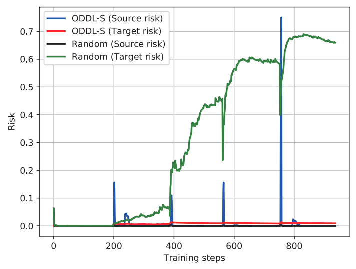
where $\mathbf{x}_c$ and $\mathbf{x}_t^{k_t}$ are the c-th sample from $\mathcal{M}_i$ and the generated one drawn from $G_t^{k_t}$, respectively. $h_t^{k_t}(\cdot)$ is the classifier of $G_t^{k_t}$ in $\mathcal{G}$. Eq. (19) evaluates the average discrepancy distance between the knowledge generated by each trained component and the stored samples, which guides for the sample selection at the training step $(\mathcal{T}_i)$ as :

$$\mathcal{M}_i = \bigcup\nolimits_{j=1}^{|\mathcal{M}_i'|-b} \mathcal{M}_i'[j] \,, \tag{20}$$

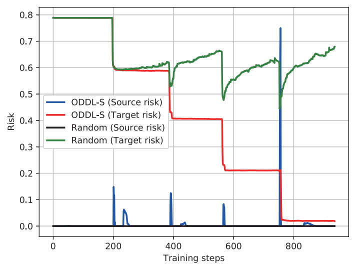
where $\mathcal{M}_i'$ is the sorted memory that satisfies the condition $\mathcal{L}_d^s(\mathcal{G}, \mathcal{M}_i'[a]) > \mathcal{L}_d^s(\mathcal{G}, \mathcal{M}_i'[q])$ for $a < q$. $\mathcal{M}_i'[j]$ represents the j-th stored sample and $b = 10$ is the batch size. We name the approach with sample selection as ODDL-S.

4.4 Algorithm implementation

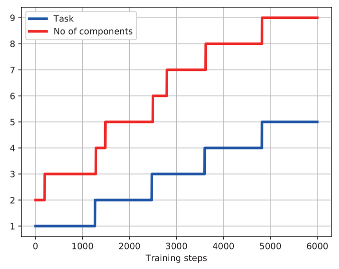
We provide the pseudocode (**Algorithm 1**) in Appendix-E from SM. The algorithm has three main stages :



(a) Risk on the training set.



(b) Risk on all target sets.



(c) Results on MNIST.

Figure 2: The forgetting analysis of a single model and DEM in (a) and (b) where the batch size is 64 in the data stream. Data distribution shift and increasing the number of components in ODDL-S during the training in (c).

- **(Initial training).** We start by building two components $\{G_1, G_2\}$ where we only add G_1 in $\mathcal{G}$ and consider $G_2 = G_e$ as the Evaluator. G_1 and G_2 are trained jointly using Eq. (13) and Eq. (12) during the initial training stage until the memory $\mathcal{M}_j$ reaches its maximum size $|\mathcal{M}|^{max}$ at a certain training step $\mathcal{T}_j$. Then we freeze the first component G_1 and perform the second stage.
- **(Evaluator training).** In this stage, we only update the Evaluator using Eq. (13) and Eq. (12) on $\mathcal{M}_{j+1}$ at $\mathcal{T}_{j+1}$. If $|\mathcal{M}_{j+1}| \geq |\mathcal{M}|^{max}$, then we evaluate the discrepancy distance using Eq. (17) to check the expansion (Eq. (18)). If the expansion criterion is satisfied then we add G_e to $\mathcal{G}$ and build a new Evaluator while cleaning up the memory $\mathcal{M}_{j+1}$, otherwise, we perform the sample selection.
- **(Sample selection).** We evaluate the diversity score for each stored sample in $\mathcal{M}_{j+1}$ using Eq. (19). We then perform the sample selection for $\mathcal{M}_{j+1}$ using Eq. (20) and return back to the second stage.

5 Experiments

We perform the experiments to address the following research questions: 1) What factors would cause the model's forgetting, and how to explain such behaviour? 2) How efficient is the proposed ODDL-S under TFCL benchmarks? 3) How important is each module in OODL-S?

In this experiment, we adapt the TFCL setting from [8] which employs several datasets including Split MNIST [22], Split CIFAR10 [18] and Split CIFAR100 [18]. The detailed information for datasets, hyperparameters and network architectures is provided in Appendix-F from the supplementary material. The code is available at `https://dtuzi123.github.io/ODDL/`.

