Multiband VAE: Latent Space Partitioning for Knowledge Consolidation in Continual Learning

Kamil Deja\textsuperscript{1}, Paweł Wawrzyński\textsuperscript{1}, Daniel Marczak\textsuperscript{1}, Wojciech Masarczyk\textsuperscript{1}, Tomasz Trzciński\textsuperscript{1,2}  
\textsuperscript{1}Warsaw University of Technology \textsuperscript{2}Tooploox  
kamil.deja.dokt@pw.edu.pl

Abstract

We propose a new method for unsupervised continual knowledge consolidation in generative models that relies on the partitioning of Variational Autoencoder’s latent space. Acquiring knowledge about new data samples without forgetting previous ones is a critical problem of continual learning. Currently proposed methods achieve this goal by extending the existing model while constraining its behavior not to degrade on the past data, which does not exploit the full potential of relations within the entire training dataset. In this work, we identify this limitation and posit the goal of continual learning as a knowledge accumulation task. We solve it by continuously re-aligning latent space partitions that we call bands which are representations of samples seen in different tasks, driven by the similarity of the information they contain. In addition, we introduce a simple yet effective method for controlled forgetting of past data that improves the quality of reconstructions encoded in latent bands and a latent space disentanglement technique that improves knowledge consolidation. On top of the standard continual learning evaluation benchmarks, we evaluate our method on a new knowledge consolidation scenario and show that the proposed approach outperforms state-of-the-art by up to twofold across all testing scenarios.\footnote{Code is available at:https://github.com/KamilDeja/multiband_vae}

Figure 1: Overview of our Multiband VAE. With each new task, we first learn a local copy of our model to encode new data examples. Then we consolidate those with our current global decoder - main generative model which is able to generate examples from previous tasks.

1. Introduction

Recent advances in generative models \cite{2019arXiv191007341C, 2019arXiv191005225G} led to their unprecedented proliferation across many real-life applications \cite{2019arXiv191000252C}, including image synthesis and manipulation \cite{2019arXiv191000252C, 2019arXiv191007341C}, medical data processing \cite{2019arXiv191000252C, 2019arXiv191000252C} or synthetic data simulation \cite{2019arXiv191000252C}. One of the main advantages of deep generative models is their ability to encode complex datasets into simpler distributions. Unfortunately, this goal remains difficult to deliver on, in real-life situations where training data is presented to the model in separate portions. In such cases neural networks including generative models suffer from \textit{catastrophic forgetting} of previously acquired knowledge when trained on something new \cite{2019arXiv191007341C}. One of the potential reasons for this behavior is the ever-changing distribution of data in the consecutive tasks fed to the model, which challenges the network to identify the relation between the incoming information and what it already learned in the previous iterations.

\textit{Continual learning} methods that aim to address these challenges focus on supervised training where the problem of forgetting is usually addressed in one of three ways, through regularization \cite{2019arXiv191007341C, 2019arXiv191000252C}, adjustment of the structure of a network to the next task \cite{2019arXiv191007341C, 2019arXiv191000252C, 2019arXiv191000252C, 2019arXiv191000252C, 2019arXiv191000252C}, or rehearsal of previously seen samples when training the network with new data \cite{2019arXiv191007341C, 2019arXiv191000252C}. Since storing past data in the memory requires a growing buffer, recently proposed methods use generative models to regenerate previously seen samples from learned probability distributions \cite{2019arXiv191007341C, 2019arXiv191000252C, 2019arXiv191000252C}.

In this work, we postulate to look at generative models not only as a source of past data generations sampled from a latent space but also as the universal mechanisms for accumulating and consolidating knowledge acquired by continually learned models. Therefore, we investigate how data samples from different distribution modes, \textit{e.g.} classes or labels, are represented in the generative model’s latent
space, depending on their spread across tasks. To that end, we prepare a set of diversified continual learning scenarios with data splits following Dirichlet distribution, inspired by a similar approach proposed for federated learning [16].

These considerations motivate the inception of Multiband VAE – a new method for continual knowledge consolidation in a generative model. The core idea behind our method is to split the process of model retraining into two steps: (1) a local encoding of data from the new task into a new partition (band) of the latent space, and (2) a global rearrangement and consolidation of new and previous bands that leverages similarities between data samples.

The main contribution of this work is a new method for unsupervised continual learning of generative models that aims at efficient consolidation of data distributions stored in the model’s latent space. The novelty of the resulting Multiband VAE lies in the rearrangement of the latent partitions that correspond to separate tasks based on the similarity between the data samples encoded within those partitions.

To support knowledge consolidation between different bands, we additionally propose a controlled forgetting mechanism that enables the substitution of degraded reconstructions of past samples with new data from the current task. For a more precise alignment of bands, we also introduce a simple disentanglement technique with additional binary latent space. The proposed solution leads to a significant improvement in the quality of the samples generated by our continually learned generative model, irrespectively of the distribution of classes between training tasks. As a result, our Multiband VAE significantly outperforms competing methods across various continual learning evaluation scenarios, reaching up to a twofold improvement.

