# Lifelong Infinite Mixture Model Based on Knowledge-Driven Dirichlet Process

Fei Ye and Adrian G. Bors

Department of Computer Science, University of York, York YO10 5GH, UK

fy689@york.ac.uk, adrian.bors@york.ac.uk

## Abstract

*Recent research efforts in lifelong learning propose to grow a mixture of models to adapt to an increasing number of tasks. The proposed methodology shows promising results in overcoming catastrophic forgetting. However, the theory behind these successful models is still not well understood. In this paper, we perform the theoretical analysis for lifelong learning models by deriving the risk bounds based on the discrepancy distance between the probabilistic representation of data generated by the model and that corresponding to the target dataset. Inspired by the theoretical analysis, we introduce a new lifelong learning approach, namely the Lifelong Infinite Mixture (LIMix) model, which can automatically expand its network architectures or choose an appropriate component to adapt its parameters for learning a new task, while preserving its previously learnt information. We propose to incorporate the knowledge by means of Dirichlet processes by using a gating mechanism which computes the dependence between the knowledge learnt previously and stored in each component, and a new set of data. Besides, we train a compact Student model which can accumulate cross-domain representations over time and make quick inferences. The code is available at* https://github.com/dtuzi123/Lifelong-infinite-mixture-model.

## 1. Introduction

Lifelong learning (LLL) aims to learn successively a series of tasks from their corresponding probabilistic representations of specific databases. The objective of the lifelong learning model is to implement all learnt tasks at any given time. Modern deep learning approaches have been successful in a variety of applications including image translation [24], image synthesis [29] and object detection [37], but all these models face a major challenge in the performance, when applied on prior tasks, while learning multiple tasks, one after the other. This challenge is caused by catastrophic forgetting which happens when a model adapts its parameters in order to learn a new task [31].
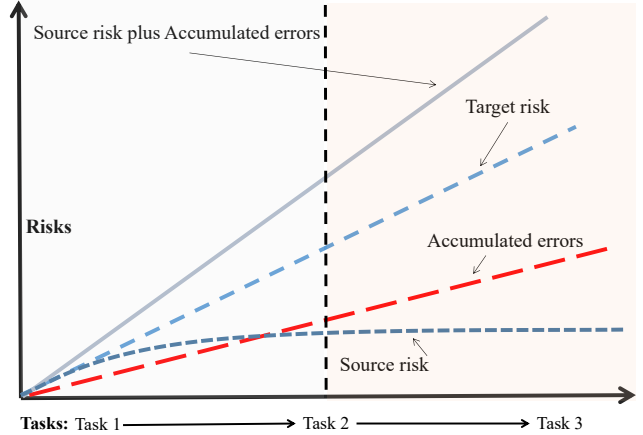


Figure 1. The process of forgetting the information from a certain dataset after learning two additional tasks. The source distribution generated by the Generative Replay Mechanims (GRM) is degenerated when learning a new task.

The Generative Replay Mechanism (GRM) [41] is a popular lifelong learning approach showing promising results overcoming catastrophic forgetting [1, 33, 39, 48, 56]. A generative replay model $G_\theta : \mathcal{Z} \to \mathcal{X}$, aims to transform a low dimensional random variable $\mathcal{Z}$ into a high dimensional variable $\mathcal{X}$. $G_\theta$ can be an implicit generative model such as a Generative Adversarial Network (GAN) [11] or an explicit latent model such as a Variational Autoencoder (VAE) [16]. Once a task is learnt, $G_\theta$ generates data which then can be combined with data sampled from a given database, corresponding to a new task, to form a joint dataset used for training. Some methods [1, 41] reduce the memory size by only generating a batch of samples for each training step, by using only a copy $G_{\theta'}$ of the GRM. However, a major challenge for GRM based methods is that of gradually losing knowledge across tasks since a GRM model is trained on its own generations repeatedly. Another drawback, when GANs are used as GRMs, is that of facing mode collapse [44]. Two solutions have been proposed to address this problem. Rao *et al.* [34] enable GRMs with a network expansion mechanism in which the model's capacity is increased when shifting data distributions. The other solution is to use the expansion mechanism [20] or an ensemble

structure [10, 14, 47] in which each expert is built on the top of a shared module and only a single expert is updated during the training. These approaches usually preserve the best performance of previous tasks, but the theoretical analyse behind these methods is not well understood.

In this paper, we provide the theoretical analysis for lifelong learning models, inspired by the idea illustrated in Fig. 1. The forgetting behaviour of the model, when learning a certain task, is affected by an ever increasing upper bound (solid line in Fig. 1) to the target-risk, during the lifelong leaning. This is mainly caused by the increased accumulated errors when learning additional tasks while the discrepancy between the target and source distribution is also gradually increased. However, the optimal source-risk can not guarantee a low target-risk since the source distribution of the trained model is gradually degenerated. Inspired by these results, the main idea of the proposed Lifelong Infinite Mixture (LIMix) model is to automatically grow its network architecture if the given task is sufficiently novel when compared to the previously learned knowledge or update an appropriate component that has a small discrepancy to the given task. The Dirichlet process, which is usually computational expensive when using the expectation-maximization algorithm to estimate component parameters [8], can be used for these mechanisms. In order to reduce the computational costs and make an accurate inference for the selection and expansion of the model architecture, we infer an indicator variable for each data sample by using a gating mechanism based on the Dirichlet process that computes the corrections between the knowledge stored in each component and the new data. Furthermore, by accumulating knowledge, while enabling fast inference across data domains, using a lightweight model is an attractive feature in LLL, which does not appear in existing lifelong mixtures or ensemble models [20, 47]. Our main contributions are :

- We analyse the forgetting behaviour during LLL by evaluating the accumulated risk and find that the discrepancy distance between the source and target distribution is key to overcome forgetting.

- This is the first study to provide theory insights when using mixture models for LLL. We also extend the theoretical analysis to explain the performance change for a model when shifting the order of tasks.

- We propose a new lifelong mixture model with theoretical guarantees for LLL. We also explore training a compact Student model from the mixture under LLL.

## 2. Related works

Artificial lifelong learning models are trained three different approaches: regularization, dynamic architectures and memory replay. Regularization methods introduce an auxiliary term in the loss function in order to penalize changes in the network's weights when learning a new task [12, 15, 17, 23, 28, 38]. This can alleviate catastrophic forgetting but can not guarantee the effective performance on the previously learnt tasks. Dynamic architectures methods would grow the size of the network by adding processing layers or increase the number of parameters in order to adapt to a growing number of tasks [6, 20, 32, 40, 50, 59]. Lee *et al*. [20] used Dirichlet processes for expanding their network architecture. However, their approach does not provide any theoretical guarantees for the performance, and mainly focuses on the generation and classification tasks and still requires to store past samples.

Memory replay approaches would either use a generator [1, 34, 33, 41, 56, 51, 52, 55, 57] or a memory buffer [2, 4, 46] as a replay mechanism generating data, which is statistically consistent to the previously learnt knowledge. Continual Unsupervised Representation Learning (CURL) [34] is a memory replay method which trains a latent generative model in order to replay data consistent with the previously learnt information. CURL expands its architecture for the inference component but not for the decoder. This can lead to catastrophic forgetting when learning certain tasks.