5.1 Empirical results for the forgetting analysis

In this section, we investigate the forgetting behaviour of the model according to the proposed theoretical framework. Firstly, we train a single classifier h under Split MNIST database, as a baseline, with a memory buffer of the maximum size of 2000, and we randomly remove a batch of stored samples (batch size is 10) when the memory is full. Then we estimate $\mathcal{R}(h, \mathcal{P}_i)$ (target risk on all visited training samples), $\mathcal{R}(h, h_{\mathcal{M}_i}, \mathbb{P}_{\mathcal{M}_i})$ (source risk on the memory). We plot the results in Fig. 2-a, where "Random (Source risk)" represents the source risk of a single classifier. The results show that the source risk always keeps stable, and the target risk is small for a few initial training steps since the memory can capture all information of visited samples. However, as the number of training steps grows, the target risk is increased, which is caused by the memory that loses previous sam-

Table 1: The accuracy of various continual learning models for five independent runs.

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

ples, theoretically explained in Theorem 1. We also evaluate the risk of the baseline on all testing sets (target risk) and plot the results in Fig. 2-b where it is observed that the single model always leads to a large target risk during the training.

We evaluate the source and target risks for the proposed ODDL under Split MNIST and plot the results in Fig. 2. The performance of ODDL on the distribution $\mathcal{P}_i$ (Target risk) does not degenerate during the whole training phase. At the same time, the baseline tends to increase the target risk on $\mathcal{P}_i$ as the training steps increase, as shown in Fig. 2-a. Finally, the risk on all target distributions (all categories in the testing set) from each training step is shown in Fig. 2-b, where the proposed ODDL minimizes the target risk as gaining more knowledge during the training. In contrast, the baseline invariably leads to a large target risk even when the number of training steps increases. These results show that ODDL can relieve forgetting and achieve better generalization than the random approach on all target sets.

5.2 Results on TFCL benchmark

We provide the results in Tab. 1 where * and † denote the results cited from [8] and [12], respectively. We compare with several baselines including: finetune that directly trains a classifier on the data stream, GSS [1], Dynamic-OCM [54], MIR [2], Gradient Episodic Memory (GEM) [27], Incremental Classifier and Representation Learning (iCARL) [39], Reservoir [45], CURL, CNDPM, CoPE, ER + GMED and ER$_a$ + GMED [12] where ER is the Experience Replay [42] and ER$_a$ is ER with data augmentation. The number of components in ODDL-S and ODDL for Split MNIST, Split CIFAR10 and Split CIFAR100 is 7, 9, 7, respectively. The proposed approach outperforms CNDPM, which uses more parameters, on all datasets and achieves state-of-the-art performance.

In the following, we also evaluate the performance of the proposed approach on the large-scale dataset (MINI-ImageNet [21]), and Permuted MNIST [9]. We split MINI-ImageNet into 20 tasks (See details in Appendix-F1 from SM), namely Split MImageNet. We follow the setting from [2] where the maximum memory size is $10K$, and a smaller version of ResNet-18 [10] is used as the classifier. The hyperparameter λ used for learning Split MINI-ImageNet and Permuted MNIST is equal to 1.2 and 1.5, respectively. We compare with several state-of-the-art methods under Split MImageNet, reported in Tab. 2, where the results of other baselines are cited from [12]. These results show that the proposed ODDL-S outperforms different baselines under the challenging dataset.

Table 2: Classification accuracy for 20 runs when testing various models on Split MImageNet and Permuted MNIST.

Methods	Split MImageNet	Permuted MNIST
ER$_a$	25.92 ± 1.2	78.11 ± 0.7
ER + GMED	27.27 ± 1.8	78.86 ± 0.7
MIR+GMED	26.50 ± 1.3	79.25 ± 0.8
MIR	25.21 ± 2.2	79.13 ± 0.7
CNDPM	27.12 ±1.5	80.68 ± 0.7
ODDL	27.45 ± 0.9	82.33 ± 0.6
ODDL-S	**28.68** ± 1.5	**83.56** ± 0.5

5.3 Ablation study

We investigate whether the proposed discrepancy-based criterion can provide better signals for the expansion of ODDL-S. We train ODDL-S on Split MNIST, where we record the variance of tasks and the number of components in each training step. We plot the results in Fig. 2-c where "task" represents the number of tasks in each training step. We observe that the proposed discrepancy-based criterion can detect the data distribution shift accurately, allowing ODDL-S to expand the network architecture each time when detecting the data distribution shift. This also encourages the proposed ODDL-S to use a minimal number of components while achieving optimal performance, as discussed in Lemma 2. We also provide the analysis of how to maximize the memory bound in Appendix-F2.2, while more ablation study results are provided in Appendix-F.2 from the supplementary material.