2. Related works

Most of the works incorporating generative models in the continual learning scenarios relate to the idea of generative rehearsal. In this technique, the base model is trained with a mixture of new data examples from the current task and recreation of previous samples generated by a generative model. This idea was first introduced in [40], with Generative Adversarial Networks (GAN) [11]. However, since any structure used to encode past data may suffer from catastrophic forgetting, authors propose to always retrain a generative model with a combination of current examples and generations from previous tasks in so-called Generative Replay. In [45] authors extend this idea by combining Variational Autoencoder (VAE) [19] with classification model. It was further explored also in other works with an application to the open datasets [31], or combinations with buffer [3]. In [26], authors overview how different generative models trained with the Generative Replay behave in the continual learning scenarios. Our Multiband VAE is a direct extension to this technique.

Continual learning of generative models In [32], authors adapt regularization based methods such as Elastic Weight Consolidation (EWC) [20], Laplace Propagation (LP) [41] and Synaptic Intelligence (SI) [56]. The shared idea behind those techniques is to regularize the weights adjustments during model retraining in a way that it minimizes forgetting on previous tasks.

In HyperCL [47], authors introduce an independent yet general approach where the weights of continually trained networks are generated by yet another model called a hypernetwork. This approach yields state-of-the-art results in terms of discriminative models training for task-incremental scenario and is applicable also to the generative models.

In [32], authors introduce a Variational Continual Learning (VCL) method for continual learning in VAE with a separate part of the model for each task. This approach provides a significant improvement over the standard replay technique, since it directly separates examples from different tasks, while at the same time allowing knowledge sharing in the jointly optimized parts. In order to differentiate tasks, in Continual Unsupervised Representation Learning (CURL) [35] authors propose a model that learns task-specific representation on top of a larger set of shared parameters, and deals with task ambiguity by performing task inference within the generative model. This approach directly addresses the problem of forgetting, by maintaining a buffer for original instances of poorly-approximated samples and expanding the model with a new component whenever the buffer is filled. Similar idea also applied to the classification problem was further analyzed in CN-DPM [25], where authors propose resources extension with a set of neural network experts.

There are several works that train GANs in the continual learning scenarios either with memory replay [49] or regularization [4], with the extension to VAEGAN [22] in [53].

Continual learning with disentanglement In VASE [1], authors propose a method for continual learning of shared disentangled data representation from examples in different tasks and datasets. While encoding images with a standard VAE, VASE also seeks shared generative factors. In our work, we use a simplified disentanglement method with additional binary latent space. A similar concept of mixed-type latent space was introduced in LifelongVAE [34], where latent is composed of discrete and continuous values. Additionally, authors combine this idea with a teacher and student networks working in the generative rehearsal procedure with additional regularization and latent space expansion.

In this work, we propose a continual learning method in which we first train the separate generative model with only currently available data in order to combine it with the global model. Conceptually similar ideas were proposed for
supervised tasks. In Progress & Compress [39] authors use distillation techniques to consolidate previous knowledge, while in [57] authors employ additional unlabeled data.

3. Method

In this section, we first introduce a general intuition behind Multiband VAE followed by a detailed description of the method. Although our multiband training applies to various generative models, here we focus on the Variational Autoencoder.

Our model consists of three components: an encoder which maps original images into the first latent space $L$, a translator that translates samples from $L$ to the second latent space $Z$, and a decoder that generates images from embeddings in $Z$. The latent space $L$ consists of regularized embeddings of examples from separate tasks. Those vectors combined with appropriate task codes are then translated into common latent space $Z$ in which examples are aligned independently of their source task. In training, we use auxiliary local copies of our models, while the final solution consists of the global translator and decoder.

The general idea of Multiband VAE is to disentangle two parts of continual learning training: Acquisition of new knowledge and Consolidation of new and previous knowledge. To achieve this goal, we divide the training of our generative model into three phases:

In the first one called local training, we train a local copy of our generative model using only data available in the current task. We do it without any constraints regarding past knowledge. For the first task, this is the only part of the training, after which we store the decoder and translator of our local model as the global ones. For the following tasks, we keep the local encoder which is used to encode new data examples. Fig. 2 depicts this phase.

In the second phase, we learn the relation between newly accumulated data and previous examples. To that end, we use a separate model called translator that maps encoded images accompanied by their task codes into the so-called bands in the latent space $Z$. The role of the translator is to find the overlapping parts of data distributions from different tasks and place them accordingly in the latent space $Z$. If a new task contains only examples from an unseen domain, they should be placed in the separate area of $Z$.

