Task-Free Continual Learning via Online Discrepancy Distance Learning

Anonymous Author(s)
Affiliation
Address
email

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 there are some recently proposed algorithms for TFCL, these methods lack theoretical guarantees. Moreover, there are no theoretical studies for forgetting analysis of TFCL. This paper develops a new theoretical analysis framework that derives generalization bounds based on the discrepancy distance between the visited samples and the entire information made available for training the model. This analysis provides new insights into the forgetting behaviour in classification tasks. Inspired by this theoretical model, we propose a new approach enabled with the dynamic component expansion mechanism for a mixture model, namely Online Discrepancy Distance Learning (ODDL). ODDL estimates the discrepancy between the current memory and the already accumulated knowledge 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 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 maintaining the knowledge of past experiences [31]. Such an ability can be used in many real-time applications such as in robotics, health investigative systems, for autonomous vehicles [19] or an agent exploring artificial (meta) universes, which require 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 [12].

One of the popular attempts 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 [5, 1]. However, when there are restrictions on the available memory capacity, the memory-based approach would still suffer from degenerated performance on past tasks, especially when aiming to learn an infinite number of tasks. Recently, the Dynamic Expansion Model (DEM) [47] has shown promising results in CL, aiming to guarantee optimal performance by preserving the previously learnt knowledge into the parameters of frozen components trained on past data, while adding a new component when learning a novel task. However, these approaches require knowing where and when the knowledge associated with a given task is changed, which is not always applicable in real environments.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

In this paper, we address a more realistic scenario, called Task-Free Continual Learning (TFCL) [3], where the 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 [11, 7]. However, such an approach requires carefully designing 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 [5]. This issue can be relieved by DEM in which previously learnt samples are preserved into frozen components and do not have any effect when learning new samples [36, 23]. 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 [47, 45], NP-hard problem [16], Teacher-Student framework [48, 22] and game theory [35]. However they require strong assumptions, such as defining the task identities, and such information 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 new theoretical classification framework, which provides insights for 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 the forgetting as a generalization error in the domain adaptation theory. Based on this analysis we extend the domain adaptation theory [28] 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) algorithm that 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. We list our contributions as : 1) To our best knowledge, 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. 2) Inspired by the theoretical analysis, we develop a novel dynamic expansion approach, which ensures a compact network architecture and an optimal performance. 3) We propose a new sample selection approach that selects appropriate samples for the memory, further improving performance. 4) We achieve the 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 [6, 10, 12, 15, 18, 24, 30, 32, 38, 39, 25, 33], where the network's parameters that 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, 43, 4, 34, 27, 42, 50] to store a few past data or trains a generator to provide the replay samples when learning new tasks [36, 41, 49, 48, 45, 46]. However, these approaches usually rely on 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. Using a small memory buffer to store past samples has shown benefits for TFCL and was firstly investigated in [3]. This memory replay approach was then extended by employing Generative Replay Mechanism (GRMs) for training both the classifier and Variational Autoencoder (VAEs) [14], 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) [7] 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 Gradient

based Memory EDiting (GMED) [11], which can be employed in the existing CL models to further enhance their performance.

Dynamic expansion model aims 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) [36], dynamically builds new inference models when meeting the expansion criterion. However, since CURL still requires GRMs 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 for adding new components while freezing all previously learnt members, in a model called the Continual Neural Dirichlet Process Mixture (CN-DPM), [23]. However, the expansion criterion of these approaches relies on the change of the loss while 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 analyse the forgetting behaviour of a single as well as for a dynamic expansion model by deriving the Generalization Bounds (GB).

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 $\{1, 2, \ldots, n'\}, n' > 2$ for multi-class classification. Let $\mathcal{D}_i^T = \{\mathbf{x}_j^T, y_i^T\}_{j=1}^{N_i^T}$ and $\mathcal{D}_i^S = \{\mathbf{x}_j^S, y_i^S\}_{j=1}^{N_i^S}$ 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 distribution 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 [28] to measure this gap through the analysis of the model's risk.

