REPLAY-BASED CONTINUAL LEARNING WITH CONSTANT TIME COMPLEXITY

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose a new constant-time approach to replay-based continual learning (CL). GR is a strategy that protects existing knowledge from previous and inaccessible training sessions (sub-tasks) by having auxilliary generator networks produce samples from those sub-tasks and merging them with the current sub-task. The main innovation we propose is twofold: first of all, we propose **selective modification** of existing knowledge only where it overlaps with the current sub-task. On the other hand, we propose **selective replay** of samples, to protect existing knowlege in opverlapping areas only.

Since only the overlapping parts of previous knowledge need to be protected against CF, fewer samples can be replayed. We use this to maintain a constant ratio between real and generated samples, irrespectively of the number of processed sub-tasks. Consequently, training time is kept constant for all sub-tasks. We test the proposed strategy on CL problems derived from visual classification and reinforcement learning problems, and present a comparison to VAE-based generative replay.

1 INTRODUCTION

This contribution is in the context of continual learning (CL), a recent flavor of machine learning that investigates learning from data with a non-stationary distribution. A common effect in this context is catastrophic forgetting (CS), an effect where previously acquired knowledge is abruptly lost after a change in the data distribution. In the default scenario for CL (see, e.g., **?**), a number of assumptions are made in order to render CL more tractable: first of all, distribution changes are assumed to be abrupt, partitioning the data stream into *sub-tasks* of stationary distribution. Then, sub-task onsets are supposed to be known, instead of inferring them from data. And lastly, sub-tasks are assumed to be disjoint, i.e., not containing the same classes in supervised scenarios. Please refer to fig. 1 for a visualization of the default scenario. Together with this goes the constraint that no, or only a few, samples may be stored. A very promising line of work on CL in the default scenario are rehearsal methods. Rehearsal aims at preventing CF by using samples from previous sub-tasks to augment the current one. On the one hand, there are "true" rehearsal methods which use a small number of

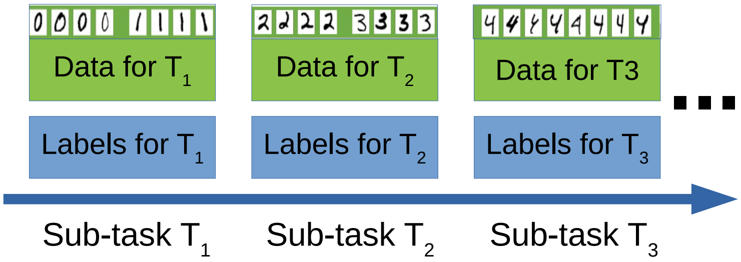

Figure 1: The default scenario for (supervised) CL. The data stream is assumed to be partitioned into *sub-tasks* T_i. Statistics within a sub-task are considered stationary, and sub-task data and labels (targets) are assumed to be disjoint, i.e., from different classes (MNIST classes used here for visualization). Sub-task onsets are assumed to be known as well.

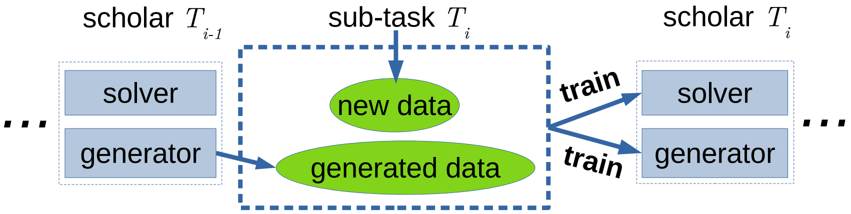

Figure 2: In generative replay, a *scholar* is trained at every sub-task. Scholars are composed of generator and solver modules. The solver performs the task, e.g., classification, whereas the generator serves as a memory for samples from previous sub-tasks $T_{i'}$, $i' < i$. Please note that the amount of generated data usually far exceeds the amount of new data.