Besides these three directions of research, there are other methods such as [10, 53, 54] which create network architectures consisting of a shared module and other multiple task-specific modules. The shared module would not change its parameters too much, while the task-specific modules would only update their parameters when learning certain tasks. These approaches may guarantee the full performance on the previously learnt tasks but would still require knowing the number of tasks and the task labels during both training and testing phases. In this paper, we assume that our lifelong learning model does not know the exact number of tasks to be learnt while the task boundaries are only provided during the training phase. Although some methods can be used in a task-free manner [34], they are still limited to learning a sequence of tasks from a single domain.

## 3. Methodology

### 3.1. Problem setting

The lifelong artificial learning systems aim to learn a sequence of tasks, where each time we have a training set $D_S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ of $N$ paired instances of data $\mathbf{x}_i$, considered as images, and their corresponding labels $y_i$. Let us consider the training of a model (classifier or generator) $\mathcal{M}_t$, sequentially, with $t$ tasks, each defined by the training set and the testing set $D_T$. The learning goal of $\mathcal{M}_t$ is to make precise predictions for classification tasks on all testing data sets $\{D_T^1, \ldots, D_T^t\}$ after the training with a sequence of sets $\{D_S^1, \ldots, D_S^t\}$. When considering the unsupervised learning setting, the learning goal of $\mathcal{M}_t$ is to learn meaningful data representations without having any labels.

## 3.2. The lifelong infinite mixture (LIMix) model

In this section, we first introduce a mixture of deep learning networks for unsupervised learning, and then extend this framework for a supervised setting. Let us define a deep learning mixture of $K$ components at the $t$-th task learning:

$$p(\mathbf{x}, \mathbf{z} \mid \Theta, \pi_1, \ldots, \pi_K) = \sum_{j=1}^{K} \pi_j p_{\theta_j}(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z}), \quad (1)$$

where $\Theta = \{\theta_1, \ldots, \theta_K\}$ are the parameters of components. $\mathbf{x} \in \mathcal{X}$ and $\mathbf{z} \in \mathcal{Z}$ are the observed and latent variables where $\mathcal{X}$ and $\mathcal{Z}$ are the input and latent space. Each $p_{\theta_j}(\mathbf{x} \mid \mathbf{z})$ is implemented as a Gaussian distribution $\mathcal{N}(g_{\theta_j}(\mathbf{z}), \Sigma)$ with $\Sigma$ considered as a diagonal matrix, and $g_{\theta_j}$ represents the deterministic mapping that maps $\mathbf{z}$ into the mean of $p_{\theta_j}(\mathbf{x} \mid \mathbf{z})$, implemented as a deep learning network [53, 54]. $\pi_j$ is the mixing parameter for the $j$-th component. $p(\mathbf{z})$ is the prior, implemented by the normal distribution. One approach for training this model is to maximize the marginal likelihood of $p(\mathbf{x}, \mathbf{z} \mid \Theta, \Omega, \pi_1, \ldots, \pi_K)$ as:

$$p\left(\mathbf{x}_1^1, \ldots, \mathbf{x}_{N_M}^M \mid \Theta, \pi_1, \ldots, \pi_K\right) = \prod_{t=1}^{M} \prod_{n=1}^{N_t} \int \sum_{j=1}^{K} \pi_j p_{\theta_j}\left(\mathbf{x}_n^t \mid \mathbf{z}\right) p(\mathbf{z}) \, \mathrm{d}\mathbf{z}, \quad (2)$$

where $M$ and $N_i$ are the total number of tasks and the number of data samples considered for each $i$-th task, $i = 1, \ldots, M$. This optimization problem is intractable in practice since we cannot access data samples from previous tasks with associated training sets $\{D_S^i \mid i = 1, \ldots, t-1\}$ after learning the $t$-th task. Furthermore, by maximizing Eq. (2) when learning only a single task would cause the mixture model to forget the information learnt previously, as the network parameters are updated to new values during the training with the data $\mathbf{x}_i^t \sim D_S^t$. To address this problem, we propose to adapt the number of components in the mixture according to the complexity of the tasks being learnt. The Dirichlet process is suitable for the selection and expansion mechanisms for the mixture [35]. In this paper we adapt a Dirichlet process by defining a probabilistic measure of similarity in order to be able to train the same mixing component with several tasks. We introduce an indicator variable $c_i^t$ for each $\mathbf{x}_i^t$ which indicates which component is assigned to $\mathbf{x}_i^t$. Estimating the mixing weights $\{\pi_1, \ldots \pi_K\}$ can be indirectly realized by the inference of the indicator variable, [5] :

$$p\left(c_1^1, \ldots, c_{N_K}^K \mid \pi_1, \ldots, \pi_K\right) = \prod_{j=1}^{K} \pi_j^{N_j}, \quad (3)$$

where $\{\pi_1, \pi_2, \ldots, \pi_K\} \sim Dir(\mathbf{a}), \mathbf{a} = \{a/1, \cdot, a/K\}$ and $Dir(\mathbf{a})$ is a symmetric Dirchlet distribution with $\mathbf{a}$ its parameter vector. Inferring a single indicator $c_i^t$ can be im-

plemented by integrating over $\pi_j$ and allowing $K$ to increase to infinity, [36] :

$$p\left(c_i^t = j \mid c_{-i}^t, a\right) = \frac{n_{-i,j}}{n-1+a}, \quad (4)$$

where $n_{-i,j}$ is the number of samples that are associated with the $j$-th component, excluding $\mathbf{x}_i^t$, where the subscript $-i$ denotes all indices except $i$. Eq. (4) represents the probability for the $i$-th data sample to be associated with the $j$-th component of the mixture. The hyperparameter vector $\mathbf{a}$ can influence the prior probability of assigning a sample to a new component and the total number of components after training [36]. However, this probability does not infer properly $c_i^t$ given that it does not evaluate the consistency of the new sample $\mathbf{x}_i^t$ with the information learnt by each component. Comparing the previously learned knowledge with the incoming data is useful for selecting the most suitable mixture component in order to be updated, or for adding a new component to the mixture. In this paper, instead of comparing the new sample with those already stored [20, 36], we propose to incorporate the knowledge learned by each component for estimating $n_{-i,j}$ in order to consider the similarity between the prior knowledge and the new sample :

$$n_{-i,j} = (n-1) \times \frac{e^{(1/\mathbf{K}_{i,j})}}{\sum_{q=1}^{K} e^{(1/\mathbf{K}_{i,q})} + e^{(1/\mathcal{V})}}, \quad (5)$$

where $\mathcal{V}$ is a constant controlling the expansion of the mixture model and

$$\mathbf{K}_{i,j} = \left| F\left(\mathbf{x}_i^t \mid c_i^t, \theta_j, \omega_j\right) - F\left(\mathbf{x}'_{i,j} \mid c_i^t, \theta_j, \omega_j\right)\right| \quad (6)$$