Therefore, we retrain the global translator with a frozen global decoder, to encode new data into the area of latent space $Z$ that yields the lowest reconstruction error. The overview of this part of the training is presented in Fig. 3.

In the third phase of the training, we consolidate knowledge accumulated within previous tasks with new knowledge encoded by the local copy of our model. To that end, we train the global translator and global decoder to reproduce the same data examples as previously, while at the same time learning to encode original images from encodings provided by the local encoder. To simplify new data incorporation and improve previous generations we propose a method for controlled forgetting of previous generations whenever a new example is encoded into similar embeddings in latent space $Z$. The overview of knowledge consolidation is presented in figure Fig. 4.

3.1. Formal description

Let $X_1 \ldots X_i$ be datasets available in the following tasks. After each $i$-th task we have two main parts of our model:

- **Global translator** $t(L, c(j); \theta_l^j)$, where $\theta_l^j$ are model’s weights, $L \sim \mathcal{P}(\cdot)$ is a noise sampled from distribution $\mathcal{P}$, and $c$ is a coding function (e.g. one-hot) for the index $j$ ($j \leq i$) of already processed task. Global translator translates noise vectors and task codes into values $Z$ in the latent space $Z$.

- **Global decoder** $d(Z; \omega_g^j)$ where $\omega_g^j$ are the model’s weights, and $Z$ are embeddings in the latent space $Z$. Global decoder translates $Z$ into generated samples $X'$.

When training our model with new data, we use additional copy of VAE consisting of three auxiliary models:

- **Local encoder** $e(X; v_l^j)$ with weights $v_l^j$ which encodes original inputs $X \in X_i$ into examples $\mu$ that define locations $L \sim \mathcal{P}_L(\cdot|\mu)$ in the latent space $L$, as in VAE.

- **Local translator** $t(L, c(i); \theta_t^i)$ with weights $\theta_t^i$, inputs $L$, $c(i)$ and outputs $Z$ in the latent space $Z$.

- **Local decoder** $d(Z; \omega_l^i)$ with weights $\omega_l^i$, inputs $Z$ and outputs $X'$, which are the reconstructions of original images $X \in X_i$. 

![Figure 2: In the local training procedure we train local encoder, local translator and local decoder models only with samples from currently available data. After local training we use local encoder to encode data from new task.](image)

![Figure 3: We start the translator training with a frozen global decoder in order to find similar data examples shared between previous and current tasks in the latent space $Z$.](image)
3.2. Knowledge acquisition - local training

In this step, we train a copy of our model to adjust to the data available in the current task. For \( i \)-th task where \( i > 1 \), we start by creating our auxiliary local model that consists of three parts: encoder \( e(\cdot; \nu_i) \), translator \( t(\cdot; \theta_i^t) \) and decoder \( d(\cdot; \omega_i) \).

For decoder and translator we initialize the weights with the copy of those from the global versions \( \omega_g \) and \( \theta_g \), while for local encoder, \( \nu_i \) with weights of the previous local encoder \( \nu_i' \). We train the local model using only examples from the current task in the standard VAE manner. We first sample the task index \( j \) from \( \mathcal{X}_i \). Then, we encode them with local encoder \( e(\cdot; \nu_i') \) to obtain encodings \( \mu \). On the basis of those encodings, we sample \( L \) from distribution \( P_L(\cdot|\mu) \). We pass sampled encodings with recent task codes \( c(i) \) through the local translator to obtain embeddings \( Z = t(L, c(i); \theta_i^t) \). Finally we propagate \( Z \) through the local decoder which generates reconstructions \( X' = d(Z; \omega_i') \). We train local VAE (local encoder, translator, and decoder) with a sum of reconstruction loss and regularization on distribution \( P_L \).

\[
\text{Loss} = ||X - X'||^2 + \gamma r(\mathcal{P}_L(\cdot|\mu), \mathcal{P}(\cdot))
\]

where \( r \) is a regularization function on distribution \( \mathcal{P}_L(\cdot|\mu) \), such as KL divergence in case of \( \mathcal{P} \) equal to normal distribution, and \( \gamma > 0 \) is a coefficient that scales this regularization in \( \text{Loss} \) as in \( \beta \)-VAE [15].

3.3. Shared knowledge discovery - data bands arrangement

After local training, we start the knowledge consolidation by finding the best fit for the encodings of currently available data in the latent space \( Z \). At the beginning of the new task, we create a copy of the current global decoder’s and translator’s weights: \( \omega' \leftarrow \omega_g', \theta' \leftarrow \theta_g \). These are needed to generate previous data generations for the examples which we want to remember. Additionally, we use the locally trained encoder \( e(\cdot; \nu_i') \) to calculate encodings of currently available data. In order to find the best fit for the new band of data, we first train the global translator to translate encodings of examples from the current task into the area of the latent space \( Z \) that yields the lowest reconstruction error for the current version of the global decoder. Therefore, we freeze the weights of the global decoder \( \omega_g' \) and use it only to propagate gradient to the translator. We prepare input-output training pairs for the translator as follows. We first sample the task index \( j \sim \{1, \ldots, i\} \):

If \( j = i \), we sample a training examples from currently available data \( X \subset \mathcal{X}_i \) and pair them with the encoded noise by sampling from the distribution \( P_L(\cdot|\mu) \) conditioned on the encodings provided by the local encoder \( L \sim P_L(\cdot|e(x; \nu_i')) \).

