Closed-Loop Memory GAN for Continual Learning

Amanda Rios* and **Laurent Itti**
University of Southern California, Los Angeles, USA
{amandari, itti}@usc.edu

Abstract

Sequential learning of tasks using gradient descent leads to an unremitting decline in the accuracy of tasks for which training data is no longer available, termed catastrophic forgetting. Generative models have been explored as a means to approximate the distribution of old tasks and bypass storage of real data. Here we propose a cumulative closed-loop memory replay GAN (CloGAN) provided with external regularization by a small memory unit selected for maximum sample diversity. We evaluate incremental class learning using a notoriously hard paradigm, "single-headed learning," in which each task is a disjoint subset of classes in the overall dataset, and performance is evaluated on all previous classes. First, we show that when constructing a dynamic memory unit to preserve sample heterogeneity, model performance asymptotically approaches training on the full dataset. We then show that using a stochastic generator to continuously output fresh new images during training increases performance significantly further meanwhile generating quality images. We compare our approach to several baselines including fine-tuning by gradient descent (FGD), Elastic Weight Consolidation (EWC), Deep Generative Replay (DGR) and Memory Replay GAN(MeRGAN). Our method has very low long-term memory cost, the memory unit, as well as negligible intermediate memory storage.

1 Introduction

Since early development and throughout life humans are constantly faced with unknowns in the environment which demand a persistent adaptation and expansion of past knowledge. In addition, as knowledge is expanded, learning is often facilitated since objects and tasks are often closely related and interconnected. For instance, during development, infants learn to categorize animals according to dimensions such as size, texture, shape, sound, among others. However, subsequent addition of new species rarely corrupts classification performance on the already learned categories. In fact, learning broad-species domains can aid in finer species discriminatory capability [Flesch *et al.*, 2018].

Nonetheless, recreating human-like lifelong continual learning remains a central challenge in Artificial Intelligence. State of the art deep neural networks (DNN) trained to perform supervised continual learning are known to undergo a phenomenon termed "catastrophic forgetting", which describes a sharp decline in the performance of the model on previously learned tasks as soon as a new task is introduced [French, 1999; McCloskey and Cohen, 1989]. This behavior does not come as a surprise if one recalls that in DNNs, learning an input output mapping implies parameterizing the network with an optimal weight set, through loss minimization. Thus, if training data is unavailable for previous tasks, there will be no more loss term for the old data and a weight parametrization may blatantly deviate from the previous optimal state incurring severe memory erasure.

2 Prior Work

In the recent literature, several methods have been proposed aiming to ameliorate catastrophic forgetting. They can be roughly subdivided into 3 groups: regularization, network-growing and replay approaches. With regularization methods, one constrains the change of learnable parameters to prevent "overwriting" what was previously encoded. For instance, [Li and Hoiem, 2017] perform distillation between multiple realizations of a network at distinct time-points, ensuring that the new weights do not shift significantly from the old. In a similar vein, [Kirkpatrick *et al.*, 2017] operate within a single network model and use a Fisher information matrix computed with saved samples drawn from past tasks, which then acts as a regularizer preserving highly correlated weights. Similarly, [Zenke *et al.*, 2017] use path integrals of loss-derivatives to constrain weights crucial to past tasks, yielding an intermediate parameterization with minimal combined loss.

Alternatively, in region-growing algorithms, the architecture itself is altered to accommodate new tasks followed by retraining. For instance, [Fernando *et al.*, 2017] freeze the most important paths in the network, therefore forcefully preventing forgetting, and incrementally add new network chunks to incorporate new tasks. Lastly, In replay methods, the models no longer preserve a key pathway or weights. In these algorithms, one estimates the distribution of the old data

*Contact Author

either by saving a small fraction of the original dataset into a memory buffer or by training a generator to mimic the lost data and labels. At each new task, these methods learn by presenting a network with both new images as well as replay of estimated or buffered old images, reverting the continual framework into a multi-task setting and thus alleviating forgetting [Shin *et al.*, 2017]. Other works have built on the idea of using a buffer of real data to approximate the past distribution [Rebuffi *et al.*, 2017; Lopez-paz and Ranzato, 2017; Nguyen *et al.*, 2018].

Yet, despite a growing number of appealing solutions, catastrophic forgetting is not a solved issue. Regularization methods have been shown to perform poorly in single-headed incremental class learning, for instance [Kemker and Kanan, 2018; Parisi *et al.*, 2019], and here we reproduce this limitation in our own results for elastic weight consolidation [Kirkpatrick *et al.*, 2017]. On the other hand, region growing approaches, while usually providing a clean solution for constrained incremental problems, can quickly become memory expensive since they require both an architectural expansion and the storage of at least a portion of old data for retraining.

