Article:
Ye, Fei and Bors, Adrian Gheorghe (2025) Continual Unsupervised Generative Modeling. IEEE Transactions on Pattern Analysis and Machine Intelligence. pp. 6256-6273. ISSN: 0162-8828

https://doi.org/10.1109/TPAMI.2025.3564188

This article has been accepted for publication in IEEE Transactions on Pattern Analysis and Machine Intelligence. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2025.3564188

Continual Unsupervised Generative Modelling

Fei Ye and Adrian G. Bors, *IEEE Senior Member*
Department of Computer Science, University of York, York YO10 5GH, UK
E-mail: fy689@york.ac.uk, adrian.bors@york.ac.uk

Abstract—Variational Autoencoders (VAEs), based on their capabilities, have achieved remarkable results in modelling individual tasks, such as data representations, image generation and image-to-image translation. However, VAEs suffer from loss of information when aiming to continuously learn a sequence of different data domains. Such losses are caused by the catastrophic forgetting, which affects all machine learning methods, both classical as well as deep learning. This paper addresses the problem of catastrophic forgetting by developing a new theoretical framework which derives an upper bound to the negative sample log-likelihood when continuously learning sequences of tasks. This theoretical derivation provides new insights into the forgetting behaviour of networks, showing that optimal performance is achieved when a dynamic mixture expansion model adds new components whenever learning new tasks. In our approach we optimize the model size by introducing the Dynamic Expansion Graph Model (DEGM) that dynamically builds a graph structure promoting the positive knowledge transfer when learning new tasks. In addition, we propose a Dynamic Expansion Graph Adaptive Mechanism (DEGAM) that generates adaptive weights to regulate the graph structure, further improving the positive knowledge transfer effectiveness. Experimental results show that the proposed methodology performs better than other baselines in continual learning.

Index Terms—Variational Autoencoder, Continual learning, Unsupervised generative modelling, Dynamic expansion model, Graph Neural Network

1 Introduction

Lifelong/Continual Learning (LLL/CL) is a fundamental trait for living beings, either humans or animals, that enables them to survive in a dynamically changing environment. Lately, LLL has evolved into modern computer models that can learn dynamically changing data distributions without forgetting. However, one obstacle in applying these models to real applications is catastrophic forgetting [1], which is associated with the degradation of the model's performance on previously learnt data whenever learning new tasks. This is caused mostly by the re-writing of the model's parameters following new training leading to catastrophic forgetting.

A popular approach in LLL to counteract forgetting, called Generative Replay Mechanism (GRM) [2], consists in training a generator, implemented by a Variational Autoencoder (VAE) [3] or a Generative Adversarial Net (GAN) [4], to generate previously learned samples. Catastrophic forgetting in GRM is mitigated by re-training the model on its own generated data in a self-supervising manner [5], [6], [7], [8], [9]. However, the quality of generative replay samples is critical to the model's performance. In addition, most GRM-based approaches still suffer from forgetting when learning an increasing number of tasks due to the information knowledge degradation following repeated generative replay processes [10]. Moreover, GANs suffer from mode collapse [11], resulting in hallucinations in data, and can not provide reasonable past samples if it learns several completely different data domains in a sequential manner [12]. In recent years, several attempts have been focused on the dynamic expansion mechanism [13] and the ensemble structure [14], [15], [16], where a new module/component is built onto a shared network backbone to learn given new tasks. These approaches can relieve forgetting by freezing the weights of all previously trained components while only updating the weights of the newly created component. However, such approaches mainly focus on the classification task and do not provide theoretical guarantees for LLL.

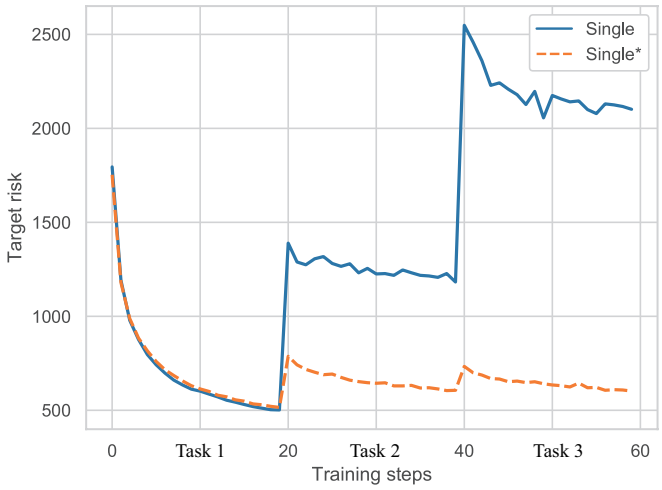


Fig. 1. The target risk for a single VAE model under the MNIST (Task 1), Fashion (Rask 2), IFashion (Task 3) lifelong learning, when considering that the model is trained on real samples from previous tasks (Single*) and when using the data produced through GRMs (Single).

In this paper, we focus on unsupervised generative modelling in the context of lifelong learning. We study the data likelihood estimation under the LLL/CL as a measure for capturing underlying data representations [17]. A VAE [3] is made up of an encoder and a decoder whose main objective function is to maximize the sample log-likelihood $\log p(\mathbf{x}) = \log \int p(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z}) \mathrm{d}\mathbf{z}$. The VAE objective function, called the Evidence Lower Bound (ELBO) [3], employs a

variational distribution $q(\mathbf{z} \mid \mathbf{x})$ which approximates the true posterior. VAEs are popular likelihood models which have acquired remarkable successes in learning unsupervised tasks by inferring probabilistic data representations [18], for image reconstruction tasks [19], [20], [21], [22], or for disentangled representation learning [17], [22]. In order to address catastrophic forgetting we consider Generative Replay Mechanisms (GRM) for reproducing the data learnt in the past. An empirical example of how lifelong learning works in VAEs is provided in Fig. 1 where "Single" and "Single*" represent a single VAE model trained using data produced through GRMs and the same model which accesses the real training samples from previous tasks, respectively. The target risk is evaluated as the performance on real data from three databases: MNIST, Fashion and Inverse Fashion. It can be observed from Fig. 1, that the target risk for a single VAE model trained through GRMs, when learning more tasks, gradually loses its performance on real data distributions.

Inspired by the empirical results discussed above, this paper aims to study the tightness between the data likelihood $\log p(\mathbf{x})$ and the Evidence Lower Bound (ELBO) for a single model $\mathcal{L}_{ELBO}(\mathbf{x}')$, where $\mathbf{x}'$ is made up by combining generated data, associated with data learnt by the model in the past and real data. We propose a new theoretical framework that derives an upper bound on the negative marginal log-likelihood, called Lifelong ELBO (LELBO). The LELBO analysis considers the discrepancy distance [23] between the target and evolved source distributions, the accumulated errors caused by the task switch, as well as other error terms. This analysis provides insights into how a VAE model loses previously learned knowledge during LLL. We also generalize the proposed theoretical analysis to the Dynamic Expansion Models (DEMs). The DEM model gradually adds a new component when a new task is learned, while freezing all previously learnt components to preserve the previously learnt information. DEM can achieve optimal performance while it requires significant computational resources. To address this problem, we introduce the Dynamic Expansion Graph Model (DEGM), which dynamically builds a graph structure which reuses most of the parameters and the currently learnt information by the previously trained components, when learning a new task. Specifically, we consider two types of processing components in DEGM : base and specific nodes. The former can be considered as an independent VAE component, aiming to learn a completely different task. Meanwhile, the latter has fewer independent parameters and relies on the information from the base nodes. This is performed through a graph structure which connects the specific node to all existing base nodes. To regulate the knowledge transfer when learning a new task, we propose a new knowledge assessment approach which calculates the weight of each base node with respect to the probabilistic representation of the new task according to their respective knowledge relevance. These weights are then used to regulate latent representations during inference and decoding processes, effectively reusing previously learned information for learning a new task. Moreover, to further explore the benefit of knowledge transfer, we propose a dynamic expansion adaptive mechanism that optimizes the weights to adapt to the new task. The proposed mechanism dynamically generates new weights when expanding the model while freezing all previously learnt weights to preserve as much as required from the existing structure. We evaluate the performance of the proposed model on multiple datasets, while the empirical results show that the proposed model achieves the best performance with fewer parameters when compared with other baselines.

Our contributions are as follows :

- We develop a novel theoretical framework for analyzing VAE's forgetting behaviour during LLL.
- We extend the proposed theoretical analysis to other generative models, providing new insights into the forgetting behaviour.
- We develop a novel dynamic expansion model which guarantees the trade-off between the optimal performance for each task and the model's complexity during lifelong learning.
- We propose a dynamic expansion adaptive mechanism to regulate the information flow from previously learnt components when learning a new task, which maximizes the benefit from knowledge transfer.
- We propose a new benchmark for the probability density estimation task under the LLL setting.
- Experimental results when learning several density estimation and image generation tasks show that the proposed model achieves better performance than other baselines while using an optimal network architecture.

The rest of the paper is organised as in the following. In Section 2 we discuss other lifelong learning approaches. In Section 3 we introduce a series of definitions and foundational concepts for the proposed methodology. The theoretical framework is described in Section 4. Then, the proposed methodology is detailed in Section 5. The experiments are described in Section 6 and the conclusions in Section 7.

2 Related work

In the following after discussing the variational autoencoder (VAE) we review the main approaches in LLL/CL learning.

2.1 The Variational Autoencoder

The objective function of a VAE, also called Evidence Lower Bound (ELBO) [3], is defined as :

$$\begin{aligned}\log p_\theta(\mathbf{x}) \geq & \mathbb{E}_{q_\omega(\mathbf{z}|\mathbf{x})}\left[\log p_\theta\left(\mathbf{x} \mid \mathbf{z}\right)\right] \\ & - KL\left[q_\omega\left(\mathbf{z} \mid \mathbf{x}\right) \mid\mid p\left(\mathbf{z}\right)\right] := \mathcal{L}_{ELBO}\left(\mathbf{x}; \{\theta, \omega\}\right) ,\end{aligned} \tag{1}$$

where $p_\theta(\mathbf{x} \mid \mathbf{z})$ and $p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ are the decoding and prior distribution, respectively. $KL[\cdot]$ denotes the Kullback–Leibler divergence and $\mathbf{I}$ is the identity matrix, while ω and θ denote the parameters of the encoder and decoder, respectively. The tightness of the VAE objective function (ELBO) is crucial for improving the performance of a VAE. One of the possible approaches is to use the Importance Weighted Autoencoder (IWELBO) [24], which generates a set of weighted samples for the given input and results in a tighter ELBO, depending on how many samples are used. Other approaches aim to use more informative approximate posterior distributions, such as normalizing flows [25], [26],

implicit distributions [27] or hierarchical variational inference [28]. Moreover, these approaches can further improve the performance by integrating IWELBO loss into their primary objective function. In addition, online variation inference [29] has been used in the VAE framework, requiring to store a subset of training samples for computing the approximate posterior, which is impractical when learning an unlimited number of tasks. Moreover, the tightness of ELBO under the lifelong learning (LLL) has not been investigated in any of these works. This paper provides the first theoretical analysis for the forgetting behaviour in VAEs under the LLL. In the following we present the two main approaches for LLL/CL employing VAEs and other deep learning models by using memory buffers and dynamic architectures.

2.2 Memory-based Continual Learning

One goal in the LLL/CL is to enable a model to prevent forgetting during the training. A natural approach to this goal is to build a memory-based replay system, which can be implemented by using either a memory buffer to store past training samples or by training a Generative Replay Mechanism (GRM) network. The former approach collects a subset of training samples for each task and replays them when learning a new task [30], [31]. The memory-based approaches can also be combined with regularisation methods [32], [33], [34], [35], [36], [37], [38], [39], using memorized samples to regularise the optimization of the model such that the network parameters deemed important for the past tasks are not changed much when learning a new task. In [39] weights corresponding to past tasks are kept in the memory using a hyper-network regularizer. However, such approaches would suffer from an ever-increasing computational complexity burden as the number of tasks increases [35]. Moreover, the memory buffer usually has a fixed capacity, which is not scalable when learning an infinite number of tasks [12]. The GRM-based approaches aim to train a generative model such as a Generative Adversarial Network (GAN) [4] or a Variational Autoencoder (VAE) [3] which would then be used to reproduce data similar to those from the training set. A GRM is used in a self-supervising manner in which generative replay samples are employed to relieve forgetting [2], [5], [6], [40]. In this paper, we mainly focus on the GRMs as it does not require storing past data.

Shin *et al.* in the first study considering GRMs for lifelong learning classification [2], employed a GAN for producing generative replay samples, which are then integrated with the newly given training samples in order to learn the whole system. However, the main drawback of this approach is that it can only be used for classification and cannot learn underlying data representations due to lacking an inference mechanism. The Variational Autoencoder with Shared Embeddings (VASE) employs the Minimum Description Length (MDL) principle to induce disentangled representations over time, [5]. Meanwhile, the VAE-based LLL model was extended to a teacher-student framework in the Lifelong Generative Modelling (LGM) [6], where a teacher and a student network teach each other in turns in order to progressively accumulate knowledge. Then, VAE-based models were further enhanced by using a hybrid model integrating the power generation ability of GANs and the inference mechanisms of VAEs into a unified optimization framework, called Lifelong VAEGAN [10]. The advantage of this approach over LGM is that Lifelong VAEGAN can produce better generative replay samples, through its GAN component, empowered by a variational autoencoder, which then can be used in more complex datasets reducing forgetting. More recently, VAE-based models have been applied in disentangled representation learning [5], [10], which can learn meaningful latent variables across multiple domains under the LLL. However, these models depend on the generative capabilities of generators, which usually produce fuzzy images when learning a long sequence of tasks. Moreover, these models can not handle an unlimited number of tasks due to the fixed model capacity while employing repeated GRM processes.

2.3 Dynamic Architectures

Dynamic expansion LLL models aim to enhance the model's capacity by gradually increasing the number of parameters or by adding layers of hidden units to adapt to a growing number of tasks [13], [41], [42], [43], [44], [45], [46], [47], [48], [49]. Lee *et al.* [13] employs Dirichlet processes as a criterion for expanding the number of components in the mixture model. However, this method also requires a small memory buffer to prevent the frequent expansion of the network architecture. Meanwhile, the Continual Unsupervised Representation Learning (CURL) [40], integrates a GRM-based approach with the dynamic expansion mechanisms into a unified system. CURL only expands the network architecture for the inference model while the decoder is continually trained on the GRM's samples together with the training samples from the memory buffer. CURL still suffers from forgetting due to the repeated decoder updating.

The Continual Adaptation Modules for Generative Adversarial Networks (CAM-GAN) [50] employs dynamic GANs to enable image generation under the lifelong learning. The main idea of this approach is to learn a set of global and task-specific parameters in which the latter aims to capture specific information for each task. Although the model from [50] uses transfer learning by initializing parameters from a selected task-specific module, it can not fully explore the positive knowledge transfer from all prior tasks. In contrast, the approach proposed in this paper can account for the information flow from all base nodes that have learnt many tasks in the past and regulate it through a dynamic expansion adaptive mechanism, which can maximize the benefit of information transfer learning. In addition, learning global and task-specific parameters for GAN models have also been discussed in [51], [52]. Different from [50], [51], [52], which are based on the GAN framework, the proposed approach aims to learn meaningful latent representations for the data, which can implement image reconstruction and density estimation tasks. Furthermore, this paper introduces a new theoretical framework for analyzing the forgetting behaviour of lifelong learning models, which is lacking in other works [50], [51], [52].