Otherwise, for \( j < i \), we create target values \( X \) from previous tasks, by sampling random noise \( L \sim \mathcal{P}(\cdot) \), and propagating it through the copy of global models \( X' = d(t(L, c(j); \theta'; \omega_g') \).

The final pair of inputs-outputs is therefore defined as:

\[
((L, c(j)), (X))
\]

With combination of such pairs we translate input data \((L, c(j))\) into points \( Z \) which propagated through the frozen global decoder results in reconstruction \( X' = d(Z, \omega_g') \). We adjust the weights of the translator \( \theta_g' \) to generate \( Z \) which minimizes reconstruction error between \( X \) and \( X' \).

3.4. Knowledge consolidation - global training

With a new band of encodings arranged we finally train both of our global models to reconstruct currently available data, while retaining previously learned examples. Therefore, we create a combination of training examples in the same way as in Eq. (2), but this time we adjust both the weights of the global translator \( \theta_g' \) and decoder \( \omega_g' \).
We train the global decoder to reproduce the same outputs as its previous—frozen version, while at the same time learning how to generate new data examples from the encodings provided by the local encoder. Therefore the loss for our global training is a reconstruction error between generations \(X'\) and targets \(X\).

### 3.5. Controlled forgetting

In the process of shared knowledge discovery, we arrange the encodings \(Z\) of different data examples according to the similarity of their reconstructions from the same global decoder, regardless of their original task. Then, in the knowledge consolidation part, we train the global translator and decoder on the basis of new original images encodings and previous data reconstructions. However, when generating previous data examples, the probability of sampling from the exact embeddings of original previous images approaches zero. Therefore, previous data generations are in fact interpolations between several original data samples. On the basis of this observation, we propose a simple controlled forgetting mechanism.

Whenever we prepare a batch of training pairs \(\langle (L, c(j)), (X) \rangle\), we first calculate encodings \(Z = t(L, c(j), \theta_g)\) for all of the samples. Then, we check whether some examples from the previous tasks \((j < i)\) have similar encodings as samples from the current task \((j = i)\). If this is true, we overwrite the target generations \(X\) from the previous task, with corresponding original images from the train-set. Therefore, we allow our global model to forget the generation of certain previous examples, whenever we have an original image that can replace it.

We use cosine distance as a metric for encodings similarity, while the threshold for data replacement is one of the method’s hyperparameters. In our experiments with various data domains, we observed that for the similarity exceeding 0.9 forgetting occurs only for similar previous reconstructions with no overwriting of different examples.

### 3.6. Two latents Variational Autoencoder

In the global part of our training, we rely on the regularization of VAE’s latent space. In practice, when encoding various examples from different classes into the same latent space of standard VAE, we can observe that some latent variables are used to distinguish encoded class, and therefore they do not follow desired continuous distribution [43, 29]. The extended experimental analysis of this phenomenon can be found in the supplementary material.

In this work, we propose a simple disentanglement method with an additional binary latent space that addresses this problem. To that end, we train our encoder to encode input data characteristics into a set of continuous variables \(\mu_c\) and binary variables \(\mu_b\), which are used to sample vectors \(L_c\) and \(L_b\) that together form \(L\) – the input to the translator model. For the continuous variables, we follow the reparameterization trick introduced in [19]. To sample vector \(L_c\), we train our encoder to generate two vectors: means \(\mu_m\) and standard deviations \(\mu_s\). Those vectors are used as parameters of Normal distribution from which we sample \(L_c \sim \mathcal{N}(\mu_m, \text{diag}(\mu_s^2))\). For binary variables, we introduce a similar procedure based on the Gumbel soft-max [17] approximation of sampling from Bernoulli distribution. Therefore, we train our encoder to produce probabilities \(\mu_p\) with which we sample binary vectors \(L_b \sim B(\mu_p)\). To allow generations of new data examples, for continuous values, we regularize our encoder to generate vectors \(L_c\) from the standard normal distribution \(\mathcal{N}(0, I)\) with a Kullback-Leibler divergence. For binary vectors \(L_b\), during inference, we approximate probabilities \(\mu_p'\) with the average of probabilities \(\mu_p\) for all of the examples in the train-set. We calculate \(\mu_p'\) during the last epoch of the local training. Therefore, to generate new data examples we sample random continuous variables \(L_c \sim \mathcal{N}(0, I)\) and binary variables \(L_b \sim B(\mu_p')\) and propagate them through the global translator and decoder.