Likewise, replay methods also run into scalability issues. So far, generative replay models learn a data distribution by resorting to intermediate copy states of the generator. In Deep Generative Replay (DGR) an unconditional GAN is trained at each task to cumulatively generate and discriminate images. Since the proposed GAN is unconditional, they employ an additional classifier (Solver) which is trained in parallel to classify the generated images and assign corresponding labels [Shin *et al.*, 2017]. During each task switch, DGR makes a copy of the generator and classifier networks and uses them to generate sample images and labels for the old tasks. In Memory Replay GAN (MeRGAN) with joint replay, [Wu *et al.*, 2018] propose a modification in the DGR framework by substituting the unconditional GAN for an ACGAN, thereby eliminating the need for the additional solver. Copy operations are both expensive and often lead to image quality being degraded through consecutive tasks. Moreover, replicating network states successively is not a fully desirable solution since, from the biological perspective, a human brain cannot produce an "intermediate copy" of itself to transfer knowledge. Lastly, methods which rely rather on small subsets of past data, memory buffers, have shown to yield good results but they do not make explicit how much of the performance is due to the algorithm developed and how much is intrinsically due to the variability included in the buffer unit.

3 Closed Loop memory GAN

3.1 Model Overview

In this paper, we propose a hybrid approach between memory buffers and deep generative models aiming to specifically reduce memory costs and maximize both the classification performance and generated image quality throughout training. In our model, there is only one generator and embedded classifier trained cumulatively, with no intermediate copy step. In this framework, as a new task is learned, the old data is approximated by continuously sampling from the generator at its present state, forming a closed loop training paradigm.

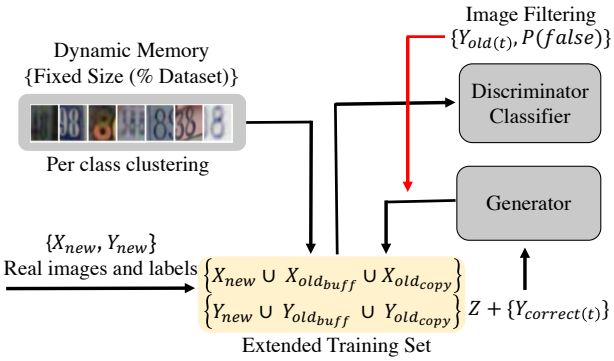

Figure 1: CloGAN Model used for cumulative and continual learning. Past data is sampled from the generator and filtered by the embedded classifier. Old data is a combination of the fresh stochastic generator output and a small memory buffer used to "smoothen" the old data distribution for quality output.

Of course, since a new task also modifies the parameterization of the generator, this procedure cannot be applied without some verification that the generated images are reasonable approximations of the old distribution that has been lost. Our method tackles this issue by, first, using an image filtering step in which either the classifier or the discriminator is used to assess the sample image quality and, as a result, blocking bad images from entering the training loop. Second, we employ external regularization by constructing a small dynamic memory buffer with real data samples chosen to maximize image heterogeneity and to enforce smoothness in the representation of old classes. The image buffer has fixed memory allotment. Therefore, it is not allowed to grow which requires eliminating some old images to make room for new ones. The sampling for the old data is then always a combination of buffer samples and "on-the-fly" generated samples, which provide a stochastic up-sampling of the memory unit.

3.2 Model Architecture

A vanilla GAN consists of two networks, a Generator and a Discriminator, competing with each other in a zero-sum game framework. The core block of our model (CloGAN), see figure (1), is a modified GAN termed Auxiliary Conditional Generative Adversarial (AC-GAN) [Odena *et al.*, 2017]. The AC-GAN is also composed of 2 networks, but it includes a classifier combined in the same architecture as the discriminator, via an expansion to K+1 output nodes, for K classes plus the original vanilla Real/Fake discriminator output.

In an AC-GAN framework the generator is fed a uniform noise $z \sim p_z$ appended with a corresponding class label $c \sim p_c$. Thus, the conditional generator, described by θ_G, generates an image $x = G_{\theta_G}(z, c)$ and the AC-GAN learns a mapping in which the noise z is independent of the class c, enabling multiple class outputs for a fixed noise input. While the generator θ_G is trained to generate images as closely resembling the input image distribution, the discriminator, θ_D, is conversely trained to discriminate these generated images as fake, loss L_{FT}. The embedded classifier, θ_C, shares most weights with the discriminator and generates a label prediction which, if incorrect, contributes to the overall loss of both generator and

discriminator, L_C. Overall, an AC-GAN is easier to train than a conventional vanilla GAN while also producing higher quality images. The loss functions are given as follows in (1) and (4) for generator and discriminator/classifier respectively.

