Supplementary materials for the paper "Training a Dynamic Growing Mixture Model for Lifelong Learning"

Fei Ye and Adrian G. Bors

Appendix A
Proof of Theorem 1

We adopt similar derivations to Theorem 8 from Domain Adaptation theory [1]. We consider fixing the classifier $h \in \mathcal{H}$. According to the triangle inequality property of $\mathcal{L}(\cdot,\cdot)$ (Definition 4) and the definition of the discrepancy distance $\mathcal{L}_d(\cdot,\cdot)$ form, we have [1]:

$$\begin{aligned}\mathcal{L}_{S_i}(h, f_{S_i}) &\le \mathcal{L}_{S_i}(h, h_{\tilde{S}_i^{(t-i)}}) + \mathcal{L}_{S_i}(h_{S_i}, h_{\tilde{S}_i^{(t-i)}}) \\ &\quad + \mathcal{L}_{S_i}(h_{S_i}, f_{S_i}) \\ &\le \mathcal{L}_{\tilde{S}_i^{(t-i)}}\big(h, f_{\tilde{S}_i^{(t-i)}}\big) + \mathcal{L}_d\big(S_{i,X}, \tilde{S}_{i,X}^{(t-i)}\big) \\ &\quad + \mathcal{L}_{S_i}(f_{S_i}, h_{S_i}) + \mathcal{L}_{\tilde{S}_i^{(t-i)}}(h_{S_i}, h_{\tilde{S}_i^{(t-i)}})\,.\end{aligned} \tag{1}$$

Then we rewrite Eq. (1) as :

$$\begin{aligned}\mathcal{L}_{S_i}\big(h, f_{S_i}\big) &\le \mathcal{L}_{\tilde{S}_i^{(t-i)}}\big(h, f_{\tilde{S}_i^{(t-i)}}\big) + \mathcal{L}_{\tilde{S}_i^{(t-i)}}\big(h, h_{\tilde{S}_i^{(t-i)}}\big) \\ &\quad - \mathcal{L}_{\tilde{S}_i^{(t-i)}}\big(h, f_{\tilde{S}_i^{(t-i)}}\big) \\ &\quad + \mathcal{L}_{S_i}(f_{S_i}, h_{S_i}) + \mathcal{L}_{\tilde{S}_i^{(t-i)}}(h_{S_i}, h_{\tilde{S}_i^{(t-i)}}) \\ &\quad + \mathcal{L}_d\big(S_{i,X}, \tilde{S}_{i,X}^{(t-i)}\big)\end{aligned} \tag{2}$$

□

This proves Theorem 1.

Appendix B
Proof of Theorem 2

Firstly, we take $\tilde{S}_i^{(t-i)}$ and $\tilde{S}_i^{(t-i-1)}$ to be the source and the target distribution and we then have a bound, according to *Theorem 1*:

$$\begin{aligned}&\mathcal{L}_{\tilde{S}_i^{(t-i-1)}}\big(h, f_{\tilde{S}_i^{(t-i-1)}}\big) \le \mathcal{L}_{\tilde{S}_i^{(t-i)}}\big(h, f_{\tilde{S}_i^{(t-i)}}\big) \\ &\quad + \mathcal{L}_d\big(\tilde{S}_{i,X}^{(t-i-1)}, \tilde{S}_{i,X}^{(t-i)}\big) + f'\big(S_i^{(t-i-1)}, \tilde{S}_i^{(t-i)}\big).\end{aligned} \tag{3}$$

Furthermore, we take $\tilde{S}_i^{(t-i-1)}$ and $\tilde{S}_i^{(t-i-2)}$ to be the source and the target distribution and we then have a bound:

$$\begin{aligned}&\mathcal{L}_{\tilde{S}_i^{(t-i-2)}}\big(h, f_{\tilde{S}_i^{(t-i-2)}}\big) \le \mathcal{L}_{\tilde{S}_i^{(t-i-1)}}\big(h, f_{\tilde{S}_i^{(t-i-1)}}\big) \\ &\quad + \mathcal{L}_d\big(\tilde{S}_{i,X}^{(t-i-2)}, \tilde{S}_{i,X}^{(t-i-1)}\big) + f'\big(S_i^{(t-i-2)}, \tilde{S}_i^{(t-i-1)}\big).\end{aligned} \tag{4}$$