6 Conclusion

In this paper, we develop a novel theoretical framework for Task-Free Continual Learning (TFCL), by defining a statistical discrepancy distance. Inspired by the theoretical analysis, we propose the Online Discrepancy Distance Learning enabled by a memory buffer sampling (ODDL-S) model, which trades off between the model's complexity and performance. The memory buffer sampling mechanism ensures the information diversity learning. The proposed theoretical analysis provides new insights into the model's forgetting behaviour during each training step of TFCL. Experimental results on several TFCL benchmarks show that the proposed ODDL-S achieves state-of-the-art performance.

References

[1] R. Aljundi, M. Lin, B. Goujaud, and Y. Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems (NeurIPS), arXiv preprint arXiv:1903.08671*, 2019.

[2] Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. In *Advances in Neural Information Processing Systems (NeurIPS), arXiv preprint arXiv:1908.04742*, 2019.

[3] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proc. of IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pages 11254–11263, 2019.

[4] Verónica Álvarez, Santiago Mazuelas, and José Antonio Lozano. Minimax classification under concept drift with multidimensional adaptation and performance guarantees. In *Proc. International Conference on Machine Learning (ICML), vol. PMLR 162*, pages 486–499, 2022.

[5] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *Proc. of IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 8218–8227, 2021.

[6] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. Dokania, P. H. S. Torr, and M.'A. Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

[7] W. Dai, Q. Yang, G. R. Xue, and Y. Yu. Boosting for transfer learning. In *Proc. Int Conf. on Machine Learning (ICML)*, pages 193–200, 2007.

[8] Matthias De Lange and Tinne Tuytelaars. Continual prototype evolution: Learning online from non-stationary data streams. In *Proc. of the IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, pages 8250–8259, 2021.

[9] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. In *arXiv preprint arXiv:1312.6211*, 2014.

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

[11] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *Proc. NIPS Deep Learning Workshop, arXiv preprint arXiv:1503.02531*, 2014.

[12] Xisen Jin, Arka Sadhu, Junyi Du, and Xiang Ren. Gradient-based editing of memory examples for online task-free continual learning. In *Advances in Neural Information Processing Systems (NeurIPS), arXiv preprint arXiv:2006.15294*, 2021.

[13] H. Jung, J. Ju, M. Jung, and J. Kim. Less-forgetting learning in deep neural networks. In *Proc. AAAI Conf. on Artificial Intelligence*, volume 32, pages 3358–3365, 2018.

[14] Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. Achieving forgetting prevention and knowledge transfer in continual learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[15] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

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

[17] Jeremias Knoblauch, Hisham Husain, and Tom Diethe. Optimal continual learning has perfect memory and is NP-hard. In *Proc. Int. Conf. on Machine Learning (ICML), vol PMLR 119*, pages 5327–5337, 2020.

[18] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Univ. of Toronto, 2009.

[19] Richard Kurle, Botond Cseke, Alexej Klushyn, Patrick van der Smagt, and Stephan Günnemann. Continual learning with Bayesian neural networks for non-stationary data. In *Int. Conf. on Learning Representations (ICLR)*, 2020.

[20] Sampo Kuutti, Richard Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):712–733, 2020.

[21] Ya Le and Xuan Yang. Tiny imageNet visual recognition challenge. Technical report, Univ. of Stanford, 2015.

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

[23] Sebastian Lee, Sebastian Goldt, and Andrew Saxe. Continual learning in the teacher-student setup: Impact of task similarity. In *Proc. Int. Conf. on Machine Learning (ICML), vol. PMLR 139*, pages 6109–6119, 2021.

[24] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee 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.

[25] Z. Li and D. Hoiem. Learning without forgetting. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2017.

[26] Sen Lin, Li Yang, Deliang Fan, and Junshan Zhang. TRGP: Trust region gradient projection for continual learning. In *Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:2202.02931*, 2022.

[27] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pages 6467–6476, 2017.

[28] Divyam Madaan, Jaehong Yoon, Yuanchun Li, Yunxin Liu, and Sung Ju Hwang. Representational continuity for unsupervised continual learning. In *Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:2110.06976*, 2022.

[29] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *Proc. Conf. on Learning Theory (COLT), arXiv preprint arXiv:2002.06715*, 2009.