### 4. Experiments

To visualize the difference between Generative Replay and Multiband VAE, in Fig. 5 we present a toy-example with the MNIST dataset limited to 3 tasks with data examples from 3 classes. At the beginning of the second task (on the left) both methods start from a similar latent space where representations of examples from different tasks are mixed together. Given two data distributions with separate classes, we can see that after 5 epochs of training our Multiband VAE was able to separate them in the latent space \(Z\). On the contrary, the same model trained with standard Generative Rehearsal started to transform some examples from the previous task into currently available samples.

After 40 epochs of training, we see the outcome of this behavior, in which for generative replay a lot of examples from the previous task, was affected by the new data. In Multiband VAE we can observe that separate bands of data retained their characteristics. Nevertheless, when presented with a third task with similar data distribution (partially the same class), our model is able to properly adjust bands so that truly similar examples are encoded in the same area of the latent space, while different examples regardless of their source task are separated.

### 4.1. Evaluation setup

In order to compare with other methods, we propose a simple Variational Autoencoder architecture similar to the one in [32], with 9 fully connected layers. However, our Multiband VAE is not restricted to any particular architecture, therefore we also include experiments with a convolutional version. The exact architecture and training hy-
perparameters are enlisted in the appendix and code repository. In our experiments, we do not condition our generative model with class identity since it greatly simplifies the problem of knowledge consolidation, and is applicable to all of the evaluated methods. However, we condition our generative model on task number. For that purpose, we use binary encoding with co-prime numbers as proposed in [6].

4.2. Evaluation

To assess the quality of our method we conduct a series of experiments on benchmarks commonly used in continual learning (MNIST [24], Omniglot [21]) and generative modeling (FashionMNIST [51]). Since the performance of VAE on diverse datasets like CIFAR-10/100 is limited, in order to evaluate how our method scales with more complex data we include tests on CelebA [27].

For each dataset, we prepare a set of training scenarios that are designed to evaluate various aspects of continual learning challenges. To simulate a situation in which different tasks come with separate data distributions, we run class incremental scenarios [46] for all benchmarks. In these tasks, we evaluate whether the model suffers from catastrophic forgetting. To test different methods in more challenging setups we also include the Omniglot dataset which we split up to 20 separate tasks and combined dataset MNIST $\rightarrow$ FashionMNIST in which new examples come from an entirely different distribution.

While catastrophic forgetting is an important topic, the class incremental scenario simplifies the problem of learning data distribution in the generative model’s latent space, since the identity of the task precisely conditions final generations. Therefore in this work we also employ more complex data splits introduced for the evaluation of federated learning methods [16]. Following [16], we split examples from the same classes into tasks, according to the probability $q \sim \text{Dir}(\alpha p)$ sampled from the Dirichlet distribution, where $p$ is a prior class distribution over all classes and $\alpha$ is a concentration parameter that controls similarity of the tasks. Visualizations of different splits according to the parameter $\alpha$ are presented in Figure 6. In this work, we evaluate models with challenging splits according to Dirichlet $\alpha = 1$ distribution, in which the model has to learn the differences between tasks, while also discovering that part of the available data is already embedded in the latent space. Therefore, ideally, we would expect backward knowledge transfer – improvement in the quality of generated examples also from past tasks when retrained with more examples from the same class.

To measure the quality of different methods we use the Fréchet Inception Distance (FID) [14], which assesses the quality of generations. FID compares the distribution of generated samples and original data propagated to the penultimate layer of the Inception [42] model pre-trained on the ImageNet dataset [7]. As proposed in [2], for simpler datasets such as MNIST and Omniglot, we calculate FID on the basis of the LeNet [23] based classifier pre-trained on the whole target dataset. Additionally, for a better under-

![Figure 6: Class splits for different continual learning scenarios. In class incremental split each task consists of separate classes. For $\alpha = 1$ Dirichlet distribution, we have highly imbalanced splits with randomly occurring dominance of one or two classes. For higher values of parameter $\alpha$, classes are split almost equally.](image)
standing of the strengths and drawbacks of different methods, we report the precision and recall of the distributions as proposed in [38]. As authors indicate, those metrics disentangle standard FID score into two aspects: the quality of generated results (Precision) and their diversity (Recall).

For each experiment, we report the FID, Precision, and Recall averaged over the final scores for each task. For methods that do not explicitly condition generations on the task index (CuRL and LifelongVAE), we calculate measures in comparison to the whole test-set. The results of our experiments are presented in Tab. 1 and Tab. 2, where we show scores averaged over 3 separate runs. Because of the change in evaluation models (Inception and LeNet) results between different datasets are not comparable.