$\tilde{S}_i^{(t-i-2)}$ is statistically closer to the distribution of the training set of the i-th task when comparing with $\tilde{S}_i^{(t-i-1)}$ due to the GRM process (See details in Definition 4 of the paper). So the risk bound is reasonable for Eq.(4). Similarly, we consider $\tilde{S}_i^{t-i-2}$ and $\tilde{S}_i^{t-i-3}$ as the source and target distribution and by the mathematical induction, we have the following risk bounds :

$$\begin{aligned}&\mathcal{L}_{\tilde{S}_i^{(t-i-3)}}\big(h, f_{\tilde{S}_i^{(t-i-3)}}\big) \le \mathcal{L}_{\tilde{S}_i^{(t-i-2)}}\big(h, f_{\tilde{S}_i^{(t-i-2)}}\big) \\ &\quad + \mathcal{L}_d\big(\tilde{S}_{i,X}^{(t-i-3)}, \tilde{S}_{i,X}^{(t-i-2)}\big) + f'\big(S_i^{(t-i-3)}, \tilde{S}_i^{(t-i-2)}\big) \\ &\mathcal{L}_{\tilde{S}_i^{(t-i-4)}}\big(h, f_{\tilde{S}_i^{(t-i-4)}}\big) \le \mathcal{L}_{\tilde{S}_i^{(t-i-3)}}\big(h, f_{\tilde{S}_i^{(t-i-3)}}\big) \\ &\quad + \mathcal{L}_d\big(\tilde{S}_{i,X}^{(t-i-4)}, \tilde{S}_{i,X}^{(t-i-3)}\big) + f'\big(S_i^{(t-i-4)}, \tilde{S}_i^{(t-i-3)}\big) \\ &\dots \\ &\dots \\ &\mathcal{L}_{\tilde{S}_i^{(0)}}\big(h, f_{\tilde{S}_i^{(0)}}\big) \le \mathcal{L}_{\tilde{S}_i^{(1)}}\big(h, f_{\tilde{S}_i^{(1)}}\big) \\ &\quad + \mathcal{L}_d\big(\tilde{S}_{i,X}^{(0)}, \tilde{S}_{i,X}^{(1)}\big) + f'\big(S_i^{(0)}, \tilde{S}_i^{(1)}\big) \\ &\mathcal{L}_{\tilde{S}_i^{(-1)}}\big(h, f_{\tilde{S}_i^{(-1)}}\big) \le \mathcal{L}_{\tilde{S}_i^{(0)}}\big(h, f_{\tilde{S}_i^{(0)}}\big) \\ &\quad + \mathcal{L}_d\big(\tilde{S}_{i,X}^{(-1)}, \tilde{S}_{i,X}^{(0)}\big) + f'\big(S_i^{(-1)}, \tilde{S}_i^{(0)}\big),\end{aligned} \tag{5}$$

where $\tilde{S}_i^{(-1)} = S_i$ and $\tilde{S}_i^{(0)}$ represent the distribution of the testing set and the training set of the (i)-th task, respectively. $h_{\tilde{S}_i^{(-1)}}$ is defined by $h_{\tilde{S}_i^{(-1)}} = \arg\min_{h_{\tilde{S}_i^{(-1)}} \in \mathcal{H}} \mathcal{L}_{\tilde{S}_i^{(-1)}}(h, \tilde{S}_i^{(-1)})$. Then we sum up the right-hand and left-hand sides of all equations (Eq. (3), Eq. (4) and Eq. (5)), resulting in :

$$\begin{aligned}&\mathcal{L}_{\tilde{S}_i^{(-1)}}\big(h, f_{\tilde{S}_i^{(-1)}}\big) \le \mathcal{L}_{\tilde{S}_i^{(t-i)}}\big(h, f_{\tilde{S}_i^{(t-i)}}\big) \\ &\quad + \sum_{k=-1}^{t-i-1} \Big(\mathcal{L}_d(\tilde{S}_{i,X}^{(k)}, \tilde{S}_{i,X}^{(k+1)}) + f'(\tilde{S}_i^{(k)}, \tilde{S}_i^{(k+1)})\Big).\end{aligned} \tag{6}$$

Since we have $\tilde{S}_i^{(-1)} = S_i$, we can rewrite Eq. (6) as :

$$\begin{aligned}&\mathcal{L}_{S_i}\big(h, f_{S_i}\big) \le \mathcal{L}_{S_i^{(t-i)}}\big(h, f_{\tilde{S}_i^{(t-i)}}\big) \\ &\quad + \sum_{k=-1}^{t-i-1} \Big(\mathcal{L}_d(\tilde{S}_{i,X}^{(k)}, \tilde{S}_{i,X}^{(k+1)}) + f'(\tilde{S}_i^{(k)}, \tilde{S}_i^{(k+1)})\Big)\end{aligned} \tag{7}$$

□

Appendix C
The proof of Lemma 3

Firstly, we consider the risk bound for the tasks that are trained only once, we have :

$$\begin{aligned}&\sum_{i=1}^{\mathrm{card}(B)} \mathcal{L}_{S_{b_i}}\Big(h, h_{\zeta_{f_t(b_i)}}\Big) \leq \\ &\sum_{i=1}^{\mathrm{card}(B)} \Big\{\mathcal{L}_{\tilde{S}^{(0)}_{b_i}}\big(h_{\zeta_{f_t(b_i)}}, f_{\tilde{S}^{(0)}_{b_i}}\big) + f'\big(S_{b_i}, \tilde{S}^{(0)}_{b_i}\big) \\ &+ \mathcal{L}_d\big(S_{b_i,X}, \tilde{S}^{(0)}_{b_i,X}\big)\Big\}.\end{aligned} \tag{8}$$