and $F(\cdot \mid c_i^t, \theta_j, \omega_j)$ is the log-likelihood function. $n$ is the total number of samples, and $\mathbf{x}'_{i,j}$ is the $i$-th sample generated by the component $j$. We evaluate $\mathbf{K}_{i,j}$ between the log-likelihood of the new sample $\mathbf{x}_i^t$ and the log-likelihood of the generated sample $\mathbf{x}'_{i,j}$, estimated by the $j$-th component. If $\mathbf{K}_{i,j}$ is very small, then $\mathbf{x}_i^t$ has a high likelihood to be assigned to the $j$-th component. The probability of generating a new component and of assigning the indicator variable $c_i^t$ to $\mathbf{x}_i^t$ is then defined as:

$$p\left(c_i^t = K+1 \mid c_{-i}^t, a\right) = \frac{a + (n-1)\left(\frac{\exp(1/\mathcal{V})}{Z}\right)}{n-1+a}, \quad (7)$$

where $Z = \sum_{q=1}^{K} e^{(1/\mathbf{K}_{i,q})} + e^{(1/\mathcal{V})}$ is denominator of (5). **Determining the indicator for a new task.** By using the inferring indicator variable for all samples when learning the last given $t$-th task, is computationally intensive. We also know that data samples from a database, characterizing a certain task, share similar features. We consider only the calculation of a single indicator variable for a task, after each task switch, while the indicator variables for all data samples within a task are identical. Suppose that we have finished the $(t-1)$-th task learning, we would like to infer the indicator for the $t$-th task. Firstly, we randomly select a

group of samples $\{\mathbf{x}_1^t, \ldots, \mathbf{x}_{n_G}^t\}$ from the $t$-th training set and then calculate the probability of each $\mathbf{x}_i^t$ belonging to each $j$-th component $p(c_i^t = j \mid c_{-i}^t, a), \forall i = 1, \ldots, n_G$, where $n_G$ is the group size. Then we define the indicator $c^t$ for the $t$-th task by :

$$c^t = \arg\max_{j=1,\ldots,K+1} \frac{1}{n_G} \sum_{i=1}^{n_G} p\left(c_i^t = j \mid c_{-i}^t, a\right), \quad (8)$$

Once $c^t$ is determined, at the $t$-th task learning we only update the parameters of the chosen component instead of updating the whole model $\Theta$, by maximizing the sample log-likelihood $\log \int p_{\theta_{c^t}}(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z}) \, d\mathbf{z}$ which is intractable in optimization since we need integrating over $\mathbf{z}$. Similar to [16], we introduce to maximize a lower bound to the sample log-likelihood by using a variational distribution $q_{\omega_{c^t}}(\mathbf{z} \mid \mathbf{x})$ at $t$-th task learning :

$$\log p_{\theta_{c^r}}(\mathbf{x}) \geq \mathbb{E}_{q_{\omega_{c^t}}(\mathbf{z}|\mathbf{x})} \left[\log p_{\theta_{c^t}}(\mathbf{x} \mid \mathbf{z})\right] \\ - D_{KL}\left[q_{\omega_{c^t}}(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})\right], \quad (9)$$

where the right-hand side is our log-likelihood function $F(\cdot \mid c^t, \theta_{c^t}), \omega_{c^t})$, called the Evidence Lower Bound (ELBO), used for training the model and the evaluation of Eq. (6). $p_{\theta_{c^t}}(\mathbf{x} \mid \mathbf{z})$ and $q_{\omega_{c^t}}(\mathbf{z} \mid \mathbf{x})$ are decoding and encoding distributions, implemented by the network $g_{\theta_{c^t}} : \mathcal{Z} \to \mathcal{X}$ and $f_{\omega_{c^t}} : \mathcal{X} \to \mathcal{Z}$, respectively, where the subscript denotes the component index. In Lemma 4 from Appendix-F of the Supplementary Material (SM), we show how LIMix can infer across domains by the selection process.

## 3.3. Learning prediction tasks

In this section, we extend LIMix model for prediction tasks. Conditional VAEs [43] is one of the most used generative models for predictive tasks, defined by:

$$\log p_{\varsigma_{c^t}}(y \mid \mathbf{x}) \geq \mathbb{E}_{q_{\omega_{c^t}}(\mathbf{z}|\mathbf{x},y)} \left[\log p_{\varsigma_{c^t}}(y \mid \mathbf{x}, \mathbf{z})\right] - \\ D_{KL}\left(q_{\omega_{c^t}}(\mathbf{z} \mid \mathbf{x}, y) \| p_{\varsigma_{c^t}}(\mathbf{z} \mid \mathbf{x})\right). \quad (10)$$

For classification, $y$ belongs to the discrete domain (one-hot vector), and $p_{\varsigma_{c^t}}(y \mid \mathbf{x}, \mathbf{z})$ is implemented as a classifier. We represent $p_{\varsigma_{c^t}}(\mathbf{z} \mid \mathbf{x})$ as $\mathcal{N}(0, \mathbf{I})$ in Eq. (10) for reducing the model size, and this results in the objective function:

$$\mathcal{L}_\mathrm{P} = \mathbb{E}_{q_{\omega_{c^t}}(\mathbf{z}|\mathbf{x},y)} \left[\log p_{\varsigma_{c^t}}(y \mid \mathbf{x}, \mathbf{z})\right] \\ - D_{KL}(q_{\omega_{c^t}}(\mathbf{z} \mid \mathbf{x}, y) \| p(\mathbf{z})). \quad (11)$$

We also require each component to learn a generator in order to overcome forgetting when reusing a selected component to model more than one task. Therefore, we define the ELBO for the generative model, $p_{\theta_{c^t}}(\mathbf{x} \mid y)$ as :

$$\mathcal{L}_\mathrm{G} = \mathbb{E}_{q_{\omega_{c^t}}(\mathbf{z}|\mathbf{x},y)} \left[\log p_{\theta_{c^t}}(\mathbf{x} \mid \mathbf{z}, y)\right] \\ - D_{KL}(q_{\omega_{c^t}}(\mathbf{z} \mid \mathbf{x}, y) \| p(\mathbf{z})). \quad (12)$$

Each component in the classification setting has three

models $p_{\varsigma_{c^t}}(y \mid \mathbf{x}, \mathbf{z}), q_{\omega_{c^t}}(\mathbf{z} \mid \mathbf{x}, y), p_{\theta_{c^t}}(\mathbf{x} \mid y, \mathbf{z})$. The likelihood function $F(\cdot \mid c^t, \theta_{c^t}, \omega_{c^t})$ for the classification is only $\mathcal{L}_G$ and the main objective function for optimizing the $c^t$-th component is to maximize $\mathcal{L}_G + \mathcal{L}_P$. In practice, we optimize $\mathcal{L}_\mathrm{P}$ and $\mathcal{L}_\mathrm{G}$ separately, in the same mini-batch. In Appendix-J from SM, we provide the framework for applying LIMix for Image-to-Image Translation tasks and in Appendix-M we provide the experimental results.