Besides the dynamic expansion approaches, there are other types of models, such as [14], [16], [20], [53], which usually have multi-head network architectures where several task-specific modules are built on top of a common

module. During training, the shared module is only updated during the learning of the initial tasks and then serves as a backbone for other additionally created task-specific modules. These approaches would guarantee full performance on past tasks if the number of tasks is known [16]. However, they are not scalable when there is an increasing number of tasks to learn. This paper addresses a more general lifelong learning situation where the task label is given during the training, but where the total number of tasks is not known.

3 Foundational definitions and concepts

3.1 Problem Statement

In this paper, we study a general lifelong unsupervised learning problem, where we know task boundaries, but we consider that class labels are not available during training. Let $\mathcal{X} \in \mathbf{R}^d$ represent the data space, where d is the dimension of the space. Let $\{\mathcal{T}_1, \ldots, \mathcal{T}_N\}$ be a set of tasks, where each task $\mathcal{T}_i$ is associated with an unlabeled testing set $D_i^T \in \mathcal{X}$ and an unlabeled training set $D_i^S \in \mathcal{X}$. N represents the total number of tasks. During the learning of a certain task $\mathcal{T}_i$, a model only accesses the samples from D_i^S while all previously seen tasks are not available. When all tasks have been completed, we evaluate the model on all testing sets $\{D_1^T, \ldots, D_N^T\}$. In the following, we introduce some necessary notations and definitions.

3.2 Notations

Since this study focuses on generative modelling tasks, we introduce a loss function used to evaluate image reconstruction quality.

Definition 1. (**Loss function.**) Let consider a hypothesis function $\{h \in \mathcal{H} \mid \mathcal{H} : \mathcal{X} \to \mathcal{X}\}$, where $\mathcal{H}$ is the space of all hypotheses. Let $\mathcal{L} = \|\mathbf{x} - h(\mathbf{x})\|^2$ represent a loss function implemented by the Square Loss (SL) such that $\mathcal{L} = \sum_{i=1}^{d} (\mathbf{x}[i] - h(\mathbf{x})[i])^2$, where $[i]$ represents the entry for the i-th dimension.

In the following, we define the probability representation for each task's training and testing sets.

Definition 2. (**Probability representation.**) Let $\tilde{\mathcal{P}}_i$ and $\mathcal{P}_i$ be the probability distributions for the training D_i^S and testing D_i^T set, respectively.

Definition 3. (**Single model.**) Let $\mathcal{A} = \{f_\omega, g_\theta\}$ represent a single generative latent variable (VAE) model that has an encoder $f_\omega \colon \mathcal{X} \to \mathcal{Z}$ parameterized by ω and a decoder $g_\theta : \mathcal{Z} \to \mathcal{X}$ parameterized by θ. f_ω and g_θ are used to model the encoding $q_\omega(\mathbf{z} \mid \mathbf{x})$ and decoding distribution $p_\theta(\mathbf{x} \mid \mathbf{z})$ of a VAE model, respectively. To ensure the differentiable optimization during the encoding-decoding process, we employ the reparameterization trick [3] for the sampling procedure $\mathbf{z} = f_\omega^\mu(\mathbf{x}) + f_\omega^\delta(\mathbf{x}) \odot \gamma$, $\gamma \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ where $f_\omega^\mu(\mathbf{x})$ and $f_\omega^\delta(\mathbf{x})$ are the mean and standard deviation vectors of a Gaussian distribution, implemented by a neural network $f_\omega(\mathbf{x})$. We consider the notation $\mathcal{A}^t$ for the model trained on t tasks and $\{\omega^t, \theta^t\}$ are the associated parameters of $\mathcal{A}^t$. Let $g_\theta(f_\omega) \colon \mathcal{X} \to \mathcal{X}$ denote the encoding-decoding procedure for $\mathcal{A}$, and $\mathbb{P}^t$ be the probability distribution formed by samples drawn from the generator of the model $\mathcal{A}^t$.

Based on this definition of the model we define the model risk in the following.

Definition 4. (**Model risk**.) We consider implementing h by the encoding-decoding process $g_\theta(f_\omega)$. Let $\mathcal{L} \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ represent a loss function. We consider that $\mathcal{L}$ is bounded, $\forall(\mathbf{x}, \mathbf{x}') \in \mathcal{X}^2, \mathcal{L}(\mathbf{x}, \mathbf{x}') \leq U$ for some $U > 0$. We implement $\mathcal{L}$ as the Square Loss (SL) function $\mathcal{L}(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|^2, (\mathbf{x}, \mathbf{x}') \in \mathcal{X}$. The risk for $h(\cdot)$ on the target domain $\mathcal{P}_i$ of the i-th task is defined as :

$$\mathcal{E}_{\mathcal{P}_i}(h, f_{\mathcal{P}_i}) = \mathbb{E}_{\mathbf{x} \sim \mathcal{P}_i} \mathcal{L}(h(\mathbf{x}), f_{\mathcal{P}_i}(\mathbf{x}))\,. \tag{2}$$

where $f_{\mathcal{P}_i}(\mathbf{x})$ is an identity function for the sample $\mathbf{x}$ drawn from $\mathcal{P}_i$.

3.3 Evaluating Probabilistic Distances Between Distributions

In the following, we define the discrepancy distance for evaluating the similarity between two probabilistic representations.

Definition 5. (**Discrepancy distance [23].**) Let $\mathcal{P}_i$ and $\mathbb{P}_i$ be two distributions over $\mathcal{X}$. We define the discrepancy distance between $\mathcal{P}_i$ and $\mathbb{P}_i$ using the loss function $\mathcal{L}$:

$$\begin{aligned}\mathcal{L}_{\text{disc}}(\mathcal{P}_i, \mathbb{P}_i) = \sup_{h, h' \in \mathcal{H}} \big| &\mathbb{E}_{\mathbf{x} \sim \mathcal{P}_i}\left[\mathcal{L}\left(h'(\mathbf{x}), h(\mathbf{x})\right)\right] \\ &- \mathbb{E}_{\mathbf{x} \sim \mathbb{P}_i}\left[\mathcal{L}\left(h'(\mathbf{x}), h(\mathbf{x})\right)\right]\big|,\end{aligned} \tag{3}$$

where h and h' are two different hypotheses.

Definition 6. **(Rademacher complexity [23].)** Let $\mathcal{H}$ represent a hypothesis class. For a given unlabeled data batch $S = \{\mathbf{x}_i\}_{i=1}^m$, the Rademacher complexity of $\mathcal{H}$ with respect to S is defined as :

$$\text{Re}_S(\mathcal{H}) \triangleq \mathop{\mathbb{E}}_{\mathcal{H}}\left[\sup_{h \in \mathcal{H}} \frac{2}{m} \sum_{i=1}^{m} \mathcal{H}_i h(\mathbf{x}_i)\right], \tag{4}$$

where $\langle_i \sim \mathcal{U}[-1, 1]$ is a uniform random variable and $\mathcal{H} = \{\mathcal{H}_1, \cdots, \mathcal{H}_m\}$. The Rademacher complexity for the whole hypotheses space is defined as :

$$\text{Re}_m(\mathcal{H}) \triangleq \mathbb{E}_{S \sim (D)^m} \text{Re}_S(\mathcal{H})\,. \tag{5}$$

where D is an underlying data distribution. Based on the definition of the Rademacher complexity, we provide a practical way to estimate the discrepancy distance from a given finite sample set.

Definition 7. (**Empirical discrepancy distance [23].**) Let $U_{\mathcal{P}_i}$ and $U_{\mathbb{P}_i}$ be the sample populations of size $m_{\mathcal{P}_i}$ and $m_{\mathbb{P}_i}$ from the empirical distributions $\hat{\mathcal{P}}_i$ and $\hat{\mathbb{P}}_i$, respectively. Then, we can estimate the discrepancy distance by using the finite samples :

$$\begin{aligned}\mathcal{L}_{\text{disc}}(\mathcal{P}_i, \mathbb{P}_i) \leq\ & disc_{\mathcal{L}}(\hat{\mathcal{P}}_i, \hat{\mathbb{P}}_i) + 8\big(\text{Re}_{U_{\mathcal{P}_i}}(\mathcal{H}) \\ &+ \text{Re}_{U_{\mathbb{P}_i}}(\mathcal{H})\big) \\ &+ 3M\left(\sqrt{\frac{\log\left(\frac{4}{\delta}\right)}{2m_{\mathcal{P}}}} + \sqrt{\frac{\log\left(\frac{4}{\delta}\right)}{2m_{\mathbb{P}}}}\right).\end{aligned} \tag{6}$$

Eq. (6) holds with probability $1 - \delta, \delta \in (0, 1)$ and $M \in \mathbb{R}_+$ is a positive number.

In the following we use $\mathcal{L}^\star_{\text{disc}}(\cdot)$ to represent the right-hand side (RHS) of Eq. (6).

TABLE 1
The description of many important notations.

Notation	Description
$\tilde{\mathcal{P}}_i$	The probability distribution for the training dataset D_i^S.
$\mathcal{P}_i$	The probability distribution for the testing dataset D_i^S.
$\mathbb{P}^t$	The probability distribution of samples drawn from $\mathcal{A}^t$.
$\mathcal{A}^t$	The single VAE model updated at the t-th task learning.
$\mathcal{A}_i$	The i-th component in the mixture model.
$\mathbf{A} = \{\mathcal{A}_1, \cdots, \mathcal{A}_K\}$	The dynamic expansion model with K components.
$I_{\mathcal{T}}$	The true task-inference model.
$\mathcal{L}^\star_{\text{disc}}(\cdot)$	The empirical discrepancy distance.
$\mathcal{L}$	The square loss function.
$f_{\mathcal{P}_i}(\mathbf{x})$	The identity function for the sample $\mathbf{x}$ drawn from $\mathcal{P}_i$.

4 Theoretical Framework

In this section, we introduce a new theoretical framework using the discrepancy distance between probabilistic representations for assessing the novelty of the information when compared with the current knowledge of the model. This framework aims to analyze the forgetting behaviour of the generative model while continuously learning for defining the conditions for the model's architecture expansion ensuring the lifelong learning success. We list some of the significant notations used in this paper in Table. 1.

4.1 Analysis for A Single Model

Let us consider a VAE model $\mathcal{A}^i$ to be trained on a sequence of training sets $\{D_1^S, \ldots, D_i^S\}$. The generator $g_{\theta^i}(\cdot)$ of $\mathcal{A}^i$ can produce a large number of samples and let $\mathbb{P}^i$ be the probabilistic representation distribution for the generated replay samples. Let $\tilde{\mathcal{P}}_i$ to represent the probabilistic representation of D_i^S. A single model $\mathcal{A}^i$ usually adopts GRM to relieve forgetting when learning the $(i+1)$-th task, in which the past data samples represented by $\mathbb{P}^i$ is used to retrain the model. Therefore, such a learning process can be formulated as a recursive optimization problem :

$$\mathcal{A}^{i+1} = \underset{\omega^{(i+1)}, \theta^{(i+1)}}{\arg\min}\ \mathcal{L}^\star\left(\mathbb{P}^{i+1}, \mathbb{P}^i \otimes \tilde{\mathcal{P}}_{i+1}\right), \tag{7}$$

where $\mathcal{L}^\star(\cdot)$ is the loss function used for training the model, usually implemented by the negative ELBO. $\mathbb{P}^i \otimes \tilde{\mathcal{P}}_{i+1}$, represents the mixing distribution formed by samples uniformly drawn from both $\mathbb{P}^i$ and $\tilde{\mathcal{P}}_{i+1}$, respectively. As i increases in Eq. (7), the model learns more tasks and would suffer from forgetting. The learning goal of $\mathcal{A}^{i+1}$ at the $(i+1)$-th task learning is to approximate the distribution $\mathbb{P}^{i+1} \approx \mathbb{P}^i \otimes \tilde{\mathcal{P}}_{i+1}$ by minimizing the objective function $\mathcal{L}^\star(\cdot)$. During lifelong learning, the model accumulates more errors caused by the generative replay procedure, leading to a degenerated performance on its corresponding unseen domain D_i^T. One indicator for the generalization ability of a model $\mathcal{A}$ is to predict its performance on a testing data set by achieving a certain error rate on a training data set, [54]. In this paper, we provide, for the first time, the theoretical analysis assessing the generalization of a model under LLL, where the source distribution changes over time. Initially, we introduce the Generalization Bound (GB) for the ELBO for a single VAE model when learning a single task in Theorem 1 and then extend this to learning several tasks in Theorem 2.

Theorem 1. Let $\mathcal{P}_i$ and $\tilde{\mathcal{P}}_i$ be two distributions over $\mathcal{X}$. Let $h^*_{\mathcal{P}_i} = \arg\min_{h\in\mathcal{H}} \mathcal{R}_{\mathcal{P}_i}(h, f_{\mathcal{P}_i})$ and $h^*_{\tilde{\mathcal{P}}_i} = \arg\min_{h\in\mathcal{H}} \mathcal{R}_{\tilde{\mathcal{P}}_i}(h, f_{\tilde{\mathcal{P}}_i})$ be the two optimal models, where $f_{\mathcal{P}_i}, f_{\tilde{\mathcal{P}}_i} \in \mathcal{H}$ denote the identity functions under the encoding-decoding process for $\mathcal{P}_i$ and $\tilde{\mathcal{P}}_i$, respectively. We define a GB between $\mathcal{P}_i$ and $\tilde{\mathcal{P}}_i$ as :

$$\begin{aligned}\mathcal{E}_{\mathcal{P}_i}(h, f_{\mathcal{P}_i}) \le \mathcal{E}_{\tilde{\mathcal{P}}_i}(h, h^*_{\tilde{\mathcal{P}}_i}) + \mathcal{L}^\star_{\text{disc}}(\mathcal{P}_i, \tilde{\mathcal{P}}_i) \\ + \mathcal{E}_{\mathcal{P}_i}(h^*_{\mathcal{P}_i}, f_{\mathcal{P}_i}) + \mathcal{E}_{\mathcal{P}_i}(h^*_{\mathcal{P}_i}, h^*_{\tilde{\mathcal{P}}_i}),\end{aligned} \tag{8}$$

where we consider the following notation :

$$\varepsilon(\mathcal{P}_i, \tilde{\mathcal{P}}_i) = \mathcal{E}_{\mathcal{P}_i}(h^*_{\mathcal{P}_i}, f_{\mathcal{P}_i}) + \mathcal{E}_{\mathcal{P}_i}(h^*_{\mathcal{P}_i}, h^*_{\tilde{\mathcal{P}}_i}), \tag{9}$$

representing the optimal combined risk. We also have the model risk :

$$\mathcal{E}_{\tilde{\mathcal{P}}_i}(h, h^*_{\tilde{\mathcal{P}}_i}) = \mathbb{E}_{\mathbf{x}\sim\tilde{\mathcal{P}}_i} \mathcal{L}\big(h(\mathbf{x}), h^*_{\tilde{\mathcal{P}}_i}(\mathbf{x})\big). \tag{10}$$

The proof is provided in Appendix-A from the Supplementary Material (SM). We use $\mathcal{E}_R(\mathcal{P}_i, \tilde{\mathcal{P}}_i)$ to represent $\mathcal{L}^\star_{\text{disc}}(\mathcal{P}_i, \tilde{\mathcal{P}}_i) + \varepsilon(\mathcal{P}_i, \tilde{\mathcal{P}}_i)$. Theorem 1 explicitly evaluates the generalization error of a single model $\mathcal{A}$ trained on the source distribution $\tilde{\mathcal{P}}_i$. By using the conclusions of Theorem 1, we can generalize this GB to all previously learnt tasks in the following.