Then we derive the risk bound for the tasks that are trained more than once and therefore have accumulated error terms, we have :

$$\begin{aligned}&\sum_{i=1}^{\mathrm{card}(B')} \mathcal{L}_{S_{b'_i}}\Big(h_{\zeta_{f_t(b'_i)}}, f_{S_{b'_i}}\Big) \leq \\ &\sum_{i=1}^{\mathrm{card}(B')} \left(\begin{array}{l}\mathcal{L}_{\tilde{S}^{(t-\hat{b}_i)}_{b'_i}}(h_{\zeta_{f_t(b'_i)}}, f_{\tilde{S}^{(t-\hat{b}_i)}_{b'_i}}) + \\ \sum\limits_{k=-1}^{\hat{b}_i-1}\Big(\mathcal{L}_d(\tilde{S}^k_{b'_i,X}, \tilde{S}^{(k+1)}_{b'_i,X}) + f'(\tilde{S}^k_{b'_i}, \tilde{S}^{(k+1)}_{b'_i})\Big).\end{array}\right)\end{aligned} \tag{9}$$

Then we sum up Eq. (8) and Eq. (9), resulting in :

$$\begin{aligned}&\sum_{i=1}^{t}\Big\{\mathcal{L}_{S_i}\left(h_{\zeta_i}, S_i\right)\Big\} \leq \sum_{i=1}^{\mathrm{card}(B)}\Big\{\mathcal{L}_{\tilde{S}^{(0)}_{b_i}}\big(h_{\zeta_{f_t(b_i)}}, f_{\tilde{S}^{(0)}_{b_i}}\big) \\ &+ f'\big(S_{b_i}, \tilde{S}^{(0)}_{b_i}\big) + \mathcal{L}_d\big(S_{b_i,X}, \tilde{S}^{(0)}_{b_i,X}\big)\Big\} \\ &+ \sum_{i=1}^{\mathrm{card}(B')}\Big\{\mathcal{L}_{\tilde{S}^{(t-\hat{b}_i)}_{b'_i}}(h_{\zeta_{f_t(b'_i)}}, f_{\tilde{S}^{(t-\hat{b}_i)}_{b'_i}}) \\ &+ \sum_{k=-1}^{\hat{b}_i-1}\Big(\mathcal{L}_d(\tilde{S}^k_{b'_i,X}, \tilde{S}^{(k+1)}_{b'_i,X}) + f'(\tilde{S}^k_{b'_i}, \tilde{S}^{(k+1)}_{b'_i})\Big)\Big\},\end{aligned} \tag{10}$$

where we employ $\sum_{i=1}^{t}\Big\{\mathcal{L}_{S_i}\left(h_{\zeta_i}, S_i\right)\Big\}$ to represent the summation of the left hand sides of Eq. (8) and Eq. (9).

In the following, we prove $\mathcal{R}_{\text{single}} \geq \mathcal{R}_{\text{mixture}}$. We consider that the mixture model $\mathbf{M}$ only has a single component and we can derive the risk bound for all t tasks as :

$$\begin{aligned}\mathcal{R}_{\text{single}} = {}& \mathcal{L}_{\tilde{S}^{(0)}_t}\big(h_{\zeta_1}, f_{\tilde{S}^{(0)}_t}\big) + f'\big(S_t, \tilde{S}^{(0)}_t\big) + \mathcal{L}_d\big(S_{t,X}, \tilde{S}^{(0)}_{t,X}\big) \\ &+ \sum_{i=1}^{t-i}\Big\{\mathcal{L}_{\tilde{S}^{(t-i)}_i}(h_{\zeta_1}, f_{\tilde{S}^{(t-i)}_i}) \\ &+ \sum_{k=-1}^{t-i-1}\Big(\mathcal{L}_d(\tilde{S}^k_{i,X}, \tilde{S}^{(k+1)}_{i,X}) + f'(\tilde{S}^k_i, \tilde{S}^{(k+1)}_i)\Big)\Big\}.\end{aligned} \tag{11}$$

Then we consider an extreme case for the mixture model in which the number of components is only two. The first component learns the first task and is fixed in the following task learning. We then derive $\mathcal{R}_{\text{mixture}}$ as :