## 3.4. Training a compressed Student model

In order to reduce the complexity of LIMix, we propose to share most parameters of the generator and the inference models through a joint network, where the parameters $\theta_i = \{\theta_S, \tilde{\theta}_i\}$ and $\omega_i = \{\omega_S, \tilde{\omega}_i\}$ of each component consists of the shared part $\{\theta_S, \omega_S\}$ and the individual part $\{\tilde{\theta}_i, \tilde{\omega}_i\}$ for each component. The mixing components are built on top of the shared component. We also train a compressed Student model under the unsupervised learning only, aiming to embed knowledge from LIMix into one latent space which supports interpolation across multiple domains. The Student shares the same network architecture with the component and is trained using the knowledge distillation (KD) loss along with the sample log-likelihood at the $t$-th task learning :

$$\mathcal{L}_{stu} = \underbrace{\mathbb{E}_{S_{t,X}} \log p_{\theta_{stu}}(\mathbf{x})}_{\text{Log-likelihood}} + \underbrace{\sum_{i=1}^{K} \mathbb{E}_{\mathbb{P}_{\theta_i}} \log p_{\theta_{stu}}(\mathbf{x})}_{\text{knowledge distilltion}} \quad (13)$$

where $\theta_{stu} = \{\theta_S, \tilde{\theta}_{stu}\}$ and $\tilde{\theta}_{stu}$ is the individual set for the Student. $\log p_{\theta_{stu}}(\mathbf{x})$ is estimated by ELBO and $\mathbb{P}_{\theta_i}$ is the distribution modelled by the $i$-th component in LIMix. More details together with the LIMix model diagram are provided in Appendix-J from SM. Additionally, this paper does not focus on the improvement of KD and we find that the Student is weaker than LIMix. This is theoretically explained in Appendix-I.2 from SM.

## 4. Theoretical analysis for lifelong learning

In this section, we first provide the theoretical analysis for the proposed infinite mixture model that does not grow its architecture during the lifelong learning. In this case, the model uses GRM to overcome the catastrophic forgetting and is seen as a single model represented by $\mathcal{M}(\theta, \varsigma, \varphi)$, consisting of a generator $g_\theta : \mathcal{Z} \to \mathcal{X}$ and a classifier $h_\varsigma : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{Y}$ is an output space, which is $\{-1, 1\}$ for binary classification and $\{1, 2, \ldots, n'\}, n' > 2$ for multi-class classification. We assume that the model also contains a task-inference network $U_\varphi : \mathcal{X} \to \mathcal{T}$ where $\mathcal{T}$ is the task domain. We then provide theoretical guarantees for the convergence of LIMix. Eventually, we analyse the forgetting behaviour of existing methods and the trade-off between the model's performance and complexity.

## 4.1. Preliminaries

**Definition 1** (Approximation distribution). *Let us define a joint distribution $\tilde{S}^t$ approximated by the generator $g_{\theta^t}$ and the classifier $h_{\varsigma^t}$ of $\mathcal{M}(\theta^t, \varsigma^t, \varphi^t)$ trained on a sequence of sets $\{D_S^1, \ldots, D_S^t\}$. We assume that we have a perfect task-inference network $U_\varphi$, which can exactly predict the task label for a given sample $\mathbf{x}$. With the optimal task-inference network $U_\varphi$, we can form several joint distributions $\{\tilde{S}_1^t, \ldots, \tilde{S}_t^1\}$ where each $\tilde{S}_i^{(t-i+1)}$ is made up of a set of samples where each paired sample is drawn by using the sampling process $\{\mathbf{x}, h_{\varsigma^t}(\mathbf{x})\} \sim \tilde{S}^t$ if $U_\varphi(\mathbf{x}) = i$. We use the superscript $(t-i+1)$ in $\tilde{S}_i^{(t-i+1)}$ to denote that $\tilde{S}_i^1$ is refined for $(t-i+1)$ times through the GRM processes after the $t$-th task learning. We further use $\tilde{S}_{i,\mathcal{X}}^{(t-i+1)}$ to denote the marginal distribution of $\tilde{S}_i^{(t-i+1)}$. $\tilde{S}_i^n$ is formed by the samples from $D_S^i$ if $n = 1$, otherwise, by samples drawn from $G_{\theta^n}$ and $h_{\varsigma^n}$ with the optimal task-inference network.*

**Definition 2** (Data distribution across tasks). *Let $S_i$ represent the joint distribution characterizing the probabilistic representation for the testing set in the $i$-th database $D_T^i$, and $S_{i,\mathcal{X}}$ is its marginal distribution along $\mathcal{X}$.*

**Assumption 1** *We assume $\tau \colon \mathcal{Y} \times \mathcal{Y} \to [0,1]$ be a symmetric and bounded loss function $\forall (y, y') \in \mathcal{Y}^2, \tau(y, y') \leq M'$ and $\tau(\cdot, \cdot)$ obeys the triangle inequality, where $M'$ is a positive number.*

**Definition 3** (Discrepancy distance). *For two given joint distributions $\tilde{S}_i^{(t-i+1)}$ and $S_i$ over $\mathcal{X} \times \mathcal{Y}$ and $\tau : \mathcal{Y} \times \mathcal{Y} \to [0,1]$ is a loss function satisfying Assumption 1. Let $h, h' \in \mathcal{H}$ be two classifiers, where $\mathcal{H}$ is the space of all classifiers, and we define the discrepancy distance $\Psi$ between the two marginals $\tilde{S}_{i,\mathcal{X}}^{(t-i+1)}$ and $S_{i,\mathcal{X}}$ as:*

$$
\Psi\left(\tilde{S}_{i,X}^{(t-i+1)}, S_{i,X}\right) = \sup_{(h,h')\in\mathcal{H}^2} \left| \underset{\tilde{S}_{i,X}^{(t-i+1)}}{\mathbb{E}} \left[\tau\left(h'\left(\mathbf{x}\right), h\left(\mathbf{x}\right)\right)\right] \right.
$$
$$
\left. - \underset{S_{i,X}}{\mathbb{E}}\left[\tau\left(h'\left(\mathbf{x}\right), h(\mathbf{x})\right)\right] \right|. \tag{14}
$$

**Definition 4** (Empirical risk). *For a given loss function $\tau \colon \mathcal{Y} \times \mathcal{Y} \to [0,1]$ and a joint distribution $S_i$, we form an empirical set where we draw each paired sample as $\{\mathbf{x}_j^i, y_j^i\} \sim S_i$. the empirical risk $\mathrm{R}(h, S_i)$ for a given classifier $h \in \mathcal{H}$, is evaluated by $n$ number of independent runs.*

$$
\mathrm{R}\left(h, S_i\right) = \frac{1}{n}\sum_{j=1}^{n} \tau\left(h\left(\mathbf{x}_j^i\right), y_j^i\right). \tag{15}
$$

## 4.2. Risk bounds for lifelong learning

The discrepancy distance, defined through Eq. (14), was used to derive generalization bounds for domain adaptation methods [3, 7, 26] and also for matching generated and real data distributions in the GAN's discriminator during training [9, 21, 22]. In the following we derive the risk bound for the lifelong learning model based on the discrepancy distance. The main idea for analyzing the degenerated performance of the model is to evaluate the risk between the target and the dynamically degenerated source distribution caused by the retraining process using GRMs. In this case, the errors accumulated when learning each task can be measured in an explicit way.