Theorem 2. Let $\{\mathcal{T}_1, \ldots, \mathcal{T}_t\}$ be a sequence of tasks that have already been trained. We derive a GB for a single model between the target distribution and the evolved source distribution during the t-th task learning :

$$\begin{aligned}\frac{1}{t}\sum_{i=1}^{t} \mathcal{E}_{\mathcal{P}_i}(h, f_{\mathcal{P}_i}) \le \mathcal{E}_{\mathbb{P}^{t-1}\otimes\tilde{\mathcal{P}}_t}(h, h^*_{\mathbb{P}^{t-1}\otimes\tilde{\mathcal{P}}_t}) \\ + \mathcal{E}_R(\mathcal{P}_{(1:t)}, \mathbb{P}^{t-1}\otimes\tilde{\mathcal{P}}_t),\end{aligned} \tag{11}$$

where $\mathcal{P}_{(1:t)}$ is the mixture distribution $\{\mathcal{P}_1 \otimes \mathcal{P}_2, \ldots, \otimes\mathcal{P}_t\}$.

The proof for Theorem 2 is provided in Appendix-B from SM.

Remark. Theorem 2 has the following observations:

- The generalization performance on the target distributions relies on the discrepancy distance term. Therefore, minimizing the source risk can not guarantee a tight GB for Eq. (11).
- When using GRM, $\mathbb{P}^{t-1}$ is gradually degenerated as t increases due to the repeated retraining on generated data [10], which enlarges the discrepancy distance term. This explains the forgetting process of a single model when learning a long sequence of tasks.

In Appendix-J from SM, we show that Theorem 2 can be generalized to other generative models, such as for example GANs, demonstrating that the discrepancy distance between the target and the generator's distribution is key to the generalization performance of GANs under lifelong learning, exhibiting similar forgetting behaviour to VAEs. In the following, we generalize this GB to the negative ELBO $\mathcal{L}^\star$.

Lemma 1. Let us consider the random samples $\mathbf{x}_i^T \sim \mathcal{P}_i$, for $i = 1, \ldots, t$. The sample log-likelihood and its ELBO for all $\{\mathcal{P}_1, \ldots, \mathcal{P}_t\}$ can be represented by $\sum_{i=1}^{t} \log p_\theta(\mathbf{x}_i^T)$ and $\sum_{i=1}^{t} \mathcal{L}_{ELBO}(h, \mathbf{x}_i^T)$. Let $\tilde{\mathbf{x}}^t$ represent the random sample drawn from $\mathbb{P}^{t-1} \otimes \tilde{\mathcal{P}}_t$. We know that $KL(q_{\omega^t}(\mathbf{z} \mid \mathbf{x}_i^T) \mid\mid p(\mathbf{z})) \neq KL(q_{\omega^t}(\mathbf{z} \mid \tilde{\mathbf{x}}^t) \mid\mid p(\mathbf{z}))$ if $q_{\omega^t}(\mathbf{z} \mid \mathbf{x}_i^T) \neq q_{\omega^t}(\mathbf{z} \mid \tilde{\mathbf{x}}^t)$, and we have :

$$\begin{aligned} &\frac{1}{t}\sum_{i=1}^{t} \mathbb{E}_{\mathcal{P}_i} KL\big(q_{\omega^t}(\mathbf{z} \mid \mathbf{x}_i^T) \mid\mid p(\mathbf{z})\big) \leq \\ &\mathbb{E}_{\mathbb{P}^{t-1} \otimes \tilde{\mathcal{P}}_t} KL\big(q_{\omega^t}(\mathbf{z} \mid \tilde{\mathbf{x}}^t) \mid\mid p(\mathbf{z})\big) + \big|KL_1 - KL_2\big|, \end{aligned} \tag{12}$$

where $q_{\omega^t}(\cdot)$ represents the inference model for $\mathcal{A}^t$. KL_1 and KL_2 represent the Left-Hand Side (LHS) term and the first term of the RHS of Eq. (12), respectively. We also know that ELBO contains a negative reconstruction error term, a KL divergence term and a constant $(-\frac{1}{2}\log\pi)$ [55], if the decoder in VAEs is to model a Gaussian distribution with a diagonal covariance matrix (the diagonal element is $1/\sqrt{2}$). We then derive a GB on -ELBO by combining Eq. (11) and Eq. (12) :

$$\begin{aligned} &\frac{1}{t}\sum_{i=1}^{t} \mathbb{E}_{\mathcal{P}_i}\Big[-\mathcal{L}_{ELBO}\big(\mathbf{x}_i^T; h\big)\Big] \leq \mathcal{E}_R\big(\mathcal{P}_{(1:t)}, \mathbb{P}^{t-1} \otimes \tilde{\mathcal{P}}_t\big) \\ &+ \mathbb{E}_{\mathbb{P}^{t-1} \otimes \tilde{\mathcal{P}}_t}\Big[-\mathcal{L}_{ELBO}\big(\tilde{\mathbf{x}}^t; h\big)\Big] + \big|KL_1 - KL_2\big|, \end{aligned} \tag{13}$$

where $\mathbf{x}_i^T \sim \mathcal{P}_i$ and $\tilde{\mathbf{x}}^t \sim \mathbb{P}^{t-1} \otimes \tilde{\mathcal{P}}_t$.

The proof is provided in Appendix-C from SM. We call the RHS of Eq. (13) as the Lifelong ELBO (LELBO), denoted as $\mathcal{L}_{LELBO}$, which can measure the information loss of a single model in an infinite number of tasks ($t \to \infty$). As the number of tasks (t) increases, the discrepancy distance term in Eq. (13) becomes larger due to the repeated GRM processes, leading to a degenerated performance on the target distributions. We also can show that $\mathcal{L}_{LELBO}$ is an upper bound to the data log-likelihood $-\sum_{i=1}^{t} \mathbb{E}_{\mathcal{P}_i}\left[\log p(\mathbf{x}_i^T)\right]/t$, estimated by $\mathcal{A}^t$.

We also extend the GB from Eq. (13) when considering the Importance Weighted VAE (IWELBO), whose objective function is defined as :

$$\mathcal{L}_{ELBO_{K'}}(\mathbf{x}; \mathcal{A}) = \mathbb{E}_{\mathbf{z}_1, .., \mathbf{z}_{K'} \sim q(\mathbf{z}|\mathbf{x})}\left[\log \frac{1}{K'}\sum_{i=1}^{K'} w_i\right], \tag{14}$$

where w_i is defined as :

$$w_i = \frac{p\left(\tilde{\mathbf{x}}^t, \mathbf{z}_i\right)}{q\left(\mathbf{z}_i \mid \mathbf{x}\right)}, \tag{15}$$

where K' represents the number of importance weighted samples $\{\mathbf{z}_1, \ldots, \mathbf{z}_{K'}\}$ [56]. $\mathcal{L}_{ELBO_{K'}}(\mathbf{x}; \mathcal{A})$ is tighter than $\mathcal{L}_{ELBO}(\mathbf{x}; \mathcal{A})$ and we have $\mathcal{L}_{ELBO_{K'}}(\mathbf{x}; \mathcal{A}) \geq \mathcal{L}_{ELBO}(\mathbf{x}; \mathcal{A})$, where $K' > 1$, [56]. Then we generalize the IWELBO bounds to the LLL setting as:

$$\begin{aligned} &\frac{1}{t}\sum_{i=1}^{t} \mathbb{E}_{\mathbf{x}_i^T \sim \mathcal{P}_i}\left[-\log p\left(\mathbf{x}_i^T\right)\right] \leq \\ &\mathbb{E}_{\tilde{\mathbf{x}}^t \sim \mathbb{P}^{t-1} \otimes \tilde{\mathcal{P}}_t}\left[-\mathbb{E}_{\mathbf{z}_1 \ldots, \mathbf{z}_{K'} \sim q(\mathbf{z}|\mathbf{x})}\left[\log \frac{1}{K'}\sum_{i=1}^{K'} w_i\right]\right] \\ &+ |KL_1 - KL_2| + \mathcal{E}_R\left(\mathcal{P}_{(1:t)}, \mathbb{P}^{t-1} \otimes \tilde{\mathcal{P}}_t\right). \end{aligned} \tag{16}$$

The derivation is provided in Appendix-G from SM. We omit the subscript for $q(\cdot)$. We call the RHS of Eq. (16) as $\mathcal{L}_{LELBO_{K'}}$, and $\mathcal{L}_{LELBO_{K'=1}} = \mathcal{L}_{LELBO}$.

Remark. We have several conclusions from Lemma 1 :

- If $|KL_1 - KL_2| = 0$ and $\mathbb{P}^{t-1}$ is fixed, then we have $\mathcal{L}_{LELBO_{K'+1}} \leq \mathcal{L}_{LELBO_{K'}}$.
- A large ELBO on the source distribution $\mathbb{P}^{t-1} \otimes \tilde{\mathcal{P}}_t$ (the second term of RHS from Eq. (16)) can not guarantee a tight bound on the testing data log-likelihood since the RHS of Eq. (16) contains the discrepancy distance term and other error terms.

When considering Lemma 1, we note that a tight GB is mainly based on the minimum discrepancy distance term, which can be obtained by training a powerful generator that accurately approximates the target domains, *e.g.* by using the Autoencoding VAE [57] or a GAN [4]. However, these approaches would fail when learning multiple completely different tasks and cannot handle an infinite number of tasks due to their limited capacity and the mode collapse problem [11]. The following section shows that this issue can be addressed by increasing the model's capacity through an architecture expansion mechanism.

4.2 Analysis for the Dynamic Expansion Model

In this section, we extend the theoretical analysis to the dynamic expansion model.

Definition 8. (**Dynamic expansion model.**) Let $\mathbf{A} = \{\mathcal{A}_1, \ldots, \mathcal{A}_K\}$ be a dynamic expansion model, where each component $\mathcal{A}_i$ is trained by means of Generative Replay Mechanisms (GRMs) for learning new tasks.

Definition 9. (**Probabilistic representations for tasks.**) Let us define $\mathbb{P}^{(i,s)}$ as the distribution of the samples generated by the s-th component ($\mathcal{A}_s$) after it was trained on a number of i tasks. We assume that a certain j-th task was learnt by $\mathcal{A}_s$. We can define an approximation distribution $\mathbb{P}_j^{(m,s)}$ for the j-th task following the sampling $\mathbf{x} \sim \mathbb{P}^{(i,s)}$, if $I_{\mathcal{T}}(\mathbf{x}) = j$, where $I_{\mathcal{T}} \colon \mathcal{X} \to \mathcal{T}$ is the task-inference function that returns the true task label for the sample $\mathbf{x}$. We use m in $\mathbb{P}_j^{(m,s)}$ to represent that GRM was used m times on $\mathcal{A}_s$ for learning the j-th task. For the sake of simplifying the notation, we omit the component index s for $\mathbb{P}_j^{(m,s)}$. Let $\mathbb{P}_t^0$ represent $\tilde{\mathcal{P}}_t$.

Based on the above notations and definitions, we investigate the forgetting behaviour of the dynamic expansion model $\mathbf{A}$ by developing a new GB that allows $\mathbf{A}$ to have any number of components K.

Theorem 3. Let $C = \{c_1, \ldots, c_m\}$ represent a set, where each item c_i indicates that the c_i-th component ($\mathcal{A}_{c_i}^1$) is only trained once during LLL. We use $A = \{a_1, \ldots, a_m\}$ to represent the task label set for C, where a_i is associated to c_i. Let $C' = \{c_1', \ldots, c_k'\}$ represent a set where c_i' indicates that the c_i'-th component $\mathcal{A}_{c_i'}$ is trained more than once and is associated with a task label set $A'_{c_i'} = \{a(i,1), \ldots, a(i,n)\}$. Let $\tilde{C} = \{c(i,1), \ldots, c(i,n)\}$ be a set where $c(i,j)$ denotes the number of times $\mathcal{M}_{c_i'}$ was used for $a(i,j)$-th task. We have $|C| + |C'| = K$, $|A'_{c_i'}| > 1$, where K is the number of components in the mixture model and $|\cdot|$ is the cardinality of a set. Let

$\tilde{A} = \{\tilde{a}_1, \ldots, \tilde{a}_k\}$ represent a set where each $\tilde{a}_i$ denotes the number of tasks modelled by the probabilistic representations of the c_i'-th component $\tilde{a}_i = |A'_{c_i'}|$. We derive the bound for $\mathbf{A}$ during the t-th task learning :

$$\frac{1}{t}\sum_{i=1}^{|C'|}\Big\{\sum_{j=1}^{\tilde{a}_i}\Big\{\mathcal{E}_{\mathcal{P}_{a(i,j)}}\Big(h_{c_i'}, f_{\mathcal{P}_{a(i,j)}}\Big)\Big\}\Big\}+ \frac{1}{t}\sum_{i=1}^{|C|}\Big\{\mathcal{R}_{\mathcal{E}_{a_i}}\Big(h_{c_i}, f_{\mathcal{P}_{a_i}}\Big)\Big\} \leq \frac{1}{t}\mathcal{E}_C + \frac{1}{t}\mathcal{E}_{R'}\,, \tag{17}$$

where each $h_{c_i} \in \mathcal{H}$ and $h_{c_i'} \in \mathcal{H}$ represent the hypothesis of the c_i-th and c_i'-th component in the mixture, respectively. $\mathcal{E}_C$ is the error evaluated by the components that are trained only once :

$$\mathcal{E}_C = \sum_{i=1}^{|C|}\Big\{\mathcal{E}_{\tilde{\mathcal{P}}_{a_i}}\Big(h_{c_i}, h^*_{\tilde{\mathcal{P}}_{a_i}}\Big) + \mathcal{E}_R\Big(\mathcal{P}_{a_i}, \tilde{\mathcal{P}}_{a_i}\Big)\Big\}, \tag{18}$$

and $\mathcal{E}_{R'}$ is the accumulated error evaluated by the components that are trained more than once :

$$\mathcal{E}_{R'} = \sum_{i=1}^{|C'|}\Big\{\sum_{j=1}^{\tilde{a}_i}\{\mathcal{E}_{\mathbb{P}^{c(i,j)}_{a(i,j)}}\Big(h_{c_i'}, h^*_{\mathbb{P}^{c(i,j)}_{a(i,j)}}\Big) + \mathcal{E}_R\Big(\mathcal{P}_{a(i,j)}, \mathbb{P}^{c(i,j)}_{a(i,j)}\Big)\Big\}\}, \tag{19}$$

and after decomposing the last term it becomes

$$\mathcal{E}_{R'} = \sum_{i=1}^{|C'|}\Big\{\sum_{j=1}^{\tilde{a}_i}\Big\{\mathcal{E}_{\mathbb{P}^{c(i,j)}_{a(i,j)}}\Big(h_{c_i'}, h^*_{\mathbb{P}^{c(i,j)}_{a(i,j)}}\Big) + \sum_{k=-1}^{c(i,j)-1}\Big\{\mathcal{E}_R\Big(\mathbb{P}^{k}_{a(i,j)}, \mathbb{P}^{k+1}_{a(i,j)}\Big)\Big\}\Big\}\Big\}. \tag{20}$$

The proof is provided in Appendix-D from SM.