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

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

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

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 Rademacher complexity which is used in the domain adaptation theory as a measure of richness for a particular hypothesis space [51, 29].

Corollary 1. [28] For two given domains $\mathbb{P}_{t,j}^{T,\mathcal{X}}$ and $\mathbb{P}_{t,j}^{S,\mathcal{X}}$, let $U_{\mathcal{P}}$ and $U_{\mathbb{P}}$ represent sample sets of sizes $m_{\mathcal{P}}$ and $m_{\mathbb{P}}$, drawn independently from $\mathbb{P}_{t,j}^{T,\mathcal{X}}$ and $\mathbb{P}_{t,j}^{S,\mathcal{X}}$. Let $\widehat{\mathbb{P}}_{t,j}^{T,\mathcal{X}}$ and $\widehat{\mathbb{P}}_{t,j}^{S,\mathcal{X}}$ 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 probability $1-\delta$, we have :

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

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 (See details in Appendix-B from SM). Let $\mathcal{L}_{\widehat{d}}(\mathbb{P}_{t,j}^{T,\mathcal{X}}, \mathbb{P}_{t,j}^{S,\mathcal{X}})$ 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 $\forall(y, y') \in \mathcal{Y}^2, \mathcal{L}(y, y') \leq U'$, 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 :

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

where $\eta\big(\mathcal{P}_i, \mathbb{P}_{\mathcal{M}_i}\big)$ is the optimal combined error $\mathcal{R}(h_{\mathcal{P}_i}, h_{\mathcal{M}_i}, \mathcal{P}_i) + \mathcal{R}(h_{\mathcal{P}_i}, h_{\mathcal{P}_i}^{\star}, \mathcal{P}_i)$ 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 $h_{\mathcal{P}_i}^{\star}$ is the true labeling function for $\mathcal{P}_i$.

The proof is provided in Appendix-A from the Supplementary Material (SM). Compared to the GB used in the domain adaptation [28], 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 $\mathcal{L}_{\widehat{d}}\big(\mathcal{P}_i^{\mathcal{X}}, \mathbb{P}_{\mathcal{M}_i}^{\mathcal{X}}\big)$ 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 $\mathbb{P}_{t,j}^{T}$, we derive the GB for a model at the training step $\mathcal{T}_i$:

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

The proof is similar to that for Theorem 1. We can observe that the generalization performance on a target domain $\mathbb{P}_{t,j}^{T}$, by a model h is relying mainly on the discrepancy distance between $\mathbb{P}_{t,j}^{T,\mathcal{X}}$ 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 $S = \bigcup_{j=1}^{C_t^S} \mathcal{D}_{t,j}^{S}$ consisting of samples from $\mathcal{D}_t^S$, let $\mathcal{D}_t^T$ be the corresponding testing set and $\mathbb{P}_{t,j}^{T}$ represent the distribution of samples for the j-th category from $\mathcal{D}_t^T$, we derive the GB for multiple target domains as:

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

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) [16] 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 [5, 7] at each training step without requiring any task information (See more details in Appendix-D from SM). 3) [47] 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 [47].

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 [26] in which the performance of the model is decreased due to samples being drawn from entirely different distributions [13]. 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:

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

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

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

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. To compare with a single model (See Theorem 3), DEM can achieve the minimal upper bound to the risk (LHS of Eq. (6)). 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 :

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

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 [36, 23]. 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 :

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

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 the next section, we introduce a new CL framework according to the theoretical analysis.

4 Methodology

Firstly, we 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.