**Theorem 1** *Let $S_i$ and $\tilde{S}_i^{(t-i+1)}$ be two joint distributions over $\mathcal{X} \times \mathcal{Y}$. Let $h_i = \arg\min_{h \in \mathcal{H}} \mathrm{R}(h, S_i)$ and $\tilde{h}_i^{(t-i+1)} = \arg\min_{h \in \mathcal{H}} \mathrm{R}(h, \tilde{S}_i^{(t-i+1)})$ represent the ideal classifiers for $S_i$ and $\tilde{S}_i^{(t-i+1)}$, respectively, where $\mathcal{H}$ is the classifier space. By satisfying Assumption 1, we have:*

$$
\mathrm{R}\left(h, S_i\right) \leq \mathrm{R}'\left(h, \tilde{h}_i^{(t-i+1)}, \tilde{S}_i^{(t-i+1)}\right)
$$
$$
+ \Psi\left(S_{i,X}, \tilde{S}_{i,X}^{(t-i+1)}\right) + \sigma\left(S_i, \tilde{S}_i^{(t-i+1)}\right) \tag{16}
$$

*where the optimal combined error is represented by*

$$
\sigma(S_i, \tilde{S}_i^{(t-i+1)}) = \mathrm{R}'(h_i^*, h_i, S_i)
$$
$$
+ \mathrm{R}'(h_i, \tilde{h}_i^{(t-i+1)}, \tilde{S}_i^{(t-i+1)}) \tag{17}
$$

*and*

$$
\mathrm{R}'(h_i^*, h_i, S_i) = \mathbb{E}_{\mathbf{x}\sim S_{i,X}}\tau(h_i^*(\mathbf{x}), h_i(\mathbf{x})), \tag{18}
$$

*where $h_i^*$ is the true labeling function for $S_i$.*

We provide the proof in the Appendix-A from SM. This theorem provides a way to measure the gap on the risk bound for the model, after learning the $t$-th task, but does not provide any insight on how the previously learnt knowledge is forgotten. The following theorem provides an explicit way to measure the accumulated errors when learning a certain task.

**Theorem 2** *Let $\tilde{S}_i^{(t-i+1)}$ be the joint distribution over $\mathcal{X} \times \mathcal{Y}$, and $\tau$ be a loss function which satisfies Assumption 1. The accumulated errors in the knowledge associated with a previously learnt, $i$-th task, after learning a given $t$-th task, can be defined as :*

$$
\mathrm{R}(h, S_i) \leq \mathrm{R}'(h, \tilde{h}_i^{(t-i+1)}, \tilde{S}_i^{(t-i+1)}) +
$$
$$
\sum_{k=0}^{t-i} \left( \Psi(\tilde{S}_{i,X}^{(k)}, \tilde{S}_{i,X}^{(k+1)}) + \sigma(\tilde{S}_i^{(k)}, \tilde{S}_i^{(k+1)}) \right), \tag{19}
$$

*where the last term of the right hand side (RHS) is expressed as :*

$$
\sigma(\tilde{S}_i^{(k)}, \tilde{S}_i^{(k+1)}) = \mathrm{R}'(\tilde{h}_i^{(k)}, \tilde{h}_i^{*(k)}, \tilde{S}_i^{(k)})
$$
$$
+ \mathrm{R}'(\tilde{h}_i^{(k)}, \tilde{h}_i^{(k+1)}, \tilde{S}_i^{(k+1)}), \tag{20}
$$

where we use $\tilde{S}_i^{(0)}$ to represent $S_i$ for simplicity. The proof is provided in the Appendix-B from SM. Theorem 2 provides the analysis of forgetfulness of the previously learnt knowledge in the model $\mathcal{M}(\theta^t, \varsigma^t, \varphi^t)$, while learning the $t$-th task. When $i$ is small, *i.e.* a task which was learnt during one of the initial training stages, then the accumulated terms $\sum_{k=0}^{t-i} \tau(\tilde{S}_{i,X}^{(k)}, \tilde{S}_{i,X}^{(k+1)}) + \sigma(\tilde{S}_i^{(k)}, \tilde{S}_i^{(k+1)})$ lead to larger errors. This explains that $\mathcal{M}(\theta^t, \varsigma^t, \varphi^t)$ would tend to forget the tasks learnt earlier during its lifelong learning process. We visualize this forgetfulness process in Figure 1.

**Lemma 1** *Let us consider that we have Assumption 1, then the accumulated error after learning the probabilistic representations of all databases after $t$-th task learning is:*

$$\sum_{i=1}^{t} \mathrm{R}(h, S_i) \leq \sum_{i=1}^{t} \Big( \mathrm{R}'(h, \tilde{h}_i^{(t-i+1)}, \tilde{S}_i^{(t-i+1)}) +$$
$$\sum_{k=0}^{t-i} \Big( \Psi(\tilde{S}_{i,X}^{(k)}, \tilde{S}_{i,X}^{(k+1)}) + \sigma(\tilde{S}_i^{(k)}, \tilde{S}_i^{(k+1)}) \Big) \Big). \tag{21}$$

We consider Theorem 2 and sum up the accumulated errors from Eq. (19) for learning $t$ tasks and this results in Eq. (21) (Appendix-C from SM). Lemma 1 shows that minimizing the discrepancy distance $\Psi(\tilde{S}_{i,X}^{(k)}, \tilde{S}_{i,X}^{(k+1)})$ between the generated distribution approximated by the model and the target distributions, when learning each task, plays an important role in the improvement of the performance. However, the accumulated errors of the model will increase significantly when increasing the number of new tasks to be learnt. The following lemma shows how a mixture or ensemble model can address this problem and improve the performance.

**Lemma 2** *Let us consider Assumption 1 and assume that we are training the LIMix model with $K$ components onto the $t$-th task learning. If $K = t$, then the accumulated errors of the infinite mixture model for all tasks is defined as:*

$$\sum_{i=1}^{t} \mathrm{R}(h, S_i) \leq \sum_{i=1}^{t} \Big( \mathrm{R}'(h, \tilde{h}_i^1, \tilde{S}_i^1) + \Psi(S_{i,X}, \tilde{S}_{i,X}^1) + \sigma(S_i, \tilde{S}_i^1) \Big). \tag{22}$$

We provide the proof in Appendix-D from SM. Lemma 2 provides the framework for an optimal solution for LIMix in which the lifelong learning problem is transformed into a generalization problem under multiple target-source domains where there is no forgetting error during the training. $h$ is implemented by the mixture of $\{h_{\zeta_1}, \ldots, h_{\zeta_K}\}, h_{\zeta_i} \in \mathcal{H}, i = 1, \ldots, K$ in LIMix and therefore the performance on each target domain is relying on the generalization ability of the associated component. In practice, the number of components is smaller than the number of tasks being learnt. We investigate a specific case in Appendix-D from

SM. The following lemma provides an analysis of the relationship between the model's performance and complexity.