Remark. We have several observations from **Theorem 3** :

- The dynamic expansion model has only a single component if and only if $|C'| = 1$ and $|C| = 0$. Then the RHS of Eq. (17) does not include the term $\mathcal{E}_C$ while $\mathcal{E}_{R'}$ becomes large, leading to more accumulated error terms according to Eq. (20).
- The dynamic expansion model leads to a large network where the number of components is equal to the number of tasks if $|C| = t$. Then RHS of Eq. (17) becomes $\frac{1}{t}\mathcal{E}_C$, which has no accumulated error terms and leads to a tight GB.
- As $|C|$ increases, the model improves its generalization performance. We show that $|C|$ is related to the number of components K, according to the accumulated error term $|C'| = K - |C|$ in Eq. (20), which decreases while K increases.
- Suppose that we only have learnt a single component for learning multiple tasks ($|C'| = 1$). The GB of a single component for the initial tasks ($a(i,j)$ is small), tends to accumulate more errors when compared to the GB of a single component for the latest given tasks ($a(i,j)$ is large), demonstrated by the number of accumulated error terms $\mathcal{E}_R(\cdot,\cdot)$ in Eq. (20), controlled by $c(i,j) = t - a(i,j)$.

In the following, we extend GB from the square loss function $\mathcal{L}$ to the empirical discrepancy distance $\mathcal{L}^\star$.

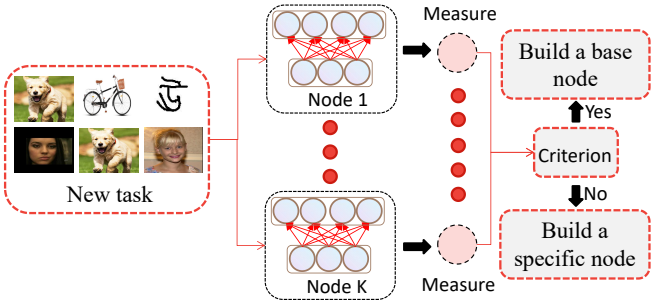


Fig. 2. The dynamic expansion process of the proposed DEGM. When seeing a new task, each node estimates its corresponding sample log-likelihood and compares the previous sample log-likelihood using Eq. (24), which is used to check the model expansion. If Eq. (25) is stratified, we then build a base node, otherwise, we build a specific node.

Lemma 2. We derive a GB for the marginal log-likelihood during the t-th task learning for the expanded model :

$$\frac{1}{t}\sum_{i=1}^{t}\mathbb{E}_{\mathcal{P}_i}\Big[-\log p(\mathbf{x}_i^T)\Big] \leq \frac{1}{t}\Big(\mathcal{R}_{R'}^{II} + \mathcal{R}_C^{II} + D^\star_{diff}\Big)+ \frac{1}{t}\sum_{i=1}^{|C'|}\Big\{\sum_{j=1}^{\tilde{a}_i}\Big\{\mathbb{E}_{\mathbb{P}^{c(i,j)}_{a(i,j)}}\Big[-\mathcal{L}_{ELBO}\Big(\mathbf{x}^t_{a(i,j)}; h_{c_i'}\Big)\Big]\Big\} + \sum_{i=1}^{|C|}\Big\{\mathbb{E}_{\tilde{\mathcal{P}}_{a_i}}\Big[-\mathcal{L}_{ELBO}(\mathbf{x}^S_{a_i}; h_{c_i})\Big]\Big\}\Big\}, \tag{21}$$

where $D_{diff}*$ and D_{diff} are defined as :

$$D_{diff}* = \sum_{i=1}^{|C|}\Big\{D_{diff}\Big(\mathbf{x}^T_{a_i}, \mathbf{x}^S_{a_i}\Big)\Big\} + \sum_{i=1}^{|C'|}\sum_{j=1}^{\tilde{a}_i}\Big\{D_{diff}\Big(\mathbf{x}^T_{a(i,j)}, \mathbf{x}^t_{a(i,j)}\Big)\Big\}, \tag{22}$$

$$D_{diff}(\mathbf{x}^T_{a_i}, \mathbf{x}^S_{a_i}) = \Big|\mathbb{E}_{\mathcal{P}_{a_i}} KL(q_{\omega_{c_i}}(\mathbf{z} \mid \mathbf{x}_i^T) \,||\, p(\mathbf{z})) - \mathbb{E}_{\tilde{\mathcal{P}}_{a_i}} KL(q_{\omega_{c_i}}(\mathbf{z} \mid \mathbf{x}_i^S) \,||\, p(\mathbf{z}))\Big|. \tag{23}$$

$\mathcal{R}_{R'}^{II}$ and $\mathcal{R}_C^{II}$ are the second terms in the RHS of Eq. (18) and Eq. (19), respectively. We omit the component's index for each $\log p(\mathbf{x}_i^T)$ for the sake of simplification. Each variable $\mathbf{x}^S_{a_i}$ is drawn from the distribution $\tilde{\mathcal{P}}_{a_i}$ and each $\mathbf{x}^t_{a(i,j)}$ is drawn from $\mathbb{P}^{c(i,j)}_{a(i,j)}$ modelled by the c_i'-th component of $\mathbf{A}$. $\mathcal{L}_{ELBO}(\mathbf{x}^S_{a_i}; h_{c_i})$ is the ELBO estimated by the c_i-th component.

Lemma 2 is proved in Appendix-E from SM. As a consequence of Lemma 2, we can measure the gap between the ELBO and the model's likelihood across all target distributions using the dynamic expansion model. Eq. (21) can be tight if and only if $|C| = 1$, $D^\star_{diff} = 0$ and the discrepancy $\mathcal{L}^\star_{\text{disc}}(\mathcal{P}_{a_i}, \mathbb{P}^0_{a_i})$ is very small. In addition, we extend the proposed theoretical framework to analyze the forgetting behaviour of other deep generative models in Appendix-J from SM.

5 The Dynamic Expansion Graph Model

Following the analysis provided by Theorem 3, an optimal GB can be achieved by employing a mixture model and enabling the training of each mixture component with a certain task. However, this learning procedure would lead to

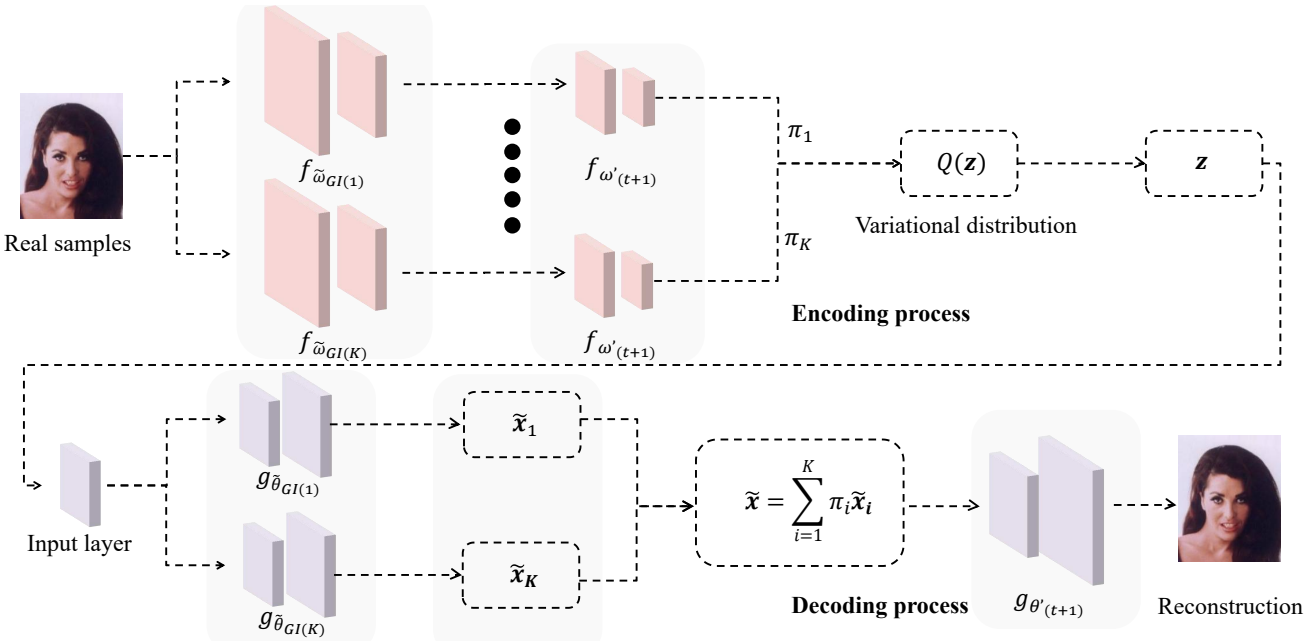


Fig. 3. The graph structure when building a specific node in the Dynamic Expansion Graph Model (DEGM). First, we use the sub-encoders from all base nodes to connect the sub-encoder of the newly created specific node. During the inference procedure, we take an image as the input to these combined modules which form a variational distribution $Q(\mathbf{z})$. The latent variable $\mathbf{z}$ is drawn from $Q(\mathbf{z})$ and then fed into the input layer. At the decoding process, we take the output of the input layer as the input to the sub-decoder of all base nodes, resulting in the variables $\{\widetilde{\mathbf{x}}_1, \cdots, \widetilde{\mathbf{x}}_K\}$. Then these, weighted by $\pi_i, i = 1, \ldots, K$ are summed up, resulting in $\widetilde{\mathbf{x}}$, which is fed to the sub-decoder of the newly created specific node for the reconstruction.

an ever-increasing memory and computation requirements as the number of tasks increases. Instead, we propose the Dynamic Expansion Graph Model (DEGM), which dynamically addresses memory requirements, when learning a long sequence of tasks. A variety of tasks, would have representation variables in common and can be modelled by using sub-modules with fewer parameters while sharing most parameters. Given that most images have features in common, especially those representing basic levels of information such as textures, edges or flat surfaces, such a graph structure can benefit knowledge transfer by reusing previously learnt information when learning a new task. We also derive a novel mixture expansion mechanism that regulates the construction of the graph structure during training. Specifically, if a new task shares similar semantic information with respect to the already learnt knowledge, it is reasonable to reuse previously learnt information and parameters to promote the new task learning.

In this section, we present the detailed structure of the proposed DEGM. We derive a knowledge transfer mechanism in order to minimize the size of the model and promote future task learning. The proposed DEGM contains two types of modules : base and specific nodes/components, where the former is implemented by an independent VAE component while the latter is implemented by sub-models with fewer independent parameters. A base node is mainly used to learn an entirely different task, while a specific node is built upon the known information while being connected to the base nodes. The specific nodes aim to learn different details of the tasks which share some similar information by means of the base nodes to which they are connected. In the following sections, we describe the DEGM structure and expansion mechanism in detail.

5.1 Base and Specific Nodes/Components

The base nodes/components have independent inference and generation processes. Therefore, we implement them as VAEs, having encoding $q_{\omega_i}(\mathbf{z}\,|\,\mathbf{x})$ and decoding $p_{\theta_i}(\mathbf{x}\,|\,\mathbf{z})$ mechanisms, trained by ELBO, as in Eq. (1). For the encoding distribution $q_{\omega_i}(\mathbf{z}\,|\,\mathbf{x})$, we introduce two sub-inference models, $f_{\tilde{\omega}_i}\colon \mathcal{X} \to \tilde{\mathcal{Z}}$ and $f_{\omega'_i}\colon \tilde{\mathcal{Z}} \to \mathcal{Z}$ for modelling $q_{\omega_i}(\mathbf{z}\,|\,\mathbf{x})$, expressed by $f_{\tilde{\omega}_i} \odot f_{\omega'_i}\colon \mathcal{X} \to \mathcal{Z}$, where $\tilde{\mathcal{Z}}$ is an intermediate latent representation space with the dimension $|\tilde{\mathcal{Z}}| > |\mathcal{Z}|$. For the decoding distribution $p_{\theta_i}(\mathbf{x}\,|\,\mathbf{z})$, we introduce two networks, $g_{\tilde{\theta}_i}\colon \mathcal{X} \to \tilde{\mathcal{X}}$ and $g_{\theta'_i}\colon \tilde{\mathcal{X}} \to \mathcal{X}$, which form the decoder $g_{\tilde{\theta}_i} \odot g_{\theta'_i}\colon \mathcal{Z} \to \mathcal{X}$, where $\tilde{\mathcal{X}}$ is an intermediate representation space and $|\tilde{\mathcal{X}}|$ represents the dimension of $\tilde{\mathcal{X}}$ and we have $|\tilde{\mathcal{X}}| < |\mathcal{X}|$. Since the inference model or decoder in a base node has two connectable sub-models $\{f_{\tilde{\omega}_i}, g_{\tilde{\theta}_i}\}$, we can easily build a specific node (j-th node) by only building two separate sub-models $\{f_{\omega'_j}, g_{\theta'_j}\}$ connecting with the sub-models $\{f_{\tilde{\omega}_i}, g_{\tilde{\theta}_i}\}$ from all base nodes, forming a graph structure. In the following, we introduce a novel dynamic expansion mechanism to regulate model building during lifelong learning.

5.2 The Dynamic Expansion Mechanism

We assume that the proposed model has learnt t tasks using t processing nodes, where we have $K < t$ base nodes $\mathcal{G} = \{B_1, \ldots, B_K\}$ and $(t-K)$ specific nodes $\mathcal{S} = \{S_1, \ldots, S_{(t-K)}\}$. Let $\mathcal{GI}(\cdot)$ and $\mathcal{SI}(\cdot)$ represent the index functions that return the node index for $\mathcal{G}$ and $\mathcal{S}$. We use the index function to represent a certain base node ($B_i \in \mathcal{G}$) which consists of four sub-models $\{f_{\tilde{\omega}_{i^*}}, f_{\omega'_{i^*}}, g_{\tilde{\theta}_{i^*}}, g_{\theta'_{i^*}}\}$, two for encoding and another two for decoding, where $i^* = \mathcal{GI}(i)$. A certain specific node ($S_i \in \mathcal{S}$) consists of two

sub-models $\{f_{\omega'_{i'}}, g_{\theta'_{i'}}\}$, where $i' = \mathcal{SI}(i)$. Let $\mathbf{V} \in \mathbb{R}^{t\times t}$ denote an adjacency edge matrix that describes the directed graph edges from $\mathcal{S}$ to $\mathcal{G}$. The importance of the node i to the node j is represented by the directed edge $\mathbf{V}(i,j)$ which is used for the knowledge transfer when learning a new task while reusing the known information from the base nodes.