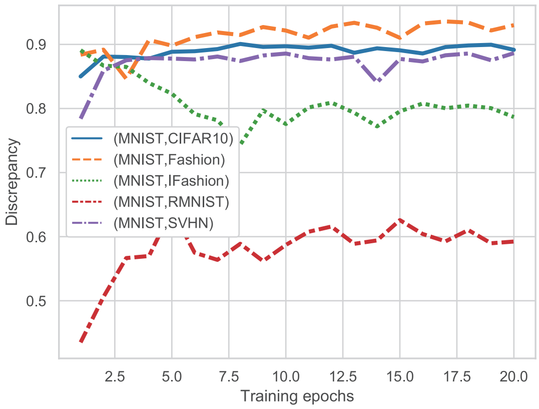


Fig. 1. The estimation of the discrepancy distance between different domains.

$$\begin{aligned}\mathcal{R}_{\text{mixture}} = {}& \mathcal{L}_{\tilde{S}^{(0)}_1}\big(h_{\zeta_1}, f_{\tilde{S}^{(0)}_1}\big) + f'\big(S_t, \tilde{S}^{(0)}_1\big) + \mathcal{L}_d\big(S_{1,X}, \tilde{S}^{(0)}_{1,X}\big) \\ &+ \mathcal{L}_{\tilde{S}^{(0)}_t}\big(h_{\zeta_2}, f_{\tilde{S}^{(0)}_t}\big) + f'\big(S_t, \tilde{S}^{(0)}_t\big) + \mathcal{L}_d\big(S_{t,X}, \tilde{S}^{(0)}_{t,X}\big) \\ &+ \sum_{i=2}^{t-i}\Big\{\mathcal{L}_{\tilde{S}^{(t-i)}_i}(h_{\zeta_2}, f_{\tilde{S}^{(t-i)}_i}) \\ &+ \sum_{k=-1}^{t-i-1}\Big(\mathcal{L}_d(\tilde{S}^k_{i,X}, \tilde{S}^{(k+1)}_{i,X}) + f'(\tilde{S}^k_i, \tilde{S}^{(k+1)}_i)\Big)\Big\}.\end{aligned} \tag{12}$$

where the third and fourth rows in Eq. (12) are the risk bounds used for the following tasks. It clearly sees that Eq. (11) $\geq$ Eq. (12) since the first task in Eq. (11) has more accumulated errors while the first task in Eq. (12) does not suffer from forgetting.

Appendix D
Pseudocode for Algorithm 1

The pseudocode of the GMM under unsupervised learning is provided in Algorithm 1.

Appendix E
Pseudocode for Algorithm 2

The pseudocode of the training of the Teacher-Student model is provided in Algorithm 2.

Appendix F
More ablation studies

Additionally, we investigate whether the proposed component selection process can determine the appropriate component that does not suffer from significant degeneration. We consider training two classifiers : one classifier h on MNIST and the second classifier h' on a joint dataset $(MNIST, A)$, where A is another dataset. Then we estimate the discrepancy distance between MNIST and A by using Eq. (2) from the paper, and evaluate the outputs of the two classifiers $\{h, h'\}$. We present the results in Fig. 1 where $(MNIST, Fashion)$ represents the discrepancy distance between MNIST and Fashion databases. From Fig. 1, we observe that the discrepancy

Algorithm 1: Unsupervised learning of GMM

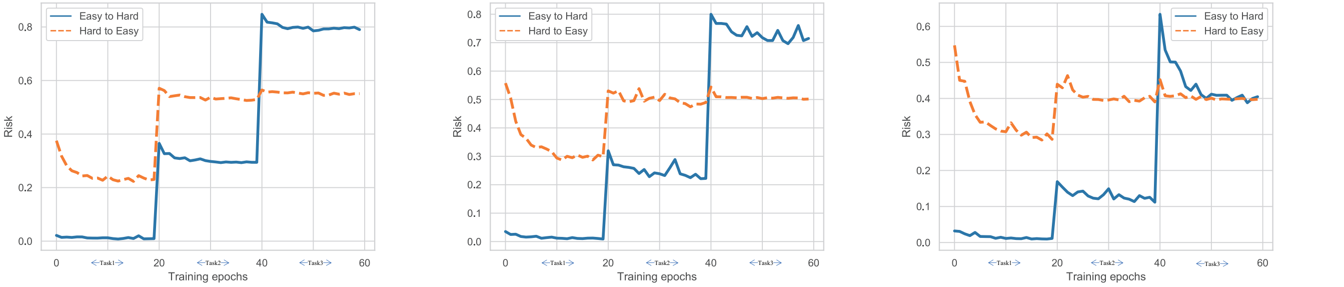
```
(Input:All training databases);
for $i < taskCount$ do
    if  $i$ == 1 then
        isAdd = True ;
    end
    $D_i^S$ Get the training set ;
    if isAdd == False then
        Generate dataset $D'$ from the $s$-th component ;
        $D_i^S = D_i^S \bigcup D'$ Form a joint dataset ;
    end
    else
        Build a new component ;
    end
    for $j < iterationNumber$ do
        $\mathbf{X}' \sim D_i^S$ data batch ;
        if isAdd == False then
            Update the selected component with $\mathbf{X}'$
              using $\mathcal{L}^{s}_{ELBO}(\mathbf{x};\theta,\omega)$ ;
            else
                Update the new component with $\mathbf{X}'$
                  using $\mathcal{L}^{K+1}_{ELBO}(\mathbf{x};\theta,\omega)$ ;
            end
        end
    end
    The evaluation of knowledge measure ;
    Calculate the knowledge measure using Eq.(17) of
      the paper;
    if $\min\{\mathrm{F}_{\mathcal{K}}(\mathcal{M}_j,\mathcal{T}_{i+1}); j=1,\cdots,K\} \geq \lambda$ , then
        isAdd = True;
        else
            Select a component ;
            $s = \operatorname*{arg\,min}_{i=1,\cdots,K} \{\mathrm{F}_{\mathcal{K}}(\mathcal{M}_i,\mathcal{T}_{t+1})\}$ ;
            isAdd = False;
        end
    end
end
```