**Lemma 3** *Let $B = \{b_1, \ldots, b_j\}$ represent the labels for the tasks that the corresponding distributions $\{\tilde{S}_{b_1}^{(1)}, .., \tilde{S}_{b_j}^{(1)}\}$ are accessed only once after lifelong learning. Let $B' = \{b_1', \ldots, b_n'\}$ indicate which task was used for re-training more than once. We also define a set $\hat{B} = \{\hat{b}_1, \ldots, \hat{b}_n\}$ that records how many times each task was used for re-training, where $\hat{b}_i > 1$ represents that the $b_i'$-th task has been retrained for $(\hat{b}_i - 1)$ times $\tilde{S}_{b_i'}^{(1)} \to \tilde{S}_{b_i'}^{(\hat{b}_i)}$ where $\tilde{S}_{b_i'}^{(\hat{b}_i)}$ represents the corresponding probabilistic representation. For a given mixture model, we have :*

$$\sum_{i=1}^{t} \mathrm{R}(h, S_i) \leq \sum_{i=1}^{\mathrm{card}(B)} \Big( \mathrm{R}'(h, \tilde{h}_{b_i}^1, \tilde{S}_{b_i}^1) + \Psi(S_{b_i,X}, \tilde{S}_{b_i,X}^1)$$
$$+ \sigma(S_{b_i}, \tilde{S}_{b_i}^1) \Big) + \sum_{i=1}^{\mathrm{card}(B')} \Big( \mathrm{R}'(h, \tilde{h}_{b_i'}^{(t-\hat{b}_i+1)}, \tilde{S}_{b_i'}^{(t-\hat{b}_i+1)})$$
$$+ \sum_{k=0}^{\hat{b}_i-1} \Big( \Psi(\tilde{S}_{b_i',X}^k, \tilde{S}_{b_i',X}^{(k+1)}) + \sigma(\tilde{S}_{b_i'}^k, \tilde{S}_{b_i'}^{(k+1)}) \Big) \Big) = \mathrm{R}_{\mathrm{mixture}} \tag{23}$$

*where $t > 1$ represents the number of tasks being learnt. $\mathrm{card}(\cdot)$ denotes the cardinal number in the set which satisfies $\mathrm{card}(B) + \mathrm{card}(B') > K$ and $0 \leq \mathrm{card}(B) \leq K, 0 \leq \mathrm{card}(B') \leq K, \mathrm{card}(B') = \mathrm{card}(\hat{B})$, where $K$ is the number of components used for training. The risk for a single learnt model $M(\theta^t, \omega^t, \psi^t)$ is defined as $\mathrm{R}_{\mathrm{single}}$, as in Eq. (21), while the risk for the mixture model, $\mathrm{R}_{\mathrm{mixture}}$ is the RHS of Eq. (23). From all these expressions we have $\mathrm{R}_{\mathrm{single}} \geq \mathrm{R}_{\mathrm{mixture}}$.*

The proof is in the Appendix-E from SM. Lemma 3 does not explicitly indicate which task is associated with the information recorded by a specific component of the LIMix model. Nevertheless, we provide an explicit way to analyse the risk bound for the mixture model in a more practical way, where we vary the number of components and the number of times the model was trained for each task, according to different learning settings such as by considering the learning order of tasks or their complexity. If $B' = \varnothing$, then Eq. (23) is reduced to Eq. (22), meaning a smaller gap in the risk bound, while requiring additional memory. On the other hand, if $\mathrm{card}(B) = 1 \Rightarrow B = \{t\}$, the RHS of Eq. (23) becomes equal to that from Eq. (21), meaning a large gap on the risk bound. We define the ratio $v = (K - \mathrm{card}(B))/(K - \mathrm{card}(B'))$ as an index which explains the trade-off between the model complexity and its performance. When $v$ increases, the model improves its performance while also increasing its complexity. In contrast, when $v$ is small, the model's complexity is reduced

| Datasets | MSE | | | | | SSMI | | | | | PSNR | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | LGM | CURL | BE | LIMix | Stud | LGM | CURL | BE | LIMix | Stud | LGM | CURL | BE | LIMix | Stud |
| MNIST | 129.93 | 211.21 | 19.24 | 26.66 | 176.82 | 0.45 | 0.46 | 0.92 | 0.88 | 0.42 | 14.52 | 13.27 | 22.57 | 21.09 | 13.72 |
| Fashion | 89.28 | 110.60 | 38.81 | 30.19 | 178.04 | 0.51 | 0.44 | 0.61 | 0.76 | 0.37 | 15.82 | 14.89 | 14.46 | 21.25 | 8.81 |
| SVHN | 169.55 | 102.06 | 39.57 | 35.07 | 146.70 | 0.24 | 0.26 | 0.66 | 0.65 | 0.47 | 8.11 | 10.86 | 18.90 | 14.92 | 13.58 |
| IFashion | 432.90 | 115.29 | 36.52 | 30.14 | 158.18 | 0.26 | 0.54 | 0.75 | 0.79 | 0.43 | 9.04 | 15.51 | 19.32 | 20.26 | 14.17 |
| RMNIST | 130.28 | 279.47 | 25.41 | 22.80 | 157.55 | 0.45 | 0.29 | 0.88 | 0.90 | 0.43 | 14.51 | 10.84 | 21.31 | 21.81 | 14.18 |
| Average | 190.38 | 163.72 | 31.91 | **28.97** | 163.45 | 0.38 | 0.39 | 0.76 | **0.79** | 0.42 | 12.40 | 13.07 | 19.31 | **19.86** | 12.89 |

Table 1. The performance of various models after the MSFIR lifelong learning.

while it accumulates more error terms when learning new tasks. In the Appendix-G from SM, we use the proposed theoretical framework to analyse the forgetting behaviour for various models such as the classical GRM model [41], a mixture model enabled with expansion mechanisms [34], an ensemble model from [47] and an episodic memory model [25, 30]. We also extend Lemmas 2 and 3 for analyzing the risk bounds for a model when changing the order of tasks in the Appendix-H from SM.

## 5. Experiments

### 5.1. Datasets and evaluation criteria

We consider the following experimental settings :

- For the unsupervised learning setting we create a sequence of learning tasks corresponding to the databases: MNIST [19], SVHN [27], Fashion [49], InverseFashion (IFashion) and Rated MNIST (RM-NIST), and this learning setting is named MSFIR.

- We add CIFAR10 [18] after MSFIR, as the last training task, resulting in MSFIRC sequence for supervised classification. All images are resized to $32 \times 32 \times 3$.

**Evaluation criteria:** In the classification tasks, we use the average accuracy over all tasks as the performance criterion. Although the proposed theoretical analysis is only used in prediction tasks, LIMix can also achieve good performance in the unsupervised setting where we use the Mean Squared Error (MSE), the structural similarity index measure (SSIM) [13] and the Peak-Signal-to-Noise Ratio (PSNR) [13] for the reconstruction quality evaluation.

### 5.2. Unsupervised learning tasks

We firstly evaluate various methods on MSFIR lifelong learning tasks and the results are provided in Table 1. We compare our proposed LIMix model with three state of the art methods: LGM [33], CURL [34] and BatchEnsemble (BE) [47]. BE is designed to be used for classification tasks and we implement BE as an ensemble made up of VAE components, where each VAE has a tuple of trainable vectors built on the top layer of a neural network which does not update its parameters in the $k$-th task learning, $k = \{2, 3, \ldots, K\}$. We use large neural networks, containing more parameters, for LGM and BE, respectively, in

| Dataset | LGM [33] | CURL [34] | BE[47] | LIMix | MRGANs [48] |
|---|---|---|---|---|---|
| MNIST | 90.54 | 91.30 | 99.40 | 91.16 | 91.24 |
| SVHN | 22.56 | 62.05 | 74.46 | 82.60 | 64.12 |
| Fashion | 68.29 | 79.18 | 88.95 | 89.14 | 80.10 |
| IFashion | 73.70 | 82.51 | 86.45 | 88.70 | 82.19 |
| RMNIST | 90.52 | 98.56 | 99.10 | 98.80 | 98.30 |
| CIFAR10 | 57.43 | 67.34 | 52.48 | 54.66 | 67.19 |
| Average | 67.17 | 80.16 | 83.47 | **84.18** | 80.52 |

Table 2. Classification accuracy of various models after the MS-FIRC lifelong learning.

order to ensure fair comparisons. The model size is provided in Appendix-L from SM. "Stud" denotes the performance of the Student model which is worse than LIMix because the Student learns its knowledge from the generation results of LMIX. Fig. 2a shows the absolute difference on the log-likelihood between the incoming task and each component $\mathbf{K}_{i,j}$, as in Eq. (6), and the number of components derived during MSFIR lifelong learning. We can observe that the first component is reused when learning the fifth task (RMNIST) and LIMix expands to 4 components after LLL. We also evaluate the LIMix model when considering more complicated tasks in the Appendix-K.2 from SM.