In the following, we introduce an expansion criterion that builds either a base node or a specific node for learning a new task. When the model sees a new task $\mathcal{T}_{t+1}$, characterized by the training set D^S_{t+1}, it evaluates the novelty of the new task by calculating $\mathcal{L}_{ELBO}(\mathbf{x}_j; B_i)$ on $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$ from the dataset D^S_{t+1}, ($n = 1000$ in experiments) :

$$h(i) = \big|\mathcal{L}_{ELBO}(B_i) - \mathbb{E}_{\mathbf{x}\sim\tilde{\mathcal{P}}_{t+1}}\mathcal{L}_{ELBO}(\mathbf{x}; B_i)\big|, \tag{24}$$

where $i = 1, \ldots, K$ and $\mathcal{L}_{ELBO}(B_i)$ preserves the best data log-likelihood estimated by the base node B_i on the previously assigned task. $\tilde{\mathcal{P}}_{t+1}$ is the distribution of D^S_{t+1}. A similar sample log-likelihood estimation approach has been used for expanding the number of components in mixture systems [40], [58]. However, these approaches do not utilize the entire previously learnt information when learning a new task, which prevents them from using the positive knowledge transfer. We employ Eq. (24) to determine the importance of each base node with respect to the new task, which is then used to regulate the knowledge transfer by weighting the information flow in a graph structure. In the following, we show how this is used for building both base and specific nodes and such a dynamic expansion process is illustrated in Fig.2.

Building a Base node. A base node is built if the incoming task $\mathcal{T}_{t+1}$ is novel enough, by considering a set of measures $\mathcal{H} = \{h(1), \ldots, h(K)\}$, each defined by Eq. (24) :

$$\min(\mathcal{H}) > \tau\,, \tag{25}$$

where the threshold τ is used to decide whether we build a base node for learning the $\mathcal{T}_{t+1}$-th task, based on the statistical difference between the learned knowledge and the information provided by the new task. Then we expand $\mathbf{V}$ to $\mathbb{R}^{(t+1,t+1)}$ and assign the edge value $\mathbf{V}(t+1, \mathcal{GI}(i)) = 0$, $i = 1, \ldots, K$ and build a new base node into $\mathcal{G}$. At the $(t+1)$-th task learning, we only train the $(t+1)$-th component (node) on the given task dataset by using the objective function from Eq. (1).

Building a Specific node. We create a specific node if Eq. (25) is not satisfied at the $\mathcal{T}_{t+1}$-th task learning. Then we update the edge matrix $\mathbf{V}$ by evaluating the importance weight, expressed as :

$$\begin{aligned}\mathbf{V}(t+1, \mathcal{GI}(i)) &= \frac{w^* - h(i)}{\sum_{j=1}^{K}(w^* - h(j))},\\ w^* &= \sum\nolimits_{j=1}^{K} h(j)\,, i = 1, \cdots, K\,,\end{aligned} \tag{26}$$

where we denote $\pi_i = \mathbf{V}(t+1, \mathcal{GI}(i))$ for simplification and $h(j)$ is calculated as in Eq. (24). According to the updated $\mathbf{V}$, we built a new sub-inference model $f_{\omega'_{(t+1)}}$, using a set of sub-models $\{f_{\tilde{\omega}_{i^*}} \,|\, i^* = \mathcal{GI}(i), i = 1, \ldots, K\}$, as $\sum_{i=1}^{K} \pi_i f_{\tilde{\omega}_{i^*}} \odot f_{\omega'_{(t+1)}}(\mathbf{x})$, which forms a mixture distribution $Q(\mathbf{z}) = \sum_{i=1}^{K} \pi_i Q_{\tilde{\omega}_{\mathcal{GI}(i)} \odot \omega'_{(t+1)}}(\mathbf{z}\,|\,\mathbf{x})$. Notice that each $Q_{\tilde{\omega}_{\mathcal{GI}(i)} \odot \omega'_{(t+1)}}(\mathbf{z}\,|\,\mathbf{x})$ is a variational distribution implemented by $f_{\tilde{\omega}_{\mathcal{GI}(i)}} \odot f_{\omega'_{(t+1)}}(\mathbf{x})$. A latent variable $\mathbf{z}$ is drawn from the distribution $Q(\mathbf{z})$. In Fig. 3, we show the structure of the decoder, where an identity function implemented by the input layer distributes the latent variable $\mathbf{z}$ to each $g_{\tilde{\theta}_{i^*}}(\mathbf{z}), i^* = \mathcal{GI}(1), \ldots, \mathcal{GI}(K)$, leading to $\tilde{\mathbf{x}} = \sum_{i=1}^{K} \pi_i g_{\tilde{\theta}_{i^*}}(\mathbf{z})$, where the intermediate feature information from $\mathcal{G}$ is weighted by π_i. Then we create a new sub-decoder $g_{\theta'_{(t+1)}}(\tilde{\mathbf{x}})$, that takes $\tilde{\mathbf{x}}$ as input and returns the reconstruction result of $\mathbf{x}$. We add this specific node $S_{(t+1)} = \{f_{\omega'_{(t+1)}}, g_{\theta'_{(t+1)}}\}$ into $\mathcal{S}$.

The procedure for building a specific node that connects with all base nodes $\mathcal{G}$ to form a graph structure in the proposed model is shown in Fig. 3. A similar procedure for determining the importance node in a neural network was considered in [59], [60]. However, the proposed DEGM is the first model where this mechanism aims to form a

Algorithm 1: The training algorithm for DEGM

Input: All training databases
Output: The model's parameters

for $i < N$ **do**
  if $i == 1$ **then**
    Build a base node B_1 which is added to $\mathcal{G}$;
    $isBase = True$;
  end
  for $index < batchCount$ **do**
    $\mathbf{x}_{batch} \sim D^S_i$;
    if *isBase == True* **then**
      Update $\{\omega_i, \theta_i\}$ by $\mathcal{L}_{ELBO}(\mathbf{x}_{batch}; \mathcal{M}_i)$;
    end
    else
      Update $\{\omega'_i, \theta'_i\}$ by $\mathcal{L}_{MELBO}(\mathbf{x}_{batch}; \mathcal{M}_i)$;
    end
  end
  Expansion mechanism;
  $\mathbf{x}_{new} \sim D^S_{i+1}$;
  Calculate the importance of each base node;
  for $k < K$ **do**
    $h(k) = |\mathcal{L}_{ELBO}(B_k) - \mathbb{E}_{\mathbf{x}\sim\mathbf{x}_{new}}\mathcal{L}_{ELBO}(\mathbf{x}; B_k)|$;
  end
  $\mathcal{H} = \{h(1), \ldots, h(K)\}$;
  if $\min\{\mathcal{H}\} \le \tau$ **then**
    The construction of the base node;
    $\mathbf{V}(i+1, \mathcal{GI}(j)) = 0, j = 1, \ldots, K$;
    Build a base component $B_{(K+1)}$ and add to $\mathcal{G}$;
    $isBase = True$;
    $K = K + 1$;
  end
  else
    The construction of the sub-graph structure;
    $\mathbf{V}(i+1, \mathcal{GI}(i)) = (w^* - ks(i))/\sum_{j=1}^{K}(w^* - ks(j))$, $w^* = \sum_{j=1}^{K} h(j)$, $i = 1, \ldots, K$;
    $\sum_{j=1}^{K} \pi_i f_{\tilde{\omega}_{\mathcal{GI}(j)}} \circ f_{\omega'_{(i+1)}}(\mathbf{x})$;
    Form the latent distribution;
    $Q(\mathbf{z}) = \sum_{i=1}^{K} \pi_i Q_{\tilde{\omega}_{\mathcal{GI}(i)} \odot \omega'_{(t+1)}}(\mathbf{z}\,|\,\mathbf{x})$;
    Obtain the intermediate representations ;
    $\tilde{\mathbf{x}} = \sum_{j=1}^{K} \pi_j g_{\tilde{\theta}_{\mathcal{GI}(j)}}(\mathbf{z})$;
    Build a sub-decoder $g_{\theta'_{(i+1)}}(\tilde{\mathbf{x}})$;
    Add $\mathcal{M}_{i+1}$ in $\mathcal{S}$;
    $isBase = False$;
  end
end

TABLE 2
Sample log-likelihood estimation results for five independent runs for the Split MNIST setting.

Methods	Run 1	Run 2	Run 3	Run 4	Run 5	Average
ELBO-GR	-97.97	-98.16	-98.70	-97.91	-98.39	-98.23 ± (0.28)
IWELBO-GR-50	-93.61	-93.69	-93.66	-93.64	-93.27	-93.57 ± (0.15)
IWELBO-GR-5	-95.96	-95.68	-95.95	-95.68	-95.73	-95.80 ± (0.12)
ELBO-GR*	-97.92	-98.79	-97.55	-98.45	-99.11	-98.36 ± (0.56)
IWELBO-GR*-50	-91.15	-91.33	-91.24	-91.23	-91.18	-91.23 ± (0.06)
CN-DPM*-IWELBO-50	-95.94	-96.35	-95.90	-95.74	-95.63	-95.91 ± (0.24)
LIMix-IWELBO-50	-95.57	-95.80	-95.84	-95.84	-95.63	-95.74 ± (0.11)
DEGM-ELBO	-93.85	-87.03	-93.15	-93.71	-93.51	-92.25 ± (2.62)
DEGM-IWELBO-50	-89.41	-89.58	-89.60	-85.78	-85.83	-88.04 ± (1.82)
DEGM-IWELBO-50	-89.61	-85.89	-89.60	-85.79	-85.75	**-87.33** ± (1.86)

graph structure that fully explores the benefits of knowledge transfer. To train this graph structure, we propose a new objective function that guarantees a lower bound to the marginal log-likelihood.

Theorem 4. Suppose that we have already learnt K base nodes at the t-th task learning ($\mathcal{T}_t$). Then, if we build a specific node at the $(t+1)$-th task learning, we can form a sub-graph structure which is optimized by using a valid lower bound (ELBO) (See details in the Appendix-H from SM) :

$$\begin{aligned}&\mathcal{L}_{MELBO}(\mathbf{x};\mathcal{M}_{(t+1)}) =: \\ &\mathbb{E}_{Q(\mathbf{z})}\left[\log p_{\theta'_{(t+1)}\odot\{\tilde{\theta}_{\mathcal{GI}(1)},\ldots,\tilde{\theta}_{\mathcal{GI}(K)}\}}(\mathbf{x}\mid\mathbf{z})\right] \\ &-\sum_{i=1}^{K}\pi_i KL\left(Q_{\tilde{\omega}_{\mathcal{GI}(i)}\odot\omega'_{(t+1)}}(\mathbf{z}\mid\mathbf{x})\,||\,p(\mathbf{z}_i)\right),\end{aligned} \tag{27}$$

where $q_{\tilde{\omega}_{\mathcal{GI}(i)}}\odot q_{\omega'_{(t+1)}}(\mathbf{z}\,|\,\mathbf{x})$ is the density function form of $Q_{\tilde{\omega}_{\mathcal{GI}(i)}\odot\omega'_{(t+1)}}(\mathbf{z}\,|\,\mathbf{x})$. $Q(\mathbf{z})$ is the variational distribution implemented by $\sum_{i=1}^{K}\pi_i Q_{\tilde{\omega}_{\mathcal{GI}(i)}\odot\omega'_{(t+1)}}(\mathbf{z}\,|\,\mathbf{x})$ which is a mixture inference model. Different from $q_\omega(\mathbf{z}\,|\,\mathbf{x})$ in Eq. (1), the variational distribution $Q(\mathbf{z})$ in Eq. (27) has continuously learnt the information of all given succession of tasks and thus would learn a new task fast by means of the positive knowledge transfer. Moreover, to relieve forgetting and ensure computational efficiency, we propose to update the current component $\{\omega'_{(t+1)},\theta'_{(t+1)}\}$ only when learning the $(t+1)$-th task. The first term in the RHS of Eq. (27) is implemented by the negative reconstruction error while the second term in Eq. (27) involves the sum of all KL divergence terms regularized by their associated weights π_i.

Model selection at the testing phase. The proposed DEGAM model does not require accessing the task information at the inference phase. Let us consider the DEGAM model with t trained nodes after LLL. We introduce the cluster assignment $\mathbf{u}$ in the DEGAM and the probability density of the output for DEGAM on n samples is represented by :

$$\begin{aligned}p(\mathbf{x}) = \prod_{i=1}^{n}\sum_{j=1}^{t} p\left(\mathbf{x}_i\mid\mathbf{u}_{(i,j)}\right)p\left(\mathbf{u}_{(i,j)}\right), \\ \mathbf{u}_{(i,j)}\in\{0,1\}.\end{aligned} \tag{28}$$

We particularly focus on the posterior $p(\mathbf{u}_{(i,j)}\,|\,\mathbf{x}_i)$ which can be rewritten by the Bayes' theorem :

$$\begin{aligned}p\left(\mathbf{u}_{(i,j)}\,|\,\mathbf{x}_i\right) &= \frac{p\left(\mathbf{x}_i\,|\,\mathbf{u}_{(i,j)}\right)p\left(\mathbf{u}_{(i,j)}\right)}{p\left(\mathbf{x}_i\right)} \\ &= \frac{p\left(\mathbf{x}_i\,|\,\mathbf{u}_{(i,j)}\right)p\left(\mathbf{u}_{(i,j)}\right)}{\sum_{k=1}^{t}p\left(\mathbf{u}_{(i,k)}\right)p\left(\mathbf{x}_i\,|\,\mathbf{u}_{(i,k)}\right)},\end{aligned} \tag{29}$$

where the prior is $p(\mathbf{u}_{(i,j)}) = 1/t$. Since $\log_a$ is a function monotonic increasing if $a > 1$, we can replace each $p(\mathbf{x}_i\,|\,\mathbf{u}_{(i,j)})$ by $\log p(\mathbf{x}_i\,|\,\mathbf{u}_{(i,j)})$ estimated by $\mathcal{L}_{ELBO}(,\mathbf{x}_i;\mathcal{M}_j)$ for the elements of $\mathcal{G}$ on the sample $\mathbf{x}$, and by $\mathcal{L}_{MELBO}(\mathbf{x}_i;\mathcal{M}_j)$ from Eq. (27) for the elements of $\mathcal{S}$. This selection process allows DEGAM to infer a related component without having task labels.