4.1 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, during the training, by using the discrepancy distance (Some notations are defined in **Theorem 3**), expressed as :

$$\mathcal{M}^\star=\underset{\mathcal{M}_i}{\arg\max}\sum\nolimits_j^{\mathrm{c}}\big\{\mathcal{L}_{\widehat{d}}(\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_{k_j}},\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i})\big\}\,, \tag{12}$$

where $i=k_c{+}1,\cdots,n$ represents the index of the training steps and k_c represents the c-th component trained at $\mathcal{T}_{k_c}$.$\mathcal{M}^\star$ is the optimal memory. Eq. (12) 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. (12) while balancing the model's complexity and performance, we derive an expansion criterion based on the discrepancy distance :

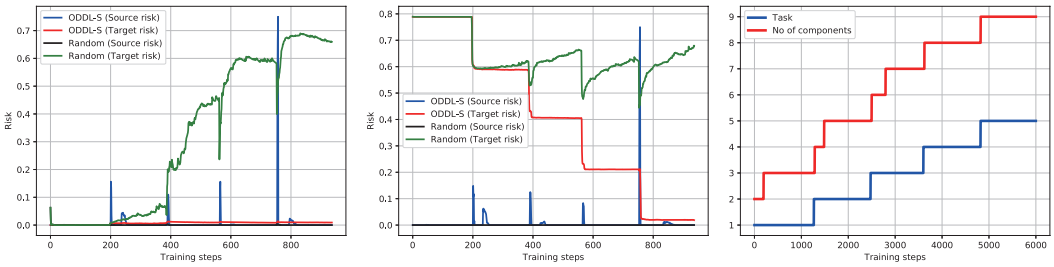
$$\min\Big\{\mathcal{L}_{\widehat{d}}\big(\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_{k_j}},\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}\big)\mid j=1,\cdots,c\Big\}\ge\lambda\,, \tag{13}$$

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. (13)), $\mathcal{G}$ 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 detailed 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. (13) at $\mathcal{T}_i$:

$$\mathcal{L}^\star_d\big(\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_{k_1}},\mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}\big)\ge\lambda\,. \tag{14}$$

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 $\mathbb{P}^{\mathcal{X}}_{v^j_1}$ formed by samples drawn from v^j_1 of G^j_1, where the superscript j represents the first component finishing the training at $\mathcal{T}_j$. $\mathcal{L}^\star_d(\cdot,\cdot)$ is the estimator of the discrepancy



(a) Risk on the training set. (b) Risk on all target sets. (c) Results on MNIST.

Figure 1: 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).

distance, achieved by h_e^i and h_1^j.

$$\mathcal{L}_d^\star(\mathbb{P}^{\mathcal{X}}_{v_1^j}, \mathbb{P}_{\mathcal{M}_i}) \triangleq \left| \mathbb{E}_{\mathbf{x}\sim\mathbb{P}^{\mathcal{X}}_{v_1^j}} \left[\mathcal{L}\big(h_1^j(\mathbf{x}), h_e^i(\mathbf{x})\big) \right] - \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_{\mathcal{M}_i}} \left[\mathcal{L}\big(h_1^j(\mathbf{x}), h_e^i(\mathbf{x})\big) \right] \right|. \tag{15}$$

If satisfies Eq. (14), 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 $G_e^i \to G_e^{i+1}$ 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 learn non-overlapping distributions during the following training steps. We can also easily extend Eq. (14) to the expansion criterion for $\mathcal{G} = \{G_1, \cdots, G_{s-1}\}$ that has already preserved $(s-1)$ components, illustrated in Fig. 2:

$$\min \left\{ \mathcal{L}_d^\star(\mathbb{P}^{\mathcal{X}}_{v_j^{k_j}}, \mathbb{P}^{\mathcal{X}}_{\mathcal{M}_i}) \,|\, j = 1, \cdots, s-1 \right\} \geq \lambda \,, \tag{16}$$

where $\mathbb{P}^{\mathcal{X}}_{v_j^{k_j}}$ is the distribution of the generated samples by $G_j^{k_j}$.