As presented in Tab. 1, our model outperforms comparable methods in terms of quality of generated samples for class incremental scenarios as well as more complex splits with Dirichlet distribution. In Tab. 2 we present results with more challenging scenarios. Results of comparison on the Omniglot dataset with 20 splits indicate that for almost all of related methods training from the complex data splits according to the Dirichlet $\alpha = 1$ distribution poses a greater challenge than the class incremental scenario. However, our Multiband VAE is able to precisely consolidate knowledge from such complex setups, which leads to similar results as in the class incremental scenario. Only CURL is able to generate data with similar quality thanks to additional model expansion. To assess how different models retain knowledge on past examples when presented new data examples from drastically different distribution we also present the results of experiments on joined datasets. In a FashionMNIST $\rightarrow$ MNIST benchmark, we train a model to generate images from FashionMNIST, followed by standard MNIST, and for the MNIST $\rightarrow$ FashionMNIST dataset the other way round. With each task, we increase the number of classes in a class incremental way. As presented in Tab. 2 all of the experiments indicate the superiority of Multiband VAE over similar approaches in various scenarios.

| Measure | FI D↓  | Prec↑  | Rec↑  | F I D↓  | Prec↑  | Rec↑  | F I D↓  | Prec↑  | Rec↑  |
|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| SI [56] |       |        |        |        |        |        |        |        |        |
| EWC [20] | 129 77 80 | 153 75 76 | 134 28 24 | 140 21 19 |        |        |        |        |        |
| Generative replay [40] | 120 79 87 | 254 70 65 | 96 43 58 | 133 35 43 |        |        |        |        |        |
| VCL [32] | 68 85 94 | 127 78 80 | 104 30 32 | 138 21 20 |        |        |        |        |        |
| HyperCL [47] | 62 91 87 | 148 78 75 | 108 46 33 | 155 35 21 |        |        |        |        |        |
| CURL* [35] | 107 95 77 | 181 84 74 | 86 47 64 | 83 46 56 |        |        |        |        |        |
| Lifelong-VAE* [34] | 173 75 72 | 224 63 73 | 131 33 62 | 201 9 49 |        |        |        |        |        |
| Multiband VAE | 24 94 97 | 41 92 96 | 61 66 69 | 82 62 65 |        |        |        |        |        |
| Multiband VAE (conv) | 23 92 98 | 30 92 97 | 56 65 72 | 77 58 69 |        |        |        |        |        |

Table 1: Average Fréchet Inception Distance (FID) and distribution Precision (Prec) and Recall (Rec) after the final task in different data incremental scenarios. Our method with vanilla architecture outperforms competing solution on all of the standard benchmarks. Additional results improvement can be observed with convolutional architecture (conv).∗For methods that do not condition generations on task, we present the scores in comparison to the whole test-set.

| Measure | F I D↓  | Prec↑  | Rec↑  | F I D↓  | Prec↑  | Rec↑  | F I D↓  | Prec↑  | Rec↑  | F I D↓  | Prec↑  | Rec↑  |
|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| SI [56] |       |        |        |        |        |        |        |        |        |        |        |        |
| EWC [20] | 48 87 81 | 115 64 28 | 140 18 16 | 146 18 15 |        |        |        |        |        |        |        |        |
| Generative replay [40] | 46 88 81 | 106 68 31 | 106 74 38 | 119 72 30 |        |        |        |        |        |        |        |        |
| VCL [32] | 45 88 82 | 74 72 62 | 92 75 53 | 99 36 45 |        |        |        |        |        |        |        |        |
| HyperCL [47] | 48 87 82 | 122 62 21 | 127 71 25 | 81 45 51 |        |        |        |        |        |        |        |        |
| CURL* [35] | 54 86 76 | 98 86 45 | 115 84 38 | 128 31 28 |        |        |        |        |        |        |        |        |
| Lifelong-VAE* [34] | 22 95 95 | 31 96 92 | 26 94 92 | 98 69 42 |        |        |        |        |        |        |        |        |
| Multiband VAE | 21 97 93 | 33 95 86 | 41 95 83 | 51 65 70 |        |        |        |        |        |        |        |        |
| Multiband VAE (conv) | 12 98 96 | 24 95 91 | 24 96 91 | 49 68 70 |        |        |        |        |        |        |        |        |

Table 2: Average Fréchet Inception Distance (FID) and distribution Precision (Prec) and Recall (Rec) after the final task in different data incremental scenarios. In a diverse Omniglot dataset our approach matches the state-of-the-art performance of CURL, without model expansion, while in a more challenging combined dataset Multiband VAE clearly outperforms different data incremental scenarios. Our method with vanilla architecture outperforms competing solution on all of the experiments indicate the superiority of Multiband VAE clearly outperforms competing solutions. We can observe additional quality improvement with convolutional architecture (conv).∗For methods that do not condition generations on task, we present the scores in comparison to the whole test-set.
Table 3: Average FID, distribution Precision (Prec), and Recall (Rec) after the final task in different data incremental scenarios on the CelebA dataset. Our Multiband VAE consolidates knowledge from separate tasks even in the class incremental scenario, therefore clearly outperforming solution with separate VAE per task. With more evenly tasks splits our method converges to the upper bound solution which is a model trained with full data availability.