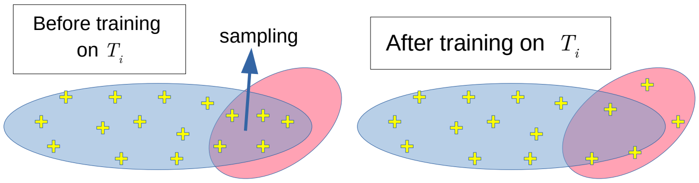

Figure 3: The essence of the proposed ReCT-CL approach, represented in data space. At every sub-task T_i, data with new statistics (red area) is presented to a scholar which has acquired knowledge of previous sub-tasks (blue area), which is encoded by GMM *components* (yellow crosses), see text for details. The diagrams represent the situation before (left) and after (right) training on sub-task T_i. Training data for T_i is created by a union of samples from T_i and samples generated from the components in the overlapping area. After training, components in this area have shifted to represent the new statistics as well.

stored samples for augmentation. On the other hand, there are *pseudo-rehearsal* or *replay* methods, where the samples to augment the current sub-task are produced in unlimited number by a generator, which removes the need to store samples. A schematics of the training process in generative replay is given in fig. 2. As promising as replay seems to be as a principled approach to CL, it presents several challenges: First of all, if DNNs are employed as solvers and generators, then all classes must be represented in equal proportion in the training data at every sub-task, since DNNs are sensible to class frequencies. For the simple case of a single new class of D samples per sub-task, the generator must produce $(i-1)D$ samples at sub-task i in order to always train with D samples per class. By this linear dependency, even very small additions to a large body of existing knowledge (a common use-case) require a large amount of samples for replay, see, e.g., **?**. Since replay is always a lossy process, this imposes a severe limit on GR-based CL performance. Lastly,

1.1 Approach

We propose a replay-based approach ((visualized in fig. 4) relying on fully probabilistic models of machine learning, namely Gaussian Mixture Models (GMMs) and corresponding convolutional extensions introduced in **??**. A key concept is an explicit similarity (probability) measure defined by centroids (prototypes) and covariance matrices, which can be used to compare incoming data to existing knowledge, represented by GMM *components*. As shown in **?**, only the GMM components that are similar to incoming data are adapted. Conversely, knowledge represented by dissimilar components is unaffected and thus protected against CF. Incoming data are augmented by sampling from protoypes in overlapping areas, i.e., prototypes with highest similarity, see fig. 3, which ensures that the prototypes are adapted but not overwritten.

1.2 Contributions

The salient points of ReCT-CL are:

Selective replay: existing knowledge is not replayed indiscriminately, but only where significant overlap with new data exists. The detection of overlaps is an intrinsic property of the employed GMMs.

Selective replacement: The scholar is not re-initialized between sub-tasks, and existing knowledge is only modified where an overlap with new data exists. Selective replacement is an intrinsic property of GMMs.

Constant time complexity: Since only a small part of existing knowledge needs to be considered for replay at each sub-task, the number of generated/replayed samples can be small as well. The ratio between new and generated data can therefore be fixed, irrespectectively of past sub-tasks.

1.3 Related work

2 Methods

2.1 Data

For the evaluation we use the following image datasets:
MNIST ? consists of 60 000 28×28 gray scale images of handwritten digits (0-9).
FashionMNIST ? consists of images of clothes in 10 categories and is structured like MNIST.
CRL ? contains 30000 mono images from an RL task performed by a simulated robot. Each of the 15×100) images is formed by concatenating line camera images from the last 3 RL time steps.

These datasets are not particularly challenging w.r.t. non-continual classification, although CL tasks formed from these datasets according to the default scenario (**??**) have been shown to be very difficult, see, e.g., **?**. Particular examples of CL tasks formed from MNIST and FashionMNIST that have been used in the literature are $D5-5$, $D9-1$, $D2^5$ or $D1^{10}$, where each number indicates the amount of classes contained in each sub-task.

2.2 ReCT-CL using GMMs and DCGMMs

In contrast to conventional replay, where a scholar is composed of a generator and a solver network (see **??**), ReCT-CL proposes a scholar where single network performs both functions. This network is composed of a Gaussian Mixture Model (GMM) layer followed by a linear regression readout layer. All layers are trained by SGD according to **?**. For improving classification and generation performance, we have also used deep convolutional GMMs (DCGMMs) as described in **??**, which can contain a succession of convolutional GMM, half-convolution and pooling layers. The mathematical treatment of ReCT-CL scholars will assume a single GMM layer only without loss of generality.
GMMs describe data distributions by a set of K *components*, consisting of component weights π_k, centroids $\boldsymbol{\mu}_k$ and covariance matrices $\boldsymbol{\Sigma}_k$. A data sample $\boldsymbol{x}$ is assigned a probability $p(\boldsymbol{x}) = \sum_k \pi_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ as a weighted sum of normal distributions $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. Training of GMMs is performed as detailed in **?** by adapting centroids, covariance matrices and component weights through the SGD-based minimization of the log-likelihood $\mathcal{L} = \sum_n \log \sum_k \pi_k \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$.