5.3 The Dynamic Expansion Adaptive Mechanism

In Section 5.2, we have shown that the edge matrix $\mathbf{V}$ is updated according to the importance of each base node when using a newly created specific node for learning a new task. However, using a fixed $\mathbf{V}$ would not allow to exploit the full potential of the knowledge transfer during the training. In this section, we propose the Dynamic Expansion Graph Adaptive Mechanism (DEGAM) algorithm which optimizes the edge matrix $\mathbf{V}$ so that the entire previously learnt knowledge is selectively used for learning the incoming task. Suppose that we have learnt K base nodes after the t-th task learning and built a new specific node for a new task $\mathcal{T}_{t+1}$. Then we calculate $\mathcal{H} = \{h(1),\cdots,h(K)\}$ by using Eq. (24) and update the edge matrix $\mathbf{V}$ by using Eq. (26). In order to continually update $\mathbf{V}$ during the $(t+1)$-th task learning, we introduce a group of adaptive parameters $\{\pi'_{1,t+1},\cdots,\pi'_{K,t+1}\}$ where $\pi'_{j,t+1}$ represents the weight of the j-th base node when learning a newly created specific node at the $(t+1)$-th task learning. These adaptive weights are normalized by using the softmax function :

$$\pi_{j,t+1} = \frac{\exp(\pi'_{j,t+1})}{\sum_{c}^{K}\exp(\pi'_{c,t+1})}. \tag{30}$$

Then we set $\mathbf{V}(t+1,\mathcal{GI}(i)) = \pi_{j,t+1}, j = 1,\cdots,K$ and update the adaptive parameters by :

$$\pi'_{j,t+1} = \pi'_{j,t+1} + l_1\nabla_{\pi'_{j,t+1}}\{-\mathcal{L}_{MELBO}(\mathbf{x};\mathcal{M}_{t+1})\}. \tag{31}$$

In practice, we update Eq. (31) along with the objective function from Eq. (27) during the $(t+1)$-th task learning. Moreover, if a new specific node is built for $\mathcal{T}_{t+2}$, we freeze previously learnt adaptive parameters $\{\pi'_{1,t+1},\cdots,\pi'_{K,t+1}\}$ to preserve the learnt graph structure and dynamically build a new group of adaptive parameters $\{\pi'_{1,t+2},\cdots,\pi'_{K,t+2}\}$ regulating the information flow from all base nodes at the $(t+2)$-th task learning. This mechanism avoids forgetting while efficiently reusing previously learnt knowledge and parameters when learning new tasks. The new model is named as the Dynamic Expansion Graph Adaptive Model (DEGAM).

5.4 Algorithm

We provide the pseudocode in **Algorithm 1**, which can be summarised in four steps :

TABLE 3
Sample log-likelihood estimation results for five independent runs for the Split Fashion setting.

Methods	Run 1	Run 2	Run 3	Run 4	Run 5	Average
ELBO-GR	-241.00	-240.38	-240.26	-240.82	-240.44	-240.58 ± (0.28)
IWELBO-GR-50	-236.48	-236.65	-236.98	-236.70	-236.49	-236.66 ± (0.18)
IWELBO-GR-5	-237.86	-238.19	-238.31	-238.00	-238.00	-238.08 ± (0.15)
ELBO-GR*	-242.54	-242.27	-244.76	-242.38	-247.61	-243.91 ± (2.06)
IWELBO-GR*-50	-236.71	-236.77	-237.04	-236.88	-237.08	-236.90 ± (0.14)
CN-DPM*-IWELBO-50	-237.60	-237.13	-237.81	-237.38	-237.43	-237.47± (0.22)
LIMix	-237.28	-237.50	-237.39	-237.37	-237.87	-237.48± (0.20)
DEGM-ELBO	-240.59	-237.68	-239.45	-237.60	-237.39	-238.54 ± (1.26)
DEGM-IWELBO-50	-236.32	-234.42	-236.66	-235.05	-234.66	-235.42 ± (0.89)
DEGM-IWELBO-50	-234.74	-232.97	-234.71	-233.32	-235.10	**-234.17** ± (0.85)

TABLE 4
The estimation of the sample log-likelihood under COFMI lifelong learning, where "D1", "D2", "D3", "D4" and "D5" represent the Caltech 101, OMNIGLOT, Fashion, MNIST and IFashion, respectively.

Methods	D1	D2	D3	D4	D5	Average
ELBO-GR	-163.68	-136.97	-247,91	-101.75	-237,03	-177,47
IWELBO-GR-50	-153.65	-131.37	-243.62	-97,29	-234.58	-172.10
IWELBO-GR-5	-166.05	-134.07	-245.78	-99.43	-235.73	-176.21
ELBO-GR*	-175.10	-140.05	-247.54	-102.73	-237.06	-180.50
IWELBO-GR*-50	-215.16	-144.42	-246.35	-102.82	-236.12	-188.97
CN-DPM*-IWELBO-50	-136.22	-150.31	-259.12	-131.34	-243.97	-184.19
LIMix-IWELBO-50	-137.32	-150.79	258.69	-131.25	-243.50	-184.32
DEGM-ELBO	-137.72	-116.07	-233.32	-122.33	-234.62	-168.81
DEGM-IWELBO-50	-133.72	-112.12	-230.46	-127.17	-232.21	-167.14
DEGM-IWELBO-50	-137.08	-113.29	-231.83	-115.10	-233.47	**-166.15**

Step 1 (First task learning) To start learning the first task, we create a base node (component) B_1 and insert it into $\mathcal{G}$. We train B_1 on D_1^S using Eq. (1).

Step 2 (Training the component) If we have a base node, then we update the current base node on D_1^S using Eq. (1), otherwise, we update the current specific node on D_1^S using Eq. (27).

Step 3 (Checking the expansion criterion) Once the learning of a particular task (i-th task) is finished, we see a new task $\mathcal{T}_{i+1}$. We collect 10,000 training samples from D_{i+1}^S, denoted as $\mathbf{x}_{new}$, and then calculate $\mathcal{H}$ using Eq. (24). Then $\mathcal{H}$ is used for the next step.

Step 4 (Build either a base or a particular node) If Eq. (25) is satisfied, we build a base node, otherwise, we build a specific node creating a graph structure for the model. We then update only the current base or specific node on the training set D_{i+1}^S using the appropriate loss function. Once the $(i+1)$-th task is finished, we go back to step 2.

Step 4 (Model evaluation) After all tasks are completed, we perform the component selection procedure for a given sample, by selecting the most suitable component for evaluation according to Eq. (29).

6 Experiments

In this section, we investigate the effectiveness of the proposed model when using density estimation and unsupervised generative modelling tasks. We also perform a detailed ablation study to analyze the performance of the proposed model under different configurations.

6.1 Density Estimation Task

Setting. We introduce a new benchmark for the density estimation task under lifelong learning. First, we consider learning several tasks within a single data domain. Following from the setting [61], we create Split MNIST by dividing MNIST [62], which contains data from 10 classes, into five tasks, with each task containing training images belonging to two successive classes. We repeat this for Fashion [63], resulting in the Split Fashion. Second, we consider the learning of five different tasks, where each task is associated with a different data domain : Caltech 101 [64], OMNIGLOT [65], Fashion, MNIST, InverseFashion (IFashion). The IFashion database is formed by creating new images with their pixels resulting by subtracting from 255 all pixel values of the images from Fashion. We name the resulting sequence of databases as COFMI. All images in each dataset are binarized according to the setting from [24].

Network architecture and hyperparameter setting. The inference and generator for the VAEs are implemented by two fully connected networks. Each network contains two layers, each with 200 hidden units. We also extend VAEs by using two stochastic layers where the latent dimension for the first and second stochastic layers is of 100 and 50, respectively. The VAE with two stochastic layers, trained with a classical ELBO, Eq. (1), is named as ELBO-GR*. If VAE uses only a single stochastic layer, then the latent dimension is 100. When importance sampling is used in VAEs, as defined by Eq. (14), we call the network as IWELBO-GR-K' where K' represents the number of weighted samples used in the objective function during training. For the experiments, we use the GeForce GTX 1080 GPU, and the operating system Linux Ubuntu 18.04.5.

The inference model and the generator in DEGM are implemented by four sub-models, and each sub-model has only one layer with 200 units. Therefore, the inference and generator model in each specific node consists of two sub-models built upon all base component units. Similarly, with a single VAE model, DEGM can be extended by considering the importance sampling framework, using Eq. (14) for training base components. We also adapt the IWELBO bound from Eq. (14) for training specific nodes using the following objective function :

$$\mathcal{L}_{MELBO_{K'}}(\mathbf{x};\mathcal{M}) = \mathbb{E}_{\mathbf{z}_1,..,\mathbf{z}_{K'}\sim Q(\mathbf{z})}\left[\log \frac{1}{K'}\sum_{i=1}^{K'}\frac{p(\mathbf{x},\mathbf{z}_i)}{Q(\mathbf{z}_i)}\right]. \tag{32}$$

Baselines. When DEGM uses ELBO or IWELBO bounds, we call it as DEGM-ELBO and DEGM-IWELBO-K', respectively. From the setting in [24], we use the same network

TABLE 5
Unsupervised generation results after the lifelong learning of MSFIR. The results of other models, used for comparison, are cited from [12].

	MSE						SSMI						PSNR					
Datasets	LGM	CURL	BE	LIMix	DEGAM	DEGM	LGM	CURL	BE	LIMix	DEGAM	DEGM	LGM	CURL	BE	LIMix	DEGAM	DEGM
MNIST	129.93	211.21	19.24	26.66	20.42	21.15	0.45	0.46	0.92	0.88	0.91	0.91	14.52	13.27	22.57	21.09	22.27	22.09
Fashion	89.28	110.60	38.81	30.19	26.48	28.81	0.51	0.44	0.61	0.76	0.80	0.79	15.82	14.89	14.46	21.25	20.88	20.51
SVHN	169.55	102.06	39.57	35.07	31.78	29.50	0.24	0.26	0.66	0.65	0.64	0.68	8.11	10.86	18.90	14.92	15.44	15.74
IFashion	432.90	115.29	36.52	30.14	27.34	28.59	0.26	0.54	0.75	0.79	0.81	0.80	9.04	15.51	19.32	20.26	20.73	20.53
RMNIST	130.28	279.47	25.41	22.80	23.23	25.81	0.45	0.29	0.88	0.90	0.90	0.89	14.51	10.84	21.31	21.81	21.66	21.26
Average	190.38	163.72	31.91	28.97	**25.85**	26.77	0.38	0.39	0.76	0.79	**0.81**	**0.81**	12.40	13.07	19.31	19.86	**20.20**	20.02

architecture for the implementation of DEGM and consider several baselines for comparison. First, a single VAE model with the generative replay mechanism is called ELBO-GR. When such a VAE model uses IWEBLO bound, we call it as IWELBO-GR-K'. In addition to the single VAE network architecture, we consider LIMix [12] and CN-DPM [13], which are dynamic expansion models. CN-DPM is primarily used in task-free continual learning. In order to adopt it to our task, we implement a variant of CN-DPM, namely CN-DPM*, where the Dirichlet-based expansion mechanism is replaced by dynamically building a new component for learning a new task. Thus CN-DPM creates new components while ensuring its optimal performance on all previously learnt tasks. Furthermore, we implement LIMix [12] using an optimal configuration where the number of components matches the number of tasks.

We use the Adam optimization algorithm for training all models, with a learning rate of 0.0001. We consider Bernoulli decoders, while we use the binary cross-entropy as the reconstruction error term in ELBO. The batch size and the number of training epochs for each training task are of 64 and 500, respectively. We search for the threshold τ for the expansion criterion from Eq. (25) within the interval 35-40 for split MNIST/Fashion.

Results. We perform five independent runs for the Split MNIST setting. We estimate the log-likelihood for the testing samples using the IWEBLO bound with $K' = 5000$ samples and we report the results in Table 2, where "*" indicates that the model uses two stochastic layers. We can observe that IWELBO-GR*-50 achieves better results than a single VAE employing a single stochastic layer. In the following, we perform five independent runs on the Split Fashion, and we report the results in Table 3. The proposed DEGM-based method learns four base nodes and one specific node after the lifelong learning of Split MNIST and Split Fashion.

In the following, we evaluate the effectiveness of various models in a more challenging learning setting. COFMI setting includes five different tasks and the model can only access training samples from the dataset associated with the current task learning. First, we train various models under COFMI lifelong learning, while searching for the best threshold τ from Eq. (25) within the range 80-100. Then, we report the results in Table 4 and the edge matrix $\mathbf{V}$ of DEGM-IWELBO-50 when learning COFMI is shown in Fig. 4. After all tasks have been completed, the proposed models have trained four base and one specific node, respectively. The empirical results for COFMI from Table 4 show that ELBO-GR* and IWELBO-GR*-50 significantly drop their performance on the earlier tasks when considering the cross-domain learning setting when compared with the VAEs that do not use stochastic layers. According to all these results, the proposed models achieve better performance than other baselines.

6.2 Unsupervised Generative Modelling

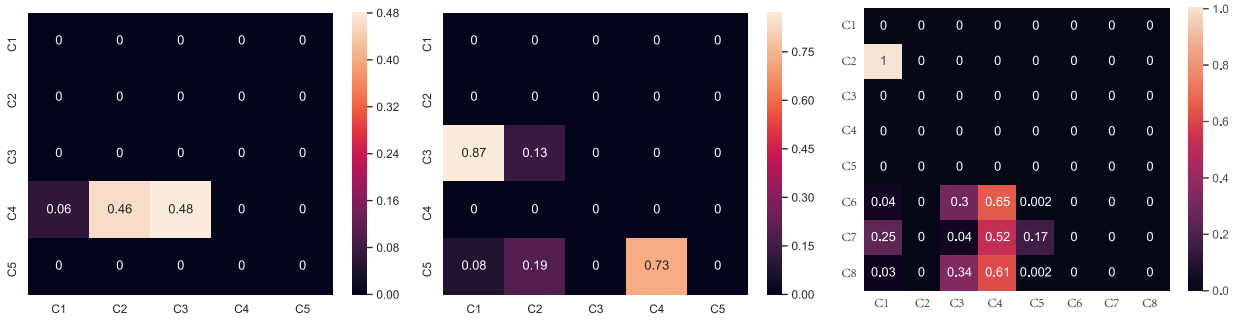
Baselines. We consider a baseline that dynamically creates a new base node whenever learning a new task, called DEGM-2, ensuring optimal performance for all past tasks. We also consider the Batch Ensemble (BE) [16] which is mainly applied in classification tasks. In order to adapt BE to be used for unsupervised generative modelling, we implement each ensemble member of the BE as a VAE. All models are trained by using ELBO. We also consider a baseline that trains DEGM considering only five epochs for training, when the newly created component is a specific node, namely DEGM-1. The number of parameters required by various models is provided in Section 6.5.

Performance criterion. In unsupervised image reconstruction, we adapt the Structural Similarity Index Measure (SSIM) [66], the Mean Squared Error (MSE) and the Peak-Signal-to-Noise Ratio (PSNR) [66], in order to evaluate the image reconstruction quality. The calculation for MSE, SSIM and PSNR is provided in **Appendix-I** from SM.