| Measure          | FID↓ | Prec↑ | Rec↑ | FID↓ | Prec↑ | Rec↑ | FID↓ | Prec↑ | Rec↑ |
|------------------|------|-------|------|------|-------|------|------|-------|------|
| Separate models  | 103  | 31    | 21   | 105  | 24.5  | 7.6  | 109  | 28.4  | 10.6 |
| Multiband VAE (conv) | 95   | 28.5  | 23.2 | 93   | 33    | 22   | 89   | 36.2  | 28   |

Table 4: Ablation study on the MNIST dataset with Dirichlet α = 1 distribution. Average FID after the last task.

| Modification          | FID↓ |
|-----------------------|------|
| Generative replay     | 254  |
| + Multiband training  | 53   |
| + Binary latent space | 44   |
| + Controlled forgetting| 41   |
| + Convolutional model | 30   |

4.3. Ablation study

The main contribution of this work is a multiband training procedure, yet we also introduce several mechanisms that improve knowledge consolidation in a continually trained model. Tab. 4 shows how those components contribute to the final score of our model. Our multiband training is the most important part of proposed solution, while additional techniques slightly support knowledge consolidation. We present results in the Dirichlet α = 1 scenario where input from controlled forgetting may be observed.

5. Conclusion

In this work, we propose a new method for unsupervised continual learning of generative models. This area of research is not yet widely explored, even though representation learning is naturally entangled in the structure of generative models. In our Multiband VAE we benefit from this observation and propose a method that aims at knowledge consolidation in Variational Autoencoder’s latent space. To that end, we disentangle two parts of the learning process Acquisition of new knowledge and Consolidation of new and past information. This allows us to directly control the process of integration of a new portion of data encodings with the previous ones. In order to evaluate this process, we propose new continual learning scenarios based on the Dirichlet distribution. Our experiments on various benchmarks, and in various standard and new training scenarios show the superiority of Multiband VAE over other related methods.
References

[1] Alessandro Achille, Tom Eccles, Loic Matthey, Christopher P Burgess, Nick Watters, Alexander Lerchner, and Irina Higgins. Life-long disentangled representation learning with cross-domain latent homologies. arXiv preprint arXiv:1808.06508, 2018. 2

[2] Mikolaj Binkowski, Dougal J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. arXiv preprint arXiv:1801.01401, 2018. 2

[3] Lucas Caccia, Eugene Belilovsky, Massimo Caccia, and Joelle Pineau. Online Learned Continual Compression with Adaptive Quantization Modules. In ICML, 2020. 2

[4] Yifan Chang, Wenbo Li, Jian Peng, Haifeng Li, Yu Kang, and Yingliang Huang. Memory protection generative adversarial network (mpgan): A framework to overcome the forgetting of gans using parameter regularization methods. IEEE Access, 8:179942–179954, 2020. 2

[5] Brian Cheung, Alexander Terekhov, Yubei Chen, Pulkit Agrawal, and Bruno Olshausen. Superposition of many models into one. In NeurIPS, 2019. 1

[6] Kamil Deja, Paweł Warzyżski, Daniel Marczak, Wojciech Masarczyk, and Tomasz Trzcinski. Binplay: A binary latent autoencoder for generative replay continual learning. arXiv preprint arXiv:2011.14960, 2020. 6

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009. 6

[8] Robert M. French. Catastrophic forgetting in connectionist networks. Trends in cog. sci., 1999. 1

[9] Siavash Golkar, Michael Kagan, and Kyunghyun Cho. Continual Learning via Neural Pruning. In NeurIPS, 2019. 1

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. In NeurIPS, 2014. 1

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. In NeurIPS, 2014. 2

[12] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. A review on generative adversarial networks: Algorithms, theory, and applications. arXiv preprint arXiv:2001.06937, 2020. 1

[13] Anvita Gupta and James Zou. Feedback gan (fbgan) for dna: a novel feedback-loop architecture for optimizing protein functions. arXiv preprint arXiv:1804.01694, 2018. 1

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Advances in neural information processing systems, pages 6626–6637, 2017. 6

[15] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016. 4

[16] Tzu-Ming Harry Hsu, Hang Qi, and Matthew Brown. Measuring the effects of non-identical data distribution for federated visual classification. arXiv preprint arXiv:1909.06335, 2019. 2, 6

[17] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144, 2016. 5

[18] Nathan Killoran, Leo J Lee, Andrew Delong, David Venuaund, and Brendan J Frey. Generating and designing dna with deep generative models. arXiv preprint arXiv:1712.06418, 2017. 1