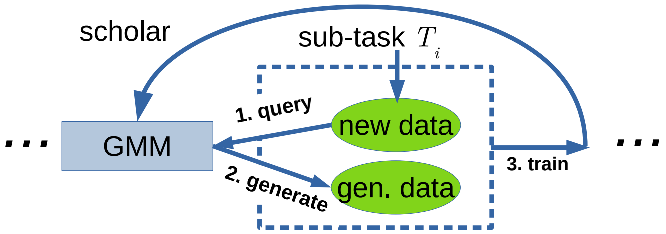

Figure 4: The proposed ReCT-CL approach. The main difference to conventional generative replay is that new data are used to *query* the current scholar for similar samples. The ratio of generated and new data can thus be chosen constant as explained in the text. The scholar is not reinitialized but re-trained from its current state with generated and new data.

The GMMs emplayed in ReCT-CL have two basic functions: classification and variant generation. For classification, we first compute the GMM *responsibilities* $\gamma_k(\boldsymbol{x}_n) = \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\boldsymbol{x}_n; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$ and feed them into a linear regression layer as $\boldsymbol{o}(\boldsymbol{x}_n) = \boldsymbol{W}\boldsymbol{\gamma}(\boldsymbol{x}_n) + \boldsymbol{b}$. Variant generation is a special case of GMM sampling: given a data sample $\boldsymbol{x}_n$, we first determine the closest component: $i = \text{argmax}_k \gamma_k(\boldsymbol{x}_n)$ and then draw a sample from the normal distribution defined by that component: $\hat{\boldsymbol{x}} \sim \mathcal{N}(\cdot; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$.

To ensure CL capacity, ReCT-CL training relies on three key concepts: loss masking, MSE classification, and max-component approximation for GMMs. Loss masking is a technique that assigns different learning rates ϵ_n to individual samples $\vec{x}_n$, depending on whether they are classified correctly ($\chi_n = 1$) or not ($\chi_n = 0$): $\epsilon_n = \chi_n \epsilon_\uparrow + (1 - \chi_n)\epsilon_\downarrow$. In order to make the classification layer suitable for CL, we employ an MSE loss instead of cross-entropy for target values $\boldsymbol{t}_n$ and model outputs $vo(\boldsymbol{x}_n)$: $\mathcal{L}_{\text{MSE}} = \sum_n ||\boldsymbol{o}(\boldsymbol{x}_n) - \boldsymbol{t}_n||^2$. Lastly, we use the annealing procedure and max-component approximation to the GMM log-likelihood described in **??** for training the GMM layer. This ensures that only best-matching (most similar) component is adapted (plus nearby components, at least during early training stages), whereas dissimilar components are unaffected. The neighbourhood of affected components around the best-matching one is defined by the annealing radius $r(t)$ which is automatically reduced over time, see **?** for details.

Figure 4 shows the high-level logic of ReCT-CL. We assume that unknown data from a new sub-task arrives in a batch. Each sample of that batch is used to *query* the GMM for a similar, known sample using variant generation. Mixing new and generated samples in a defined, constant proportion creates the training data for the current sub-task. This is the driving idea behind ReCT-CL: a new sample causes adaptation in its best-matching component. However, variants generated by that sample will almost always cause adaptation to the same component. The component will therefore converge to a compromise between new data and existing knowledge, while dissimilar components stay completely unaffected.

2.3 VAE-based generative replay

3 Experiments

3.1 Intrinsic CL capacity of GMMs

3.2 Variant generation with GMMs and DCGMMs

3.3 Model comparison at variable time complexity

3.4 Model comparison at constant time complexity

3.5 Generalization: ReCT-CL using DCGMMs

4 Discussion and conclusion

Author Contributions

AK created the library used for ReCT-CL experiments, based on earlier code by AG. AG wrote a significant part of the manuscript and designed the experiments. AK conducted he GMM-based experiments, AG the VAE-based ones. Both contributed to the evaluation and the description of the experiments in the manuscript, as well as to proof-reading.

A Appendix

You may include other additional sections here.