Datasets. First, we consider the learning of a long sequence of tasks, including MNIST [62], SVHN [67], Fashion [63], InverseFashion (IFashion), Rotated MNIST (RMNIST) namely MSFIRC. RMNIST is obtained by rotating each image from MNIST by 180 degrees. In the following, we also consider the learning of a sequence of more complex tasks (characterized by databases containing images of higher complexity), including CelebA [68], CACD [69], 3D-Chair [70], Ommiglot [65], ImageNet* [71], Car [72], Zappos [73], CUB [74], named CCCOSCZC. For CelebA, CACD, we randomly choose 10,000 samples as the testing set while the other samples are used for training. For 3D-chair, we randomly choose 1000 samples as the testing set and the remaining samples as the training set. For ImageNet, we randomly choose 10,000 and 50,000 samples as the testing and training set, respectively. For CUB, we randomly choose 1000 samples as the testing set and the other samples as the training set.

TABLE 6
The performance of various models after the lifelong learning of CCCOSCZC.

Criteria	MSE					SSMI					PSNR				
	BE	LGM	DEGM	DEGM-2	CN-DPM*	BE	LGM	DEGM	DEGM-2	CN-DPM*	BE	LGM	DEGM	DEGM-2	CN-DPM*
CelebA	213.9	535.6	229.2	217.0	215.4	0.69	0.48	0.66	0.69	0.69	23.5	19.3	23.2	23.4	23.5
CACD	414.9	814.3	368.3	281.95	347.3	0.57	0.47	0.62	0.68	0.63	20.6	17.33	21.2	22.4	21.4
3D-Chair	649.1	2705.9	324.0	291.46	513.8	0.73	0.42	0.84	0.86	0.79	19.0	13.54	22.4	23.1	20.5
Omniglot	875.1	5958.9	225.6	195.7	343.2	0.73	0.22	0.92	0.93	0.89	17.9	9.2	24.0	24.6	22.1
Sub-ImageNet	758.4	683.1	689.6	652.8	769.1	0.37	0.42	0.41	0.43	0.37	18.5	18.9	19.0	19.2	18.5
Car	745.1	583.7	588.8	565.9	709.8	0.39	0.48	0.47	0.49	0.42	18.0	19.0	19.0	19.2	18.2
Zappos	451.1	431.2	263.4	275.8	280.7	0.68	0.60	0.75	0.74	0.73	20.0	20.2	22.4	22.3	22.1
CUB	492.0	330.2	461.3	569.6	638.6	0.35	0.48	0.45	0.43	0.35	19.0	20.9	19.3	18.6	18.0
Average	575.0	1505.4	393.8	381.3	477.2	0.60	0.45	0.64	0.66	0.61	19.6	17.3	21.3	21.6	20.5

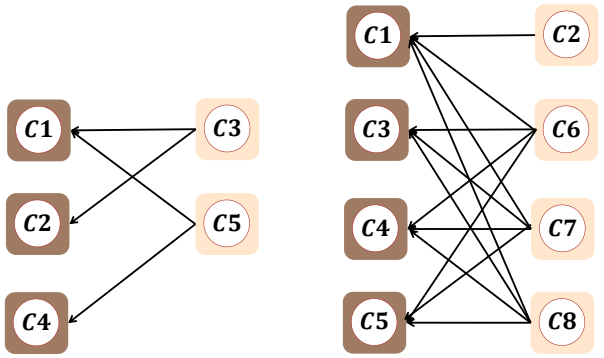


(a) $\mathbf{V}$ of DEGM-IWVAE-50 under COFMI lifelong learning. (b) $\mathbf{V}$ of DEGM after the MSFIR lifelong learning. (c) $\mathbf{V}$ of DEGM after the CCCOSCZC lifelong learning.

Fig. 4. Illustration of the Graph adjacency matrix $\mathbf{V}$, characterizing the connections between different DEGM's nodes after lifelong learning. In (a) "C1" represents the first component and "C4" is a specific node that connects the previously trained three base nodes ("C1", "C2", "C3").

Hyperparameters. For the CCCOSCZC setting, the number of training epochs for learning each task is set to 20. We train various models by using the Adam optimization algorithm [75] with a learning rate of 0.0002 while other hyperparameters are considered as the default values. The threshold τ from Eq. (25) for DEGM on MSFIR and CCCOSCZC is considered as 400 and 600, respectively. To evaluate the performance of various models under the generative modelling task, we consider the Square Loss (SL), the Structural Similarity Index Measure (SSIM) [66] and the Peak-Signal-to-Noise Ratio (PSNR) [66]. After the lifelong learning, we evaluate the performance of various models on all testing samples and the empirical results are reported in Tables 5 and 6, respectively. We can observe that BE and CN-DPM* result in significant performance degradation on the last task, which indicates that these two models suffer from the interference between data associated with different tasks. Meanwhile, the proposed DEGM is not affected by such interferences because it dynamically builds a base node for learning Cifar10 data.

Results. The adjacency matrix showing the connections between the nodes within the graph structure $\mathbf{V}$ of DEGM after MSFIR lifelong learning is provided in Fig. 4b where "C1" represents the first component node. We can observe that DEGM has learnt three base nodes and two specific nodes, respectively. These two specific nodes are built upon



(a) DEGM after MSFIR lifelong learning. (b) DEGM after CCCOSCZC lifelong learning.

Fig. 5. Edge information for DEGM after lifelong learning. Darker and lighter colours are used for the base and specific nodes, respectively.

the information associated with the three base nodes. In Fig. 4 we illustrate $\mathbf{V}$ for DEGM after the CCCOSCZC lifelong learning. DEGM learns to build the base nodes for the first, third, fourth, and fifth tasks. The edge information between the members of $\mathcal{S}$ and those of $\mathcal{G}$ is provided in Fig. 5, where the base and specific nodes are drawn with different colours.

In addition, we also compare the proposed DEGAM and

This article has been accepted for publication in IEEE Transactions on Pattern Analysis and Machine Intelligence. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2025.3564188

TABLE 7
The performance of the proposed DEGM and DEGAM when compared to other methods after the lifelong learning of MSFIR.

	MSE					SSMI					PSNR				
Datasets	GNR	DDGR	CAM-GAN	DEGAM	DEGM	GNR	DDGR	CAM-GAN	DEGAM	DEGM	GNR	DDGR	CAM-GAN	DEGAM	DEGM
MNIST	22.15	21.56	20.16	20.42	21.15	0.88	0.89	0.91	0.91	0.91	21.92	22.01	22.28	22.27	22.09
Fashion	257.49	78.26	31.61	26.48	28.81	0.36	0.62	0.75	0.80	0.79	11.21	16.72	19.99	20.88	20.51
SVHN	107.83	68.18	28.41	31.78	29.50	0.18	0.48	0.65	0.64	0.68	11.92	13.16	15.84	15.44	15.74
IFashion	94.69	59.06	32.81	27.34	28.59	0.55	0.51	0.78	0.81	0.80	16.05	17.41	19.83	20.73	20.53
RMNIST	22.19	21.37	22.99	22.80	23.23	0.88	0.89	0.90	0.90	0.90	21.92	21.96	21.70	21.66	21.26
Average	100.87	49.68	27.20	**25.85**	26.77	0.57	0.67	0.80	0.81	0.81	16.61	18.25	19.93	**20.20**	20.02

TABLE 8
The performance of several variations of the proposed model for the MSFIR lifelong learning.

	MSE									
Datasets	DEGM-4	DEGM-5	DEGM-6	DEGM-7	DEGAM	DEGM	DEGM-2	DEGM-3	CN-DPM*-1	CN-DPM*-2
MNIST	20.99	20.92	21.52	21.25	20.42	21.15	20.59	20.84	21.24	20.19
Fashion	39.18	36.51	36.27	36.24	26.48	28.8	26.84	28.08	33.07	31.93
SVHN	24.41	23.77	23.48	24.43	31.78	29.50	24.00	31.98	30.38	28.11
IFashion	35.61	37.64	36.31	37.48	27.34	28.59	27.06	27.86	34.48	32.89
RMNIST	23.42	23.01	22.85	23.65	23.23	25.81	20.70	26.79	22.88	22.20
Average	28.72	28.37	28.09	28.61	25.85	26.77	23.84	27.11	28.41	27.06

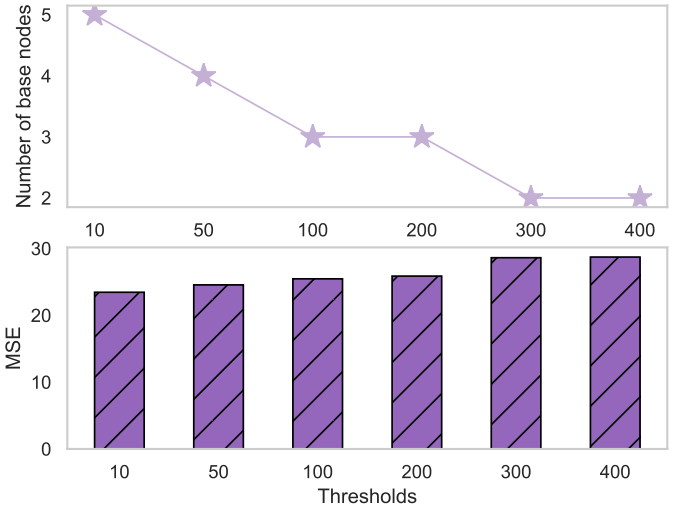


Fig. 6. The performance and the number of base nodes for DEGAM under MSFIR lifelong learning when changing threshold τ in Eq. (25).

DEGM with other state-of-the-art methods, including CAM-GAN [50], [52], Generative Negative Replay (GNR) [76] and the Deep Diffusion-based Generative Replay (DDGR) [77]. To enable CAM-GAN [50], [52] to perform the image reconstruction task, we implement each sub-model of CAM-GAN using a VAE. In addition, to enable Deep Diffusion-based Generative Replay (DDGR) [77] and Generative Negative Replay (GNR) [76] to perform the image reconstruction task, we treat DDGR and GNR as the teacher module and train an additional VAE model as the student module that learns cumulated information from the teacher module as well as from the new task. The results are provided in Tab. 7.

6.3 Ablation Study

The effects of threshold τ: In the following, we evaluate the performance of the proposed DEGAM under different threshold τ values for the architecture expansion criterion from Eq. (25). The empirical results obtained after the LLL training of DEGAM on MSFIR datasets are provided in Fig. 6. When the threshold τ decreases, the proposed DEGAM tends to use more base components while also improving its performance. A trade-off between the performance and the model size can be observed for $\tau = 300$.

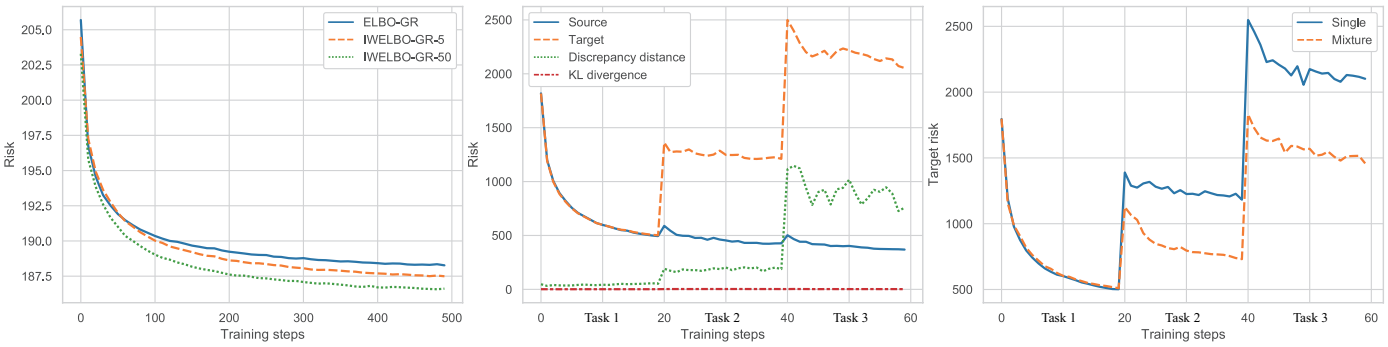
In the following, we consider several variants of the DEGM model in order to investigate the effectiveness of the proposed methodology.

DEGM-3: We consider employing fewer training epochs (10 epochs) for the training of specific nodes.

DEGM-4: This baseline generates information flows from all trained components to a new component. For instance, if a new component receives the information flow from the processing units members of $\mathcal{S}$, we will sum up the latent codes and intermediate representations from the sub-inference and sub-decoders of these components. DEGM-4 does not use the adaptive weights.

DEGM-5: We implement this baseline by creating edges without using adaptive weights. The training process for this baseline is described as follows: after finishing the t-th task learning we have a set of K probabilistic distance evaluations, denoted by $\mathcal{H} = \{h(1), \ldots, h(K)\}$, calculated according to Eq. (24), which can be used to build the edges from a new node to the processing units members of $\mathcal{G}$.

This article has been accepted for publication in IEEE Transactions on Pattern Analysis and Machine Intelligence. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2025.3564188



(a) The target risk according to Eq. (16). (b) Evaluation of the Generalization Bound (GB) on -ELBO from Eq. (13). (c) Target risk on a single and mixture model.

Fig. 7. The estimation of the target and source risks.

(a) DEGM (b) Single VAE (c) CN-DPM*

Fig. 8. Accumulated target risks when changing the learning order for the tasks during LLL, calculated as the square loss during the training.

We then set a threshold τ which is used to update $\mathbf{V}$ such that if each $h(i) < \tau$, then $\mathbf{V}(t+1, \mathcal{GI}(i)) = 1$, otherwise $\mathbf{V}(t+1, \mathcal{GI}(i)) = 0$. This means that if $g(i) > \tau$, then the construction of a new component does not reuse the information and parameters from the i-th component in DEGM.

DEGM-6: For this baseline, we consider that the adaptive weight for each edge is equal. This means that the importance of all base components is treated the same for a new task.

DEGM-7: We implement this baseline by creating only a single edge for a new component to a certain base component that has the maximum sample log-likelihood for the data associated with the new task.

CN-DPM*-1; This baseline builds new components and creates connections with previously learned components, similar to DEGM-4.

CN-DPM*-2: We implement this baseline by using the large model which contains 1.3×10^9 parameters.

We train all baselines with the same hyperparameter configuration and the empirical results are given in Table 8. These results indicate that when the adaptive weighting is combined with the dynamic extension mechanism improves the performance of DEGM. It also shows that the proposed DEGM performs better than CN-DPM*-2, which uses many more parameters. This indicates that the proposed DEGM can reuse the previously learned knowledge for learning new tasks while also reducing the overall model size.

6.4 Empirical Analysis of the Theoretical Results

In this section, we provide the results of empirical tests for the theoretical analysis provided in this paper. First, we train a single VAE model on the binarized Caltech 101 database. Then we use this VAE model to produce generative replay samples corresponding to the learnt database. We then train ELBO-GR and IWELBO-GR-K' on a joint dataset consisting of the generated data by the model combined with data sampled from the Fashion dataset, where $K' \in \{5, 50\}$ is the number of important samples used in Eq. (14). Finally, we calculate the average target risk (LHS of Eq. (16)) for these models in order to investigate the tightness between IWELBO and the negative log-likelihood (NLL). We know that Lemma 1 is only applied for the Gaussian decoder. However, the conclusion of Lemma 1 is also observed when the VAE model uses the Bernoulli decoder. The results from Fig. 7a, indicate that IWELBO-GR-50 achieves a tighter bound when compared to IWELBO-GR-5 and ELBO-GR, which empirically proves that $\mathcal{L}_{LELBO_{50}} \leq \mathcal{L}_{LELBO_5}$ when the generator distribution is fixed and $|KL_1 - KL_2| = 0$, as discussed in Lemma 1.