$$\theta_G^* = \min_{\theta_G}(L_{FT}^G(\theta,X) + L_C^G(\theta,X)) \quad (1)$$

$$L_{FT}^G(\theta,X) = -\mathbb{E}_{z\sim p_z, c\sim p_c}[D_{\theta_D}(G_{\theta_G}(z,c))] \quad (2)$$

$$L_C^G(\theta,X) = -\mathbb{E}_{z\sim p_z, c\sim p_c}[y_c log(C_{\theta_C}(G_{\theta_G}(z,c)))] \quad (3)$$

$$\theta_D^*, \theta_C^* = \min_{\theta_D,\theta C}(L_{FT}^D(\theta,X) + L_C^D(\theta,X)) \quad (4)$$

$$L_{FT}^D(\theta,X) = \mathbb{E}_{z\sim p_z, c\sim p_c}[D_{\theta_D}(G_{\theta_G}(z,c))] - \quad (5)$$

$$\mathbb{E}_{(x,c)\sim X}[D_{\theta_D}(x)] \quad (6)$$

$$L_C^D(\theta,X) = -\mathbb{E}_{(x,c)\sim X}[C_{\theta_C}(G_{\theta_G}(z,c))] \quad (7)$$

Note that a plausible alternative to using a GAN would be to use a variational auto encoder (VAE) instead [Kingma *et al.*,]. However, in our testing, we have not been able to achieve results with a VAE as good as those presented here using a GAN. Hence, in the following, we restrict our analysis to approaches based on GAN. Details of the implementation can be found in the supplementary materials link.

3.3 Closed-Loop Training with Replay

In the continual learning setting, our method approximates the likelihood of old data by employing CloGAN to continuously output fresh new images at each mini-batch during training. A combination of image filtering and external regularization by an image memory buffer confer stability to the closed-loop procedure. At each task, our model is trained using an extended dataset which includes real images for the new task, GAN replayed images for old tasks, and memory images, forming an extended training set S_t (8), see figure 2. The memory component can be given a weighted importance, λ_{mem}. The network is then trained by minimizing (9,10).

$$S_t = S_t^{real} \cup S_{t-1}^{GAN} \cup \lambda_{mem} S_{t-1}^{memory} \quad (8)$$

$$\min_{\theta_D^t, \theta_C^t}(L_R^D(\theta,S_t) + L_C^D(\theta,S_t)) \quad (9)$$

$$\min_{\theta_G^t}(L_R^G(\theta,S_t) + L_C^G(\theta,S_t)) \quad (10)$$

3.4 Image Filtering

At each mini-batch, the generator outputs fresh images approximating samples from old tasks, with the intent of producing a stochastic up-sampling of the reduced memory core. However, since these images are then used as training data in a closed loop, they have to be of the best quality possible to minimize error propagation. Thus, at each generation step, images are assessed for their quality and "filtered" out if they do not correspond to the standard.

Here, we use the embedded classifier in CloGAN to generate a prediction for the conditional image. If this prediction does not match the conditioning label, the image is filtered out. When old images are generated for closed-loop replay, they are sampled from a model which has already previously converged for generation and classification of old tasks. The rationale behind this evaluation is that images which are miss-classified have a higher probability of being distorted because of the ongoing training of the new task, and of deviating too grossly from the original distribution. We term this method Class-Conditioned Filtering (CFM).

In addition to CFM, we implemented a more complex procedure, "Discriminator Rejection Sampling" (DRS) proposed in [Azadi *et al.*, 2018]. The latter employs the discriminator of an AC-GAN to approximately correct errors in the GAN generated distribution. Details of the implementation can be found in the supplementary materials link. We compare both to a baseline case for DRS which rejects a sample if the output from its discriminator logit layer has a score below some threshold, Soft rejection Filtering (SRF) [MacKay, 2003]. Overall, we found that CFM, DRS and SFR perform equivalently well. A table with comparisons is included in the supplementary materials link. Hence, since CFM has a much faster running time, we opted for carrying out only class conditional filtering in our final model.

Algorithm 1: CloGAN Train

Input : Data $S_t^{real}, ..., S_T^{real}$;
Require: T: Number of Tasks;
I_t : Number of iterations;
B : Buffer Size;
K_c : Number of clusters per class;
λ_{mem} : Memory importance;

```
θ*_G, θ*_{D,C}, θ*_C ← Train AC-GAN(S_{t=1}) for i = 1 to I_1
S^{memory}_{t=1} ← BufferConstruct (K, B, S^{real}_t)
for t ← 2 to T do
    S*_t = S^{real}_t ∪ λ_mem S^{memory}_{t-1}
    for i ← 1 to I_t do
        S^{i*}_t ← Batch(S*_t)
        S^{GAN}_{t-1} ← Forward (G(z, y^c_{t-1}))
        S^{GAN}_{t-1} ← Filter (S^{GAN}_{t-1})
        S^i_t ← S^{i*}_t ∪ S^{GAN}_{t-1}
        θ*_G, θ*_{D,C}, θ*_C ← Train AC-GAN(S^i_t)
    end
    S^{memory}_t ← BufferConstruct (K, B, S^{memory}_{t-1}, S^{real}_t)
end
```