[19] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In ICLR, 2014. 1, 2, 5

[20] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences, 114(13):3521–3526, 2017. 1, 2, 7

[21] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. Science, 350(6266):1332–1338, 2015. 6

[22] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In International conference on machine learning, pages 1558–1566. PMLR, 2016. 2

[23] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4):541–551, 1989. 6

[24] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist, 2010. 6

[25] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. In International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE, 2019. 2

[26] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In Proceedings of International Conference on Computer Vision (ICCV), December 2015. 6

[27] David Lopez-Paz and Marc’Aurelio Ranzato. Gradient Episodic Memory for Continual Learning. In NeurIPS, 2017. 1

[28] Emile Mathieu, Tom Rainforth, N Siddharth, and Yee Whye Teh. Disentangling disentanglement in variational autoencoders. In International Conference on Machine Learning, pages 4402–4412. PMLR, 2019. 5

[29] Martin Mundt, Yong Won Hong, Iuliia Pluschch, and Visvanathan Ramesh. A Wholistic View of Continual Learning with Deep Neural Networks: Forgotten Lessons and
the Bridge to Active and Open World Learning, 2020. arXiv:2009.01797. 1

[31] Martin Mundt, Sagnik Majumder, Iulia Pluichch, Yong Won Hong, and Viswanathan Ramesh. Unified Probabilistic Deep Continual Learning through Generative Replay and Open Set Recognition, 2020. arXiv:1905.12019v4. 2

[32] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. arXiv preprint arXiv:1710.10628, 2017. 2, 5, 7

[33] Michela Paganini, Luke de Oliveira, and Benjamin Nachman. Calogan: Simulating 3d high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks. Physical Review D, 97(1):014021, 2018. 1

[34] Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Lifelong generative modeling. Neurocomputing, 404:381–400, 2020. 2, 7

[35] Dushyant Rao, Francesco Visin, Andrei A. Rusu, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Continual unsupervised representation learning. 2019. 2, 7

[36] S. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. iCaRL: Incremental Classifier and Representation Learning. In CVPR, 2017. 1

[37] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive Neural Networks, 2016. arXiv:1606.04671. 1

[38] Mehdi SM Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. arXiv preprint arXiv:1806.00035, 2018. 7

[39] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In International Conference on Machine Learning, pages 4528–4537. PMLR, 2018. 3

[40] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual Learning with Deep Generative Replay. In NeurIPS, 2017. 2, 7

[41] Alex J Smola, SVN Vishwanathan, and Eleazar Eskin. Laplace propagation. In NIPS, pages 441–448, 2004. 2

[42] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1–9, 2015. 6

[43] Jakub Tomczak and Max Welling. Vae with a vampprior. In International Conference on Artificial Intelligence and Statistics, pages 1214–1223. PMLR, 2018. 5

[44] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1415–1424, 2017. 1

[45] Gido M van de Ven and Andreas S Tolias. Generative replay with feedback connections as a general strategy for continual learning, 2018. arXiv:1809.10635. 1, 2

[46] Gido M Van de Ven and Andreas S Tolias. Three scenarios for continual learning. arXiv preprint arXiv:1904.07734, 2019. 6

[47] Johannes von Oswald, Christian Henning, João Sacramento, and Benjamin F Grewe. Continual learning with hypernetworks. arXiv preprint arXiv:1906.00695, 2019. 2, 7

[48] Yeming Wen, Dustin Tran, and Jimmy Ba. BatchEnsemble: an Alternative Approach to Efficient Ensemble and Lifelong Learning. In ICLR, 2020. 1

[49] Chenshen Wu, LuíS Herranz, Xialei Liu, Yaxing Wang, Joost van de Weijer, and Bogdan Raducanu. Memory replay gans: Learning to generate new categories without forgetting. In NeurIPS, 2018. 2

[50] Ye Xiang, Ying Fu, Pan Ji, and Hua Huang. Incremental Learning Using Conditional Adversarial Networks. In ICCV, 2019. 1

[51] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms, 2017. arXiv:1708.07747. 6

[52] Ju Xu and Zhanxing Zhu. Reinforced Continual Learning. In NeurIPS, 2018. 1

[53] Fei Ye and Adrian G Bors. Learning latent representations across multiple data domains using lifelong vaegan. In European Conference on Computer Vision, pages 777–795. Springer, 2020. 2

[54] Ran Yi, Yong-Jin Liu, Yu-Kun Lai, and Paul L Rosin. Apdrawinggan: Generating artistic portrait drawings from face photos with hierarchical gans. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10743–10752, 2019. 1

[55] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong Learning with Dynamically Expandable Networks. In ICLR, 2018. 1

[56] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In International Conference on Machine Learning, pages 3987–3995. PMLR, 2017. 1, 2, 7

[57] Juntong Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 1131–1140, 2020. 3