4.2 Sample selection

As discussed in Lemma 2, the components' probabilistic diversity, according to their intrinsic accumulated knowledge in the mixture model is crucial for the model's performance. We introduce a novel sample selection approach that further encourages component diversity. The primary motivation behind the proposed sample selection approach is that we desire to store those samples that are completely different from the knowledge characterizing the previously trained 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 :

$$\mathcal{L}_d^s(\mathcal{G}, \mathbf{x}_c) \triangleq \frac{1}{s} \sum\nolimits_{t=1}^{s} \left| \mathcal{L}(h_t^{k_t}(\mathbf{x}_c), h_e^i(\mathbf{x}_c)) - \mathcal{L}(h_t^{k_t}(\mathbf{x}_t^{k_t}), h_e^i(\mathbf{x}_t^{k_t})) \right| , \tag{17}$$

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. (17) evaluates the average discrepancy distance between the knowledge generated by each trained component and the stored sample, which guides for the sample selection at the training step ($\mathcal{T}_i$) as :

$$\mathcal{M}_i = \bigcup_{j=1}^{|\mathcal{M}_i'|-b} \mathcal{M}_i'[j] \,, \tag{18}$$

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.

4.3 Algorithm and implementation

Implementation. 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 VAE model implemented by an encoding network enc_{ω_j} as well as the corresponding decoding network dec_{θ_j}, with trainable parameters $\{\omega_j, \theta_j\}$. The loss

function for the j-th component at $\mathcal{T}_i$ is defined as :

$$\mathcal{L}_{class}(G_j, \mathcal{M}_i) \triangleq \frac{1}{|\mathcal{M}_i|} \sum_{t=1}^{|\mathcal{M}_i|} \left\{ \mathcal{L}_{ce}\big(h_j(\mathbf{x}_t), y_t\big) \right\}, \tag{19}$$

$$\mathcal{L}_{VAE}(G_j, \mathcal{M}_i) \triangleq \mathbb{E}_{q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x})} \left[\log p_{\theta_j}(\mathbf{x}_t\,|\,\mathbf{z})\right] - D_{KL}\left[q_{\omega_j}(\mathbf{z}\,|\,\mathbf{x}_t)\,||\,p(\mathbf{z})\right], \tag{20}$$

where $\{\mathbf{x}_t, y_t\} \sim \mathcal{M}_i$ and $\mathcal{L}_{ce}(\cdot)$ is the cross-entropy loss. $p_{\theta_j}(\mathbf{x}_t\,|\,\mathbf{z})$ 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. $\mathcal{L}_{VAE}(\cdot,\cdot)$ is the VAE loss [14]). We also implement $\hat{f}(\cdot,\cdot)$ in Eq. (9) by $-\mathcal{L}_{VAE}(\cdot,\cdot)$ for the component selection at the testing phase.

Algorithm. We provide the pseudocode in Appendix-E from SM. The algorithm has three main stages : 1) **(Initial training).** We start to build 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. (20) and Eq. (19) 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. 2) **(Evaluator training).** In this stage, we only update the Evaluator using Eq. (20) and Eq. (19) 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. (15) to check the expansion (Eq. (16)). 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. 3) **(Sample selection).** We evaluate the diversity score for each stored sample in $\mathcal{M}_{j+1}$ using Eq. (17). We then perform the sample selection for $\mathcal{M}_{j+1}$ using Eq. (18).

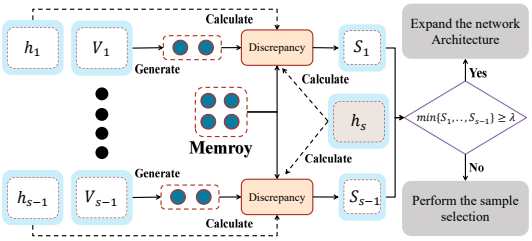


Figure 2: The evaluation of the proposed expansion criterion and sample selection. The knowledge generated by each component, is used to evaluate the discrepancy distance (Eq. (15)) with the memory. Then the expansion criterion (Eq. (16)) would perform either the expansion or the sample selection.

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 [7] which employs several datasets including Split MNIST [21], Split CIFRA10 [17] and Split CIFRA100 [17]. The the detailed information for datasets, hyperparameters and network architectures is provided in Appendix-F from SM.

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) and the discrepancy distance. We plot the results in Fig. 1-a, where "Random (Source risk)" represents the source risk of the 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

Table 1: The accuracy of five independent runs for various models on three datasets.