### 5.3. Classification tasks

In this section we present the results when considering the lifelong learning of classification tasks. In order to use LGM [33] for classification, we continually train an auxiliary classifier on the real data samples from successive tasks and by also using paired data samples $\{\mathbf{x}_i, y_i\}_{i=1}^{n}$, where $\mathbf{x}_i$ is generated by either the Teacher or Student in LGM and each $y_i$ is inferred by the classifier during the last task learning. Table 2 provides the LLL classification accuracy, where it can be observed that LIMix achieves the best results. Unlike in the image reconstruction results, CURL [34] also provides good results on the LLL of classification tasks. CURL uses a single decoder which is continually updated across multiple tasks and therefore leads to poor image reconstructions, which explains the difference between the reconstruction and the classification results for CURL.

Although the proposed LIMix is mainly designed for cross-domain lifelong learning, we also apply LIMix to the continuous learning benchmarks, Permuted MNIST and Split MNIST [58] (see Appendix-K.4 from SM). Similar to [30], we use a smaller network for the implementation of each component and perform five independent runs for cal-
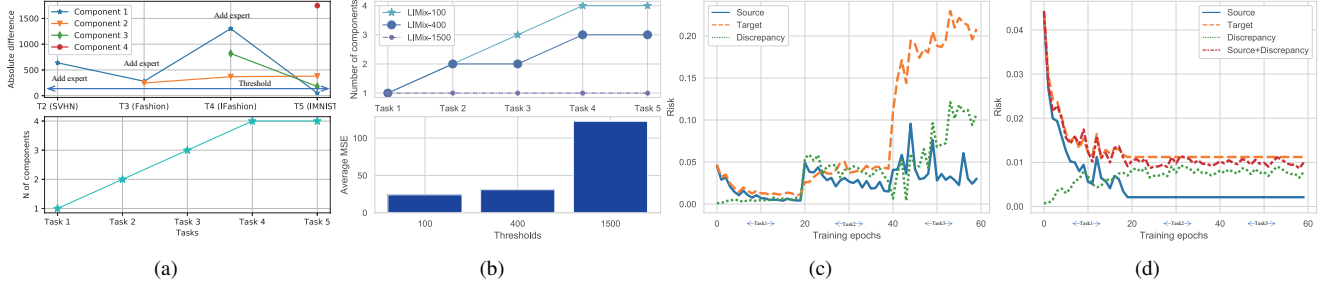
Figure 2. (a) Absolute difference on the log-likelihood and the number of components under the MSFIR lifelong learning. (b) The change of the model's performance and complexity when using different thresholds during MSFIR lifelong learning. (c) The risk and discrepancy for the first task (MNIST) by using LIMix without expansion. (d) The risk and discrepancy for MNIST by using LIMix with expansion.

| Methods | Permuted MNIST | Split MNIST |
|---|---|---|
| DLP* [42] | 82% | 61.2% |
| EWC* [17] | 84% | 63.1% |
| SI* [58] | 86% | 98.9% |
| Improved VCL* [45] | $93.1 \pm 1\%$ | $98.4 \pm 0.4\%$ |
| FRCL-RND* [46] | $94.2 \pm 0.1\%$ | $97.1 \pm 0.7\%$ |
| FRCL-TR* [46] | $94.3 \pm 0.2\%$ | $97.8 \pm 0.7\%$ |
| FROMP* [30] | $94.9 \pm 0.1\%$ | $99.0 \pm 0.1\%$ |
| LIMix | $\mathbf{96.46} \pm 0.03\%$ (10 C) | $\mathbf{99.21} \pm 0.04\%$ (5 C) |
| LIMix | 88.78% (7 C) | 96.77% (4 C) |
| LIMix | 95.25% (8 C) | 91.37% (3 C) |

Table 3. Results of continuous learning benchmark.

culating the mean and standard deviation. The results are provided in Table 3 where "*" means reporting the results from [30] and "5 C" means 5 components for LIMix. LIMix achieves the best performance for the optimal solution, validating Lemma 2, and would gradually loses performance when reducing the model size, as discussed in Lemma 3.

### 5.4. Ablation study and theoretical results

We evaluate the performance of the LIMix model when varying various hyperparameters and thresholds and the results are shown in Figures 2a and 2b. The performance is improved by expanding the model's architecture, but this would increase its complexity, as discussed in Lemma 3. We provide additional results for the hyperparameter parameter setting in the Appendix-K.1 from SM. We investigate the theoretical results for Theorem 2 and train a single model under MNIST, Fashion and SVHN (MFS) learning setting and evaluate the source-risk, target-risk and discrepancy on MNIST and present the results in Fig. 2c. We can observe that the increase of the target-risk largely depends on the discrepancy instead of the source-risk which keeps stable when learning additional tasks. We also investigate the results for LIMix under MFS lifelong learning and the results are provided in Fig. 2d, where "Source + Discrepancy" represents the source-risk plus the discrepancy on MNIST. The discrepancy does not increase for LIMix when learning additional tasks. Besides, "Source + Discrepancy" is very close to the target-risk and the gap to the target-risk is the combined error $\sigma(S_1, \tilde{S}_1^1)$, according to Eq. (19).

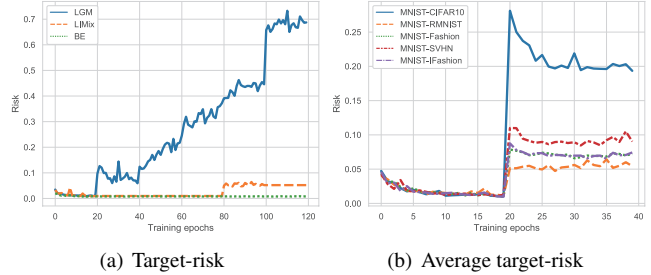We also investigate the target-risk for LIMix, BE, and



Figure 3. Evaluation of MNIST target-risk for other datasets.