Figure 2: Training Algorithm. Procedure $Train$ is described in section 3.3; $Filter$ in 3.4; $BufferConstruct$ in 3.5

3.5 Dynamic Memory Buffer

We fill a small memory buffer with samples and labels of original past data to perform external regularization. The memory can be seen as a stable reference frame throughout training that enforces a "smoothness" in the representation for each class. At each task, a selection method is employed to choose the samples from the new task which will go into the buffer, with the aim to maximize sample heterogeneity. Also, since a buffer has fixed size, this selection method is further used to determine which of the old task samples will be removed to make space for the incoming new data,

employing again the heuristics of sample heterogeneity. Several buffer selection strategies were initially experimented but the best selection scheme was K-means clustering per class, both at image insertion and removal. In more details, the construction scheme is as follows: at the end of each current task, a k-centers algorithm is run per each class in the current tasks's training labels, super-labeling each image as one of K clusters. At the time of insertion into the memory buffer, we select equal numbers of image samples from each class-specific cluster. Additionally, if the buffer is full we compute the space needed for new images and remove an equivalent number of old images. We do this by assessing their stored super-cluster labels and removing equal amounts of samples per cluster, thereby preserving heterogeneity. By storing the per-class, cluster assignment superlabels we also avoid repeating the clustering operation.

3.6 Continual Learning Baselines

We evaluate other continual learning algorithms as baseline comparisons. We implement Elastic Weight Consolidation (EWC; Kirkpatrick [2017]), Deep Generative Replay (DGR; Shin [2017]) and Memory Replay GAN (MeRGAN; Wu [2018]) . With DGR, to make our implementation a fair comparison, we use an unconditional GAN with the same architecture and complexity as our CloGAN, except that it has only one Real/Fake output node. For both EWC and DGR, we use a classifier with identical architecture as our embedded classifier/discriminator, but with one fewer output node since a pure classifier does not evaluate Real/Fake attribution. Finally, for MeRGAN we implement an AC-GAN with identical architecture as our CloGAN.

4 Experiments

4.1 Buffer Selection

We experimented with several buffer selection schemes but they under-performed class-specific K-centers. In the other selection methods, we extracted the logit or softmax layer of the discriminator/classifier network and computed measures such as Kurtosis and Peak-Difference to assess sample heterogeneity. The latter measure corresponds to the difference between softmax scores of the most probable and second most probable class for a given image. As such, we ranked the images according to each measure and kept the images with a probability proportional to their score. In other words, we performed a roulette weighting procedure such as in genetic selection [Goldberg and Deb, 1991]. Table 1 contains performance metrics for 3 buffer selection schemes and no selection (none) during CloGAN incremental class learning using the FASHION dataset with memory buffer of size 0.16%.

Method	CloGAN
Class-Kcenter	75.87 +/- 0.43
Kurtosis	64.52 +/- 0.73
Peak Difference	57.74 +/- 0.61
None	71.03 +/- 1.4

Table 1: % Correct As A Function Of Buffer Selection

4.2 Incremental Learning

We evaluate continual learning as accumulating knowledge of a growing number of disjoint classes, termed incremental learning. Furthermore, we make use of a challenging variation of incremental learning, "single headed learning". Here, each task is a disjoint subset of classes from the overall dataset. Performance is evaluated for all previous classes, resulting in a 1/K chance level, where K is the number of classes accumulated to that point. We evaluate incremental class learning in 4 datasets: MNIST [LeCun *et al.*,], FASHION [Xiao *et al.*, 2017], SVHN [Netzer *et al.*, 2011] and E-MNIST [Cohen *et al.*, 2017]. The first 3 were subdivided in disjoint subsets of 2 classes per task, with a total of 5 tasks to cover all the label types. E-MNIST, a larger dataset, was divided into tasks of 3 classes, covering 24 different classes in 8 consecutive tasks. To account for the growing number of classes, we create extra output nodes which are incrementally used, which allows us a single head for all tasks.

We distinguish our procedure from Multi-Headed learning [Farquhar and Gal, 2018] in which prediction is constrained to classes in each task. For instance, a multi-headed version of our MNIST test would use and re-use only two output nodes. After training on full disjoint MNIST with 5 tasks of 2 classes each, when evaluating the first task (digits 0 and 1), a multi-headed would only have to decide between digit 0 vs 1, as opposed to a one in ten decision for single-headed. This typically leads to much higher accuracies partially because an output node never becomes completely disabled, as it is always used for the last task. Finally note that a multi-headed network with only