Algorithm 2: Algorithm for the Teacher-Student

```
Input:All training databases);
for $i < taskCount$ do
    if  $i$ == 1 then
        isAdd = True ;
        else
            $D_i^S$ Get the training set ;
        end
        if isAdd == False then
            Generate dataset $D'$ from the $s$-th
              component ;
            $D_i^S = D_i^S \bigcup D'$ Form a joint dataset ;
            else
                Build a new component ;
            end
        end
    end
    for $j < batchCount$ do
        $\mathbf{X}' \sim D_i^s$ data batch ;
        if isAdd == False then
            Train the selected component with $\mathbf{X}'$
              using $\mathcal{L}^{s}_{ELBO}(\mathbf{x};\theta,\omega)$ ;
        end
        else
            Train the new component with $\mathbf{X}'$ using
              $\mathcal{L}^{K+1}_{ELBO}(\mathbf{x};\theta,\omega)$ ;
        end
    end
    Knowledge distillation ;
    Optimize the student module by suing $\mathcal{L}_{stu}$ ;
    The evaluation of knowledge measure ;
    Calculate the knowledge measure using Eq.(17) of
      the paper);
    if $\min\{\mathrm{F}_{\mathcal{K}}(\mathcal{M}_j,\mathcal{T}_{i+1}); j=1,\cdots,K\} \geq \lambda$ , then
        isAdd = True;
        else
            Select a component ;
            $s = \operatorname*{arg\,min}_{i=1,\cdots,K} \{\mathrm{F}_{\mathcal{K}}(\mathcal{M}_i,\mathcal{T}_{t+1})\}$ ;
            isAdd = False;
        end
    end
end
```

distance is small when two tasks are related (for example, MNIST and RMNIST). The proposed component selection procedure reuses the component trained on MNIST to learn a similar dataset, such as RMNIST, demonstrating that the proposed selection procedure can help GMM choose the component with the smallest discrepancy to the new task.

In the following, we investigate the performance of the proposed GMM when changing the task learning order. We train the proposed GMM under MNIST, SVHN and CIFAR10 (MSC) lifelong learning and then we consider learning them in reversed order, as CMS. We show the empirical results in Figures 2-a and 2-b, which indicate that GMM and BE have a significant difference in the learning when changing the order of tasks. This is because GMM and BE share most parameters between the components. However, the shared parameters are frozen after the first task learning, which only preserves the knowledge of the first task. The model's performance for a second task is changed when considering learning a different first task. We also consider VAE components that do not share any parameters with other components in the mixture model. We use the same setting for training this mixture model and the accumulated target risk is shown in Fig. 2-c. We can observe that when the GMM does not share parameters among components, its performance is less sensitive to the change in the order of tasks.



(a) The results from BatchEnsemble.

(b) GMM results when sharing a part of the parameters between components.

(c) The results when the GMM does not share parameters among components.