LGM and the results are shown in Fig 3a, where the target-risk is evaluated on MNIST for each epoch. The target-risk for BE does not change across the LLL of six tasks, given that it does not accumulate any errors. LGM continually increases its target-risk, which is bounded by $R_{\text{Single}}$, adding an error term after learning each task. LIMix only increases the target-risk when it reuses a component, which was updated when learning RMNIST after MNIST. These results are explained by Lemma 3 and further explored in Appendix-G from SM. We also estimate the average target-risk for all tasks (20 epochs for each task) when reusing a component trained on MNIST for learning a new task and the results are shown in Fig. 3b. We find that a single component would lead to a large degeneration in the performance when learning an entirely different task (CIFAR10) than a related task (RMNIST). This demonstrates that the proposed selection mechanism can choose an appropriate component minimizing the target-risk. Examples of image reconstruction, generation and Image to Image translation are shown in the Appendix-M from SM.

## 6. Conclusion

We propose a new theoretical analysis framework for lifelong learning based on the discrepancy distance between the probabilistic measures of the knowledge already learnt by the model and a target distribution. We provide an insight into how the model forgets some of the knowledge acquired during LLL, through the analysis of the model's risk. Inspired by the analysis, we propose LIMix model which performs better in cross-domain lifelong learning.

# References

[1] Alessandro Achille, Tom Eccles, Loic Matthey, Christopher Burgess, Nick Watters, Alexander Lerchner, and Irina Higgins. Life-long disentangled representation learning with cross-domain latent homologies. In *Advances in Neural Inf. Proc. Systems (NIPS)*, pages 9873–9883, 2018.

[2] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 11817–11826, 2019.

[3] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Inf. Proc Systems*, pages 137–144, 2007.

[4] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K. Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

[5] Tao Chen, Julian Morris, and Elaine Martin. Probability density estimation via an infinite Gaussian mixture model: application to statistical process monitoring. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 55(5):699–715, 2006.

[6] Corinna Cortes, Xavi Gonzalvo, Vitaly Kuznetsov, Mehryar Mohri, and Scott Yang. Adanet: Adaptive structural learning of artificial neural networks. In *Proc. of Int. Conf. on Machine Learning (ICML), vol. PMLR 70*, pages 874–883, 2017.

[7] Corinna Cortes and Mehryar Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theoretical Computer Science*, 519:103–126, 2014.

[8] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[9] Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. In *Proc. of Conf. on Uncertainty in Artificial Intell. (UAI)*, pages 258–267, 2015.

[10] Sayna Ebrahimi, Franziska Meier, Roberto Calandra, Trevor Darrell, and Marcus Rohrbach. Adversarial continual learning. In *Proc. European Conf. on Computer Vision (ECCV), vol. LNCS 12356*, pages 386–402, 2020.

[11] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Inf. Proc. Systems (NIPS)*, pages 2672–2680, 2014.

[12] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *Proc. NIPS Deep Learning Workshop, arXiv preprint arXiv:1503.02531*, 2014.

[13] Alain Hore and Djemel Ziou. Image quality metrics: PSNR vs. SSIM. In *Proc. Int. Conf. on Pattern Recognition (ICPR)*, pages 2366–2369, 2010.

[14] Ghassen Jerfel, Erin Grant, Tom Griffiths, and Katherine A Heller. Reconciling meta-learning and continual learning with online mixtures of tasks. In *Advances in Neural Information Processing Systems*, pages 9122–9133, 2019.

[15] H. Jung, J. Ju, M. Jung, and J. Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016.

[16] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

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

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

[19] Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. of the IEEE*, 86(11):2278–2324, 1998.

[20] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural Dirichlet process mixture model for task-free continual learning. *Proc. International Conference of Learning Representations (ICLR), arXiv preprint arXiv:2001.00689*, 2020.

[21] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. MMD GAN: Towards deeper understanding of moment matching network. In *Advances in Neural Inf. Proc. Systems*, pages 2203–2213, 2017.

[22] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *Proc. Int. Conf. on Machine Learning*, pages 1718–1727, 2015.

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

[24] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, pages 700–708, 2017.

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

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

[27] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

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

[29] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In *Proc. of Int. Conf. on Machine Learning (ICML), vol. PMLR 70*, pages 2642–2651, 2017.

[30] Pingbo Pan, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard Turner, and Mohammad Emtiyaz E Khan. Continual deep learning by functional regularisation of memorable past. In *Advances in Neural Information Processing Systems*, volume 33, pages 4453–4464, 2020.

[31] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

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

[33] Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Lifelong generative modeling. *Neurocomputing*, 404:381–400, 2020.

[34] Dushyant Rao, Francesco Visin, Andrei A. Rusu, Yee W. Teh, Razvan Pascanu, and Raia Hadsell. Continual unsupervised representation learning. In *Advances in Neural Information Proc. Systems (NeurIPS)*, pages 7645–7655, 2019.

[35] Carl Edwards Rasmussen. The infinite Gaussian mixture model. In *Advances in Neural Inf. Proc. Systems (NIPS)*, pages 554–560, 2000.

[36] Carl E Rasmussen and Zoubin Ghahramani. Infinite mixtures of Gaussian process experts. In *Advances in Neural Inf. Proc. systems*, pages 881–888, 2002.

[37] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

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

[39] Mohammad Rostami, Soheil Kolouri, Praveen K Pilly, and James McClelland. Generative continual concept learning. In *Proc. AAAI Conf. on Artificial Intelligence*, pages 5545–5552, 2020.

[40] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[41] Hanul Shin, Jung K. Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Proc. Advances in Neural Inf. Proc. Systems (NIPS)*, pages 2990–2999, 2017.

[42] Alex J Smola, SVN Vishwanathan, and Eleazar Eskin. Laplace propagation. In *Advances in Neural Inf. Proc. Systems (NIPS)*, pages 441–448, 2004.

[43] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Inf. Proc. Systems (NIPS)*, pages 3483–3491, 2015.

[44] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U. Gutmann, and Charles Sutton. VEEGAN: Reducing mode collapse in gans using implicit variational learning. In *Advances in Neural Inf. Proc. Systems (NIPS)*, pages 3308–3318, 2017.

[45] Siddharth Swaroop, Cuong V Nguyen, Thang D Bui, and Richard E Turner. Improving and understanding variational continual learning. In *Proc. NIPS-workshops Continual Learning, arXiv preprint arXiv:1905.02099*, 2018.

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

[47] Yeming Wen, Dustin Tran, and Jimmy 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.

[48] Chenshen Wu, Luis Herranz, Xialei Liu, Joost van de Weijer, and Bogdan Raducanu. Memory replay GANs: Learning to generate new categories without forgetting. In *Advances In Neural Inf. Proc. Systems (NIPS)*, pages 5962–5972, 2018.

[49] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[50] Tianjun Xiao, Jiaxing Zhang, K. Yang, Yuxin Peng, and Zheng Zhang. Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *Proc. of ACM Int. Conf. on Multimedia*, pages 177–186, 2014.

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

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

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

[54] Fei Ye and Adrian G. Bors. Deep mixture generative autoencoders. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[55] Fei Ye and Adrian G. Bors. Lifelong mixture of variational autoencoders. *IEEE Trans. on Neural Networks and Learning Systems*, 2021.

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

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

[58] F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. In *Proc. of Int. Conf. on Machine Learning, vol. PLMR 70*, pages 3987–3995, 2017.

[59] Guanyu Zhou, Kihyuk Sohn, and Honglak Lee. Online incremental feature learning with denoising autoencoders. In *Proc. Artif. Intell. and Statistics (AISTATS), vol. PMLR 22*, pages 1453–1461, 2012.