Methods	Split MNIST	Split CIFRA10	Split CIFRA100
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
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

target risk and the discrepancy distance are increased, which is caused by the memory that loses previous samples, theoretically explained in Theorem 1. We also evaluate the risk on all testing sets (target risk) and plot the results in Fig. 1-b where it is observed that the 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. 1. 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. 1-a. Finally, the risk on all target distributions (all categories in the testing set) from each training step is shown in Fig. 1-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 [7] and [11], respectively. We compare with several baselines including: finetune that directly trains a classifier on the data stream, GSS [1], MIR [2], Gradient Episodic Memory (GEM) [26], Incremental Classifier and Representation Learning (iCARL) [37], Reservoir [44], CURL, CNDPM, CoPE, ER + GMED and ER_a + GMED [11] where ER is the Experience Replay [40] and ER_a is ER with data augmentation. The number of components in ODDL-S and ODDL for Split MNIST, Split CIFRA10 and Split CIFRA100 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 [20]), and Permuted MNIST [8]. We split MINI-ImageNet into 20 tasks (See details into Appendix-F1 from SM), namely Split MImageNet. We follow the setting from [2] where the maximum memory size is $10K$, and a slim version of ResNet-18 [9] is used as the classifier. The hyperparameter λ for Split MINI-ImageNet and Permuted MNIST is 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 [11]. 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. 1c 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 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 the maximize memory bound in Appendix-F2.2 from SM and more ablation study results are provided in Appendix-F.2 from SM.

6 Conclusion

In this paper, we develop a novel theoretical framework for TFCL, based on defining a discrepancy distance. The proposed theoretical analysis provides new insights into the model's forgetting behaviour during each training step. Inspired by the theoretical analysis, we propose a novel dynamic expansion approach, namely ODDL-S, which trades off between the model's complexity and performance. Finally, we test on several TFCL benchmarks where 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] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8218–8227, 2021.

[5] 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.

[6] 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.

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

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

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

[15] 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.

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

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

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

[19] 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.

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

[21] 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.

[22] Sebastian Lee, Sebastian Goldt, and Andrew Saxe. Continual learning in the teacher-student setup: Impact of task similarity. In *International Conference on Machine Learning*, pages 6109–6119. PMLR, 2021.

[23] 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.

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

[25] Sen Lin, Li Yang, Deliang Fan, and Junshan Zhang. TRGP: Trust region gradient projection for continual learning. In *International Conference on Learning Representations*, 2022.

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

[27] Divyam Madaan, Jaehong Yoon, Yuanchun Li, Yunxin Liu, and Sung Ju Hwang. Representational continuity for unsupervised continual learning. In *International Conference on Learning Representations*, 2022.

[28] 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.

[29] 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.

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

[31] 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.

[32] 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.

[33] Mozhgan PourKeshavarzi, Guoying Zhao, and Mohammad Sabokrou. Looking back on learned experiences for class/task incremental learning. In *International Conference on Learning Representations*, 2022.

[34] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *European conference on computer vision*, pages 524–540. Springer, 2020.

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

[36] 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.

[37] 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.

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

[39] 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.

[40] 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.

[41] 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.

[42] Shengyang Sun, Daniele Calandriello, Huiyi Hu, Ang Li, and Michalis Titsias. Information-theoretic online memory selection for continual learning. In *International Conference on Learning Representations*, 2022.

[43] Michalis K Titsias, Jonathan Schwarz, Alexander G de G Matthews, Razvan Pascanu, and Yee Whye Teh. Functional regularisation for continual learning with Gaussian processes. In *Proc. Int. Conf. on Learning Represenations (ICLR), arXiv preprint arXiv:1901.11356*, 2019.

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

[45] 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.

[46] 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.

[47] 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.

[48] Fei Ye and Adrian G. Bors. Lifelong teacher-student network learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

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

[50] Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. In *International Conference on Learning Representations*, 2022.

[51] 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.

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]