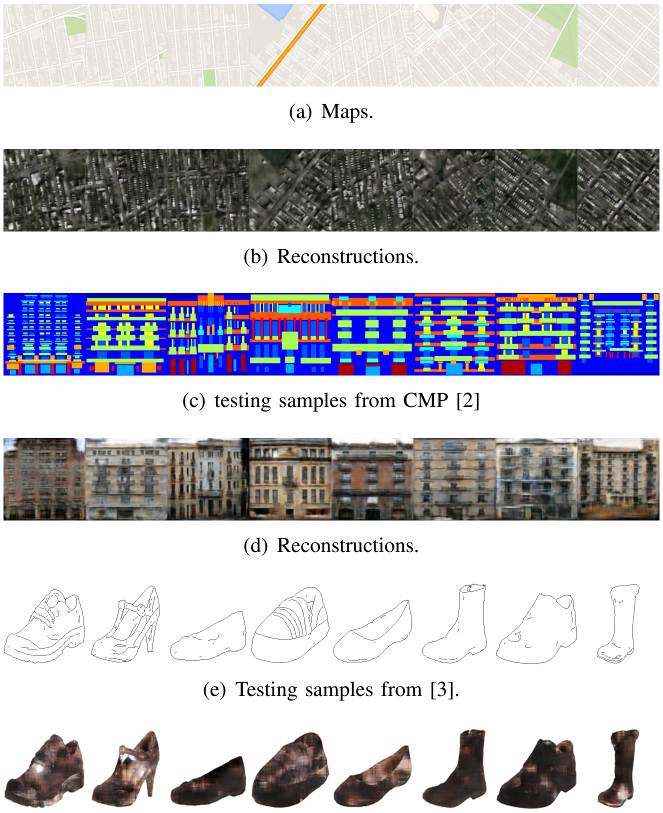
Fig. 2. The target risk on all datasets, achieved by the proposed GMM when learning the sequence of MNIST, SVHN and CIFAR10 databases, namely MSC. We employ 'Easy-to-Hard' and 'Hard-to-Easy' to denote that the model is trained under MSC and CMS lifelong learning, respectively.

TABLE I
THE NUMBER OF PARAMETERS FOR VARIOUS MODELS AFTER THE UNSUPERVISED LEARNING OF MSFIR AND CCCOS.

Datasets	LGM [4]	CURL [5]	BE [6]	GMM	Stud
MSFIR	3.3×10^8	2.3×10^8	3.6×10^8	2.1×10^8	1.4×10^8
CCCOS	1.9×10^9	2.0×10^9	2.0×10^9	7.2×10^8	1.7×10^8

TABLE II
THE NUMBER OF PARAMETERS FOR VARIOUS MODELS UNDER THE LIFELONG SUPERVISED LEARNING OF TASK SEQUENCE MSFIRC.

Datasets	LGM [4]	CURL [5]	BE [6]	GMM	MRGANs [7]
MSFIRC	5.9×10^8	3.3×10^8	3.9×10^8	3.4×10^8	3.3×10^8



(a) Maps.

(b) Reconstructions.

(c) testing samples from CMP [2]

(d) Reconstructions.

(e) Testing samples from [3].

(f) Reconstructions.

Fig. 3. Image to Image translation results when learning three different tasks under the lifelong learning.

APPENDIX G
IMAGE TO IMAGE TRANSLATION TASK

In this section, we apply GMM for the image-to-image translation tasks. We build a sequence of Map, CMP [2] and Shoe [3] datasets. We train GMM on this sequence by using the objective function from Eq. (19) and Eq. (20) from the paper, and the visual results are shown in Fig. 3. We observe that the proposed GMM achieves high-quality image-to-image translation results, by accurately modelling probabilistic correspondences between images from completely different data categories, without forgetting.

APPENDIX H
ANALYZING MODEL'S COMPLEXITY

In the following we evaluate the model size for various methods. The number of parameters for unsupervised learning of MSFIR and CCCOS sequences of tasks is reported in Table VII, where 'Stud' denotes the number of parameters for the Student module of GMM. Meanwhile in Table II we provide the number of parameters for the supervised learning of MSFIRC set of tasks.

APPENDIX I
KNOWLEDGE ASSIMILATION BY THE STUDENT

Recent studies have extended the Knowledge Distillation (KD) as a method used in the continual learning [8]. This is the first study to investigate how a Student could forget information during its lifelong learning from a good Teacher. The proposed knowledge distillation approach, described in Section-E from the paper cannot absorb entirely the knowledge of all previously learned tasks because the information accuracy depends on the generating capacity of each individual component of the Teacher's mixture module. Using the theoretical framework from Section-C from the paper, we explain Student's learning limitations. First, we observe that the Student's loss function, defined by Eq.(27) from the paper, learns the prior information from all components of the GMM Teacher module, while its effectiveness on the target set of prior tasks is inextricably linked to the quality of the approximation distribution which can be generated by each component. Thus, we evaluate a risk bound to assess the forgetting by the Student.