In the following we train a single VAE model that uses the Gaussian decoder with the identity matrix as the covariance, on the lifelong learning of MNIST, Fashion and IFashion, where all images are greyscale, with pixels values within $[0, 255]$. To estimate the reconstruction error, according to the ELBO, we normalize each image such that each pixel value is divided by the image size (28×28), as in [78]. We estimate the risk and the discrepancy distance for

TABLE 9
Model size (number of parameters) for various models when lifelong learning the task sequences CCCOSCZC and MSFIR.

	LGM	BE	DEGM	DEGM-2	CN-DPM*	LIMix
CCCOSCZC	1.9×10^9	3.9×10^9	3.2×10^8	1.3×10^9	9.4×10^9	9.4×10^9
MSFIR	1.5×10^8	9.4×10^8	1.6×10^8	8.7×10^8	4.2×10^8	-

each training epoch according to Eq. (13) from Lemma 1. We provide the results in Fig. 7b, where the source risk (the first term in RHS of Eq. (13)) remains stable and the discrepancy distance term $\mathcal{L}^{\star}_{\text{ELBO}}(\cdot)$, Eq. (3), represented in $\mathcal{R}_A(\cdot)$, increases while more tasks are learned. We observe that 'KL divergence', calculated as $|KL_1 - KL_2|$, shown in Fig. 7b, increases slowly. This demonstrates that the discrepancy distance term is key for reducing the GB gap.

We also provide empirical results for Lemma 2. We learn a dynamic expansion model $\mathbf{M} = \{\mathcal{M}_1, \mathcal{M}_2\}$ when learning the sequence of MNIST, Fashion, IFashion (MFI) databases, where the learning model has two components after lifelong learning. First, we freeze $\mathcal{M}_1$ when the first task learning is finished, while $\mathcal{M}_2$ is trained to learn Fashion and IFashion, respectively. We also train a single VAE model $\mathcal{M}$ with the generative replay mechanism under the MFI setting, which is used for comparison. We calculate the average target risk (NLL estimated by ELBO) in each training epoch and we report the results in Fig. 7c, where "single" and "mixture" represent $\mathcal{M}$ and $\mathbf{M}$, respectively. These results show that the dynamic expansion model obtains a tighter GB when compared with the case when considering a single VAE model, as discussed in Lemma 2.

We also investigate whether the proposed DEGM is robust to changing the order in which the tasks are learnt. First, we randomly generate three different orders as : Cifar10, IMNIST, Fashion, MNIST, SVHN, IFashion (CIFMSI); Fashion, SVHN, MNIST, Cifar10, IMNIST, IFashion (FSMCII); IFashion, IMNIST, Cifar10, Fashion, SVHN, MNIST (IICFSM). We then calculate the cumulative target risk for all tasks, under each ordering of the learning tasks, and report the results in Fig. 8a, b and c, for DEGM, single VAE and CN-DPM*, respectively, where "Order1", "Order2" and "Order3" denote CIFMSI, FSMCII and IICFSM, respectively. It can be observed that a single VAE model tends to have a different average target risk when the order of tasks is changed. A similar phenomenon is observed in CN-DPM*, which produces different results when trained under "Order1" and "Order2". This is because CN-DPM* can only use the knowledge of the first task when learning future tasks. In contrast, the proposed DEGM tends to achieve similar performance for whatever order of learning tasks is chosen.

6.5 Lifelong Model Size

We calculate the number of parameters required by various lifelong learning methods when learning the task sequences CCCOSCZC and MSFIR and provide them in Table 9. We can observe that the proposed DEGM architecture requires fewer parameters than other baselines.

7 Conclusion

In this paper, we propose a new theoretical framework for analyzing the forgetting behaviour of deep generative models in lifelong learning. The proposed theoretical analysis provides new insights into how previously learnt knowledge is lost when learning new tasks. Inspired by the theoretical analysis, we propose the Dynamic Expansion Graph Model (DEGM) which dynamically expands a graph structure consisting of base and specific nodes. Such nodes, representing entire neural networks, are added to the graph according to the novelty of the information being learnt. New base nodes are added when learning a completely different task. Meanwhile, specific nodes are added when a task related to the knowledge already known to DEGM, is identified. To effectively use the previously learned knowledge when learning a new task, we enable adaptive weights that regulate the information flow, within the graph structure. Moreover, we propose a Dynamic Expansion Graph Adaptive Mechanism (DEGAM) that dynamically updates the weights when integrating a new task into the existing knowledge of the model. The proposed mechanism is empowered by the benefits of knowledge transfer when learning new tasks. The theoretical and empirical results show that both DEGAM as well as DEGM can significantly reduce the number of parameters while maintaining optimal performance for all tasks during the lifelong learning.

References

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

[2] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," in *Advances in Neural Information Proc. Systems (NIPS)*, 2017, pp. 2990–2999.

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

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

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

[6] J. Ramapuram, M. Gregorova, and A. Kalousis, "Lifelong generative modeling," *Neurocomputing*, vol. 404, pp. 381–400, 2020.

[7] M. Rostami, S. Kolouri, P. K. Pilly, and J. McClelland, "Generative continual concept learning," in *Proc. AAAI Conf. on Artificial Intelligence*, 2020, pp. 5545–5552.

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

[9] F. Ye and A. G. Bors, "Lifelong teacher-student network learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 10, pp. 6280–6296, 2022.

[10] ——, "Learning latent representations across multiple data domains using lifelong VAEGAN," in *Proc. of European Conference on Computer Vision (ECCV), vol. LNCS 12365*, 2020, pp. 777–795.

[11] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, "VEEGAN: Reducing mode collapse in gans using implicit variational learning," in *Advances in Neural Inf. Proc. Systems (NIPS)*, 2017, pp. 3308–3318.

[12] F. Ye and A. G. Bors, "Lifelong infinite mixture model based on knowledge-driven dirichlet process," in *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2021, pp. 10 695–10 704.

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

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

[15] G. Jerfel, E. Grant, T. Griffiths, and K. A. Heller, "Reconciling meta-learning and continual learning with online mixtures of tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 9122–9133.

[16] Y. Wen, D. Tran, and J. Ba, "BatchEnsemble: an alternative approach to efficient ensemble and lifelong learning," in *Proc. Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:2002.06715*, 2020.

[17] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "β-VAE: Learning basic visual concepts with a constrained variational framework," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2017, pp. 1–13.

[18] L. Chen, S. Dai, Y. Pu, C. Li, Q. Su, and L. Carin, "Symmetric variational autoencoder and connections to adversarial learning," in *Proc. Int. Conf. on Artificial Intel. and Statistics (AISTATS), vol. PMLR 84*, 2018, pp. 661–669.

[19] F. Ye and A. G. Bors, "Mixtures of variational autoencoders," in *Proc. Int. Conf. on Image Processing Theory, Tools and Applications (IPTA)*, 2020, pp. 1–6.

[20] ——, "Deep mixture generative autoencoders," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 10, pp. 5789–5803, 2022.

[21] ——, "InfoVAEGAN: Learning joint interpretable representations by information maximization and maximum likelihood," in *Proc. IEEE Int. Conf. on Image Processing (ICIP)*, 2021, pp. 749–753.

[22] ——, "Learning joint latent representations based on information maximization," *Information Sciences*, vol. 567, pp. 216–236, 2021.

[23] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," in *Proc. of 22nd Conf. on Learning Theory (COLT), arXiv preprint arXiv:0902.3430*, 2009.

[24] Y. Burda, R. Grosse, and R. Salakhutdinov, "Importance weighted autoencoders," in *Proc. Int. Cont. of Learning Representations (ICLR), arXiv preprint arXiv:1509.00519*, 2015.

[25] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, J. Sutskever, and M. Welling, "Improved variational inference with inverse autoregressive flow," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2016, pp. 4743–4751.

[26] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proc. Int. Conf. on Machine Learning (ICML), vol. PMLR 37*, 2015, pp. 1530–1538.

[27] L. Mescheder, S. Nowozin, and A. Geiger, "Adversarial variational Bayes: Unifying variational autoencoders and generative adversarial networks," in *Proc. Int. Conf. on Machine Learning (ICML), vol. PMLR 70*, 2017, pp. 2391–2400.

[28] C.-W. Huang, K. Sankaran, E. Dhekane, A. Lacoste, and A. Courville, "Hierarchical importance weighted autoencoders," in *Int. Conf. on Machine Learning (ICML), vol. PMLR 97*, 2019, pp. 2869–2878.

[29] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner, "Variational continual learning," in *Proc. Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:1710.10628*, 2017.

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

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

[32] 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)*, 2019, pp. 11 817–11 826.

[33] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. K. Dokania, P. H. S. Torr, and M. Ranzato, "On tiny episodic memories in continual learning," in *arXiv preprint arXiv:1902.10486*, 2019.

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

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

[36] M. M. Derakhshani, X. Zhen, L. Shao, and C. Snoek, "Kernel continual learning," in *Proc. Int. Conf. on Machine Learning (ICML), vol. PMLR 139*, 2021, pp. 2621–2631.

[37] Y. Shi, L. Yuan, Y. Chen, and J. Feng, "Continual learning via bit-level information preserving," in *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 16 674–16 683.

[38] S. Wang, X. Li, J. Sun, and Z. Xu, "Training networks in null space of feature covariance for continual learning," in *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 184–193.

[39] J. Von Oswald, C. Henning, B. F. Grewe, and J. Sacramento, "Continual learning with hypernetworks," in *International Conference on Learning Representations (ICLR) arXiv preprint arXiv:1906.00695*, 2020.

[40] D. Rao, F. Visin, A. A. Rusu, Y. W. Teh, R. Pascanu, and R. Hadsell, "Continual unsupervised representation learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019, pp. 7645–7655.

[41] C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang, "Adanet: Adaptive structural learning of artificial neural networks," in *Proc. of Int. Conf. on Machine Learning (ICML), vol. PMLR 70*, 2017, pp. 874–883.

[42] R. Polikar, L. Upda, S. S. Upda, and V. Honavar, "Learn++: An incremental learning algorithm for supervised neural networks," *IEEE Trans. on Systems Man and Cybernetics, Part C*, vol. 31, no. 4, pp. 497–508, 2001.

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

[44] T. Xiao, J. Zhang, K. Yang, Y. Peng, and Z. Zhang, "Error-driven incremental learning in deep convolutional neural network for large-scale image classification," in *Proc. of ACM Int. Conf. on Multimedia*, 2014, pp. 177–186.

[45] G. Zhou, K. Sohn, and H. Lee, "Online incremental feature learning with denoising autoencoders," in *Proc. Int. Conf. on Artificial Intelligence and Statistics (AISTATS), vol. PMLR 22*, 2012, pp. 1453–1461.

[46] F. Ye and A. G. Bors, "Lifelong generative modelling using dynamic expansion graph model," in *Proc. of AAAI on Artificial Intelligence*, 2022, pp. 8857–8865.

[47] A. Douillard, A. Ramé, G. Couairon, and M. Cord, "Dytox: Transformers for continual learning with dynamic token expansion," in *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

[48] Z. Wang, Z. Zhang, C.-Y. Lee, H. Zhang, R. Sun, X. Ren, G. Su, V. Perot, J. Dy, and T. Pfister, "Learning to prompt for continual learning," in *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 139–149.

[49] T. Doan, S. I. Mirzadeh, J. Pineau, and M. Farajtabar, "Efficient continual learning ensembles in neural network subspaces," *arXiv preprint arXiv:2202.09826*, 2022.

[50] S. Varshney, V. K. Verma, P. Srijith, L. Carin, and P. Rai, "Camgan: Continual adaptation modules for generative adversarial networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 15 175–15 187, 2021.

[51] C. P. Le, J. Dong, A. Aloui, and V. Tarokh, "Few-shot continual learning for conditional generative adversarial networks," *arXiv preprint arXiv:2305.11400*, 2023.

[52] Y. Cong, M. Zhao, J. Li, S. Wang, and L. Carin, "Gan memory with no forgetting," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 481–16 494, 2020.

[53] F. Ye and A. G. Bors, "Lifelong mixture of variational autoencoders," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 34, no. 1, pp. 461–474, 2023.

[54] S. Kuroki, N. Charoenphakdee, H. Bao, J. Honda, I. Sato, and M. Sugiyama, "Unsupervised domain adaptation based on source-guided discrepancy," in *Proc. AAAI Conf. on Artificial Intelligence*, vol. 33, 2019, pp. 4122–4129.

[55] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[56] J. Domke and D. R. Sheldon, "Importance weighting and variational inference," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 4470–4479.

[57] T. Cemgil, S. Ghaisas, K. Dvijotham, S. Gowal, and P. Kohli, "The autoencoding variational autoencoder," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 15 077–15 087.

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

[59] R. Aljundi, K. Kelchtermans, and T. Tuytelaars, "Task-free continual learning," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 11 254–11 263.

[60] S. Jung, H. Ahn, S. Cha, and T. Moon, "Continual learning with node-importance based adaptive group sparse regularization," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 3647–3658.

[61] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proc. of Int. Conf. on Machine Learning (ICML), vol. PLMR 70*, 2017, pp. 3987–3995.

[62] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[63] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[64] L. Fei-Fei, R. Fergus, and P. Perona, "Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories," *Computer Vision and Image Understanding*, vol. 106, pp. 59–70, 2007.

[65] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[66] A. Hore and D. Ziou, "Image quality metrics: PSNR vs. SSIM," in *Proc. Int. Conf. on Pattern Recognition (ICPR)*, 2010, pp. 2366–2369.

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

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

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

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

[71] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, 2012, pp. 1097–1105.

[72] L. Yang, P. Luo, C. Change Loy, and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3973–3981.

[73] A. Yu and K. Grauman, "Semantic jitter: Dense supervision for visual comparisons via synthetic images," in *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, 2017, pp. 5571–5580.

[74] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD birds-200 dataset," California Institute of Technology, Tech. Rep. CNS-TR-2010-001, 2010.

[75] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:1412.6980*, 2015.

[76] G. Graffieti, D. Maltoni, L. Pellegrini, and V. Lomonaco, "Generative negative replay for continual learning," *Neural Networks*, vol. 162, pp. 369–383, 2023.

[77] R. Gao and W. Liu, "DDGR: continual learning with deep diffusion-based generative replay," in *Proc. International Conference on Machine Learning, vol. PMLR 202*, 2023, pp. 10 744–10 763.

[78] Y. Park, C. Kim, and G. Kim, "Variational Laplace autoencoders," in *Proc. Int. Conf. on Machine Learning (ICML), vol. PMLR 97*, 2019, pp. 5032–5041.