[30] Mehryar Mohri and Andres Munoz Medina. New analysis and algorithm for learning with drifting distributions. In *International Conference on Algorithmic Learning Theory*, pages 124–138. Springer, 2012.

[31] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. In *Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:1710.10628*, 2018.

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

[33] Anastasia Pentina and Christoph H Lampert. Lifelong learning with non-iid tasks. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 1540–1548, 2015.

[34] R. Polikar, L. Upda, S. S. Upda, and Vasant Honavar. Learn++: An incremental learning algorithm for supervised neural networks. *IEEE Trans. on Systems Man and Cybernetics, Part C*, 31(4):497–508, 2001.

[35] Mozhgan PourKeshavarzi, Guoying Zhao, and Mohammad Sabokrou. Looking back on learned experiences for class/task incremental learning. In *Int. Conf. on Learning Representations (ICLR)*, 2022.

[36] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. GDumb: A simple approach that questions our progress in continual learning. In *Proc. European Conference on Computer Vision (ECCV), vol. LNCS 12347*, pages 524–540, 2020.

[37] Krishnan Raghavan and Prasanna Balaprakash. Formalizing the generalization-forgetting trade-off in continual learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[38] Dushyant Rao, Francesco Visin, Andrei A. Rusu, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Continual unsupervised representation learning. In *Proc. Neural Inf. Proc. Systems (NIPS)*, pages 7645–7655, 2019.

[39] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. iCaRL: Incremental classifier and representation learning. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2001–2010, 2017.

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

[41] Hippolyt Ritter, Aleksandar Botev, and David Barber. Online structured Laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pages 3742–3752, 2018.

[42] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems 34 (NeurIPS)*, pages 348–358, 2019.

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

[44] Shengyang Sun, Daniele Calandriello, Huiyi Hu, Ang Li, and Michalis Titsias. Information-theoretic online memory selection for continual learning. In *Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:2204.04763*, 2022.

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

[46] Fei Ye and Adrian G. Bors. Learning latent representations across multiple data domains using lifelong VAEGAN. In *Proc. European Conf. on Computer Vision (ECCV), vol. LNCS 12365*, pages 777–795, 2020.

[47] Fei Ye and Adrian G. Bors. Lifelong learning of interpretable image representations. In *Proc. Int. Conf. on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6, 2020.

[48] Fei Ye and Adrian G Bors. Mixtures of variational autoencoders. In *Proc. Int. Conf. on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6, 2020.

[49] Fei Ye and Adrian G. Bors. InfoVAEGAN: Learning joint interpretable representations by information maximization and maximum likelihood. In *Proc. IEEE Int. Conf. on Image Processing (ICIP)*, pages 749–753, 2021.

[50] Fei Ye and Adrian G Bors. Learning joint latent representations based on information maximization. *Information Sciences*, 567:216–236, 2021.

[51] Fei Ye and Adrian G. Bors. Lifelong infinite mixture model based on knowledge-driven Dirichlet process. In *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*, pages 10695–10704, 2021.

[52] Fei Ye and Adrian G. Bors. Lifelong mixture of variational autoencoders. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–14, 2021.

[53] Fei Ye and Adrian G. Bors. Lifelong twin generative adversarial networks. In *Proc. IEEE Int. Conf. on Image Processing (ICIP)*, pages 1289–1293, 2021.

[54] Fei Ye and Adrian G Bors. Continual variational autoencoder learning via online cooperative memorization. *arXiv preprint arXiv:2207.10131*, 2022.

[55] Fei Ye and Adrian G. Bors. Deep mixture generative autoencoders. *IEEE Transactions on Neural Networks and Learning Systems*, 33(10):5789–5803, 2022.

[56] Fei Ye and Adrian G Bors. Learning an evolved mixture model for task-free continual learning. *arXiv preprint arXiv:2207.05080*, 2022.

[57] Fei Ye and Adrian G. Bors. Lifelong generative modelling using dynamic expansion graph model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(8):8857–8865, Jun. 2022.

[58] Fei Ye and Adrian G. Bors. Lifelong teacher-student network learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6280–6296, 2022.

[59] Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. In *Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:2106.01085*, 2022.

[60] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. Bridging theory and algorithm for domain adaptation. In *Proc. International Conference on Machine Learning (ICML), vol. PMLR 97*, pages 7404–7413, 2019.

[61] Shengjia Zhao, Jiaming Song, and Stefano Ermon. InfoVAE: Balancing learning and inference in variational autoencoders. In *Proc. AAAI Conf. on Artif. Intel.*, volume 33, pages 5885–5892, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]