Lemma 1: Suppose that we have trained an optimal GMM in which the count of experts is the same with the number of tasks. The Student is implemented using a classifier $h_s \in \mathcal{H}$, trained by means of knowledge distillation. We derive a risk bound for the Student module during the tth task learning :

$$\sum_{i=1}^{t} \Big\{ \mathcal{L}_{S_i}\left(h_s, f_{S_i}\right) \Big\} \leq \sum_{i=1}^{1} \{ \mathcal{L}_{\tilde{S}_i^{(1)}}\big(h_s, f_{\tilde{S}_i^{(1)}}\big) + f'\big(S_i, \tilde{S}_i^{(1)}\big) \\ + \mathcal{L}_d\big(S_{i,X}, \tilde{S}_{i,\mathcal{X}}^{(1)}\big) \} + \mathcal{L}_{\tilde{S}_t^{(0)}}\big(h_s, f_{\tilde{S}_t^{(0)}}\big) \\ + f'\big(S_t, \tilde{S}_t^{(0)}\big) + \mathcal{L}_d\big(S_{t,X}, \tilde{S}_{t,\mathcal{X}}^{(0)}\big) \,. \tag{13}$$

Using Lemma 2 fro, we find that the optimal GMM can reach a tight risk bound, which is not true for the Student module. The reason for this can be explained by Eq. (13). The Student module is trained with the knowledge learned from each component in the GMM module. However, this knowledge represents the degenerate distributions $\{\tilde{S}^1_{1,\mathcal{X}}, \dots, \tilde{S}^1_{(t-1),\mathcal{X}}\}$, while the Student does not have access to the real training samples from all the previously learned tasks $\{\mathcal{T}_1, \dots, \mathcal{T}_{t-1}\}$. Moreover, the Student module is a static network architecture trained on multiple tasks involving different data domains. Such a static network architecture may also lead to a degraded performance in the target distribution due to the negative backward transfer [9]. A possible approach to reduce the degraded performance of the Student module consists in applying regularisation [9], which would regulate the network optimisation to reduce the negative transfer.

In practice, it would be hard for an GMM model to achieve an optimal network architecture following the lifelong learning procedure explained above. In the following, we analyze the Student's forgetting when considering that the Teacher continually modifies its network architecture, by deriving a new risk bound.

Lemma 2: Let $h_s \in \mathcal{H}$ be a Student trained on the knowledge learnt by the Teacher module (GMM) that has an arbitrary number of components during training (usually by training fewer components than the number of tasks t). The risk bound for h_s when learning the (t)th task is :

$$\sum_{i=1}^{t} \Big\{ \mathcal{L}_{S_i}\left(h_s, S_i\right) \Big\} \leq \sum_{i=1}^{\mathrm{card}(B)} \Big(\mathcal{L}_{\tilde{S}_{b_i}^{(1)}}\big(h_s, f_{\tilde{S}_{b_i}^{(1)}}\big) + f'\big(S_{b_i}, \tilde{S}_{b_i}^{(1)}\big) \\ + \mathcal{L}_d\big(S_{b_i,\mathcal{X}}, \tilde{S}_{b_i,\mathcal{X}}^{(1)}\big) \Big) + \sum_{i=1}^{\mathrm{card}(B')} \Big\{ \mathcal{L}_{\tilde{S}_{b_i'}^{(t-\hat{b}_i+1)}}\big(h_s, f_{\tilde{S}_{b_i'}^{(t-\hat{b}_i+1)}}\big) \\ + \sum_{k=-1}^{\hat{b}_i} \Big(\mathcal{L}_d(\tilde{S}^k_{b_i',\mathcal{X}}, \tilde{S}^{(k+1)}_{b_i',\mathcal{X}}) + f'(\tilde{S}^k_{b_i'}, \tilde{S}^{(k+1)}_{b_i'}) \Big) \Big\} \\ + \mathcal{L}_{\tilde{S}_t^{(0)}}\big(h_s, f_{\tilde{S}_t^{(0)}}\big) + f'\big(S_t, \tilde{S}_t^{(0)}\big) + \mathcal{L}_d\big(S_{t,\mathcal{X}}, \tilde{S}_{t,\mathcal{X}}^{(0)}\big) \,, \tag{14}$$

where B' does not involve the (t)-th task because the (t)-th task is only trained once. Let $\mathrm{R}_s^{\mathrm{mixture}}$ represent the right hand side of Eq. (14). According to Lemma 3 from the paper, we have $\mathrm{R}_s^{\mathrm{mixture}} \geq \mathrm{R}^{\mathrm{mixture}}$. We can easily prove Lemma 2 since the Student module h_s is trained on samples drawn from the $\tilde{S}^2_{b_i}$ even if b_ith task is trained just once by one of the GMM' components.

We only consider implementing the Student, within the Teacher-Student architecture, for unsupervised learning. We observe that the Student module under unsupervised learning performs worse than the Teacher module in GMM. This is justified by Eq. (13) and (14), although these are mainly considered for supervised learning.

Appendix J
The results on the class-incremental learning

Despite the fact that GMM is used for task-incremental learning, we also investigate the performance of GMM in the class-incremental setting. Following the setting from [10], we create a new dataset, namely Permuted MNIST, where MNIST is divided into ten tasks, and each task would process images containing a pixel permutation of the images from MNIST. Split MNIST [11] splits MNIST into five tasks, where each task contains samples belonging to two successive classes of digits. For Permuted MNIST, the classifier in each expert is implemented by using a Multilayer Perceptron (MLP) of 2 hidden layers, with one hundred units each. The classifier of each expert is a neural network comprised of 2 layers with 256 units on each layer when training on Split MNIST. We vary the value of the threshold λ from Eq.(17) of the paper, between 80 to 120 for Permuted MNIST and between 30 and 60 for Split MNIST. We also consider a more challenging dataset, Split CIFAR [11], in continual learning. For Split CIFAR, we use the same procedure from [10], in which CIFAR10 is considered as the initial task to be learnt and then we learn five tasks where we select training samples from 10 categories in their order from CIFAR100 as a task. Following from [10], we adopt a network architecture that consists of four convolution layers and two fully connected layers. We can observe that the shared classifier is implemented by four convolution layers. During the expansion process, we build a sub-classifier by using two fully connected layers, on the top layer of the shared classifier. Therefore, each sub-classifier reuses parameters from this shared classifier, which can reduce the whole model size. We only update the parameters of the shared classifier when learning the first task in order to relieve forgetting. For this experiment we vary the threshold λ in the proposed GMM from 80 to 100. We perform five independent runs and calculate the average classification accuracy for Permuted MNIST, Split MNIST and Split CIFAR, which are shown in Table IV and Table III, where all other results, except for GMM, are cited from [10]. We denote by '6 C' that GMM trains six components. These results demonstrate that we can achieve optimal performance by ensuring that GMM has a number of components equal to that of learnt tasks, and it does not suffer from forgetting, as discussed in Lemma 2 of the paper.

References

[1] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," in *Proc. Conf. on Learning Theory (COLT), arXiv preprint arXiv:2002.06715*, 2009.

TABLE III
Results for Split CIFAR where 'C' denotes the number of components for the proposed GMM model.

Methods	Split CIFAR
FROMP-L_2	$75.6\% \pm 0.4$
FROMP	$76.2\% \pm 0.4$
SI	$73.5\% \pm 0.5$
VCL + random coreset	$67.4\% \pm 1.4$
EWC	$71.6\% \pm 0.9$
GMM	$76.40\% \pm 0.3$ (6 C)
GMM	65.70% (5 C

TABLE IV
Results for the continuous learning benchmark where 'C' denotes the number of components for the proposed GMM model.

Methods	Permuted MNIST	Split MNIST
Improved VCL* [12]	$93.1\% \pm 1$	$98.4\% \pm 0.4$
EWC* [13]	84%	63.1%
DLP* [14]	82%	61.2%
SI* [11]	86%	98.9%
FROMP* [10]	$94.9\% \pm 0.1$	$99.0\% \pm 0.1$
FRCL-TR* [15]	$94.3\% \pm 0.2$	$97.8\% \pm 0.7$
FRCL-RND* [15]	$94.2\% \pm 0.1$	$97.1\% \pm 0.7$
GMM	**96.46**$\%\pm0.03$ (10 C)	**99.21**$\%\pm0.04$ (5 C)
GMM	88.78% (7 C)	96.77% (4 C)
GMM	95.25% (8 C)	91.37% (3 C)

[2] R. Š. Radim Tyleček, "Spatial pattern templates for recognition of objects with regular structure," in *Proc. German Conf. on Pattern Recognition, vol. LNCS 8142*, 2013, pp. 364–374.

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

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

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

[6] 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.

[7] 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 (NIPS)*, 2018, pp. 5962–5972.

[8] M. Zhai, L. Chen, F. Tung, J. He, M. Nawhal, and G. Mori, "Lifelong GAN: Continual learning for conditional image generation," in *Proc. of IEEE Int. Conf. on Computer Vision (ICCV)*, 2019, pp. 2759–2768.

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

[10] P. Pan, S. Swaroop, A. Immer, R. Eschenhagen, R. Turner, and M. E. E. Khan, "Continual deep learning by functional regularisation of memorable past," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 4453–4464.

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

[12] S. Swaroop, C. V. Nguyen, T. D. Bui, and R. E. Turner, "Improving and understanding variational continual learning," in *Proc. NIPS-workshops Continual Learning, arXiv preprint arXiv:1905.02099*, 2018.

[13] 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)*, vol. 114, no. 13, pp. 3521–3526, 2017.

[14] A. J. Smola, S. Vishwanathan, and E. Eskin, "Laplace propagation," in *Advances in Neural Inf. Proc. Systems (NIPS)*, 2004, pp. 441–448.

[15] M. K. Titsias, J. Schwarz, A. G. d. G. Matthews, R. Pascanu, and Y. W. Teh, "Functional regularisation for continual learning with Gaussian processes," in *Proc. Int. Conf. on Learning Represenations (ICLR), arXiv preprint arXiv:1901.11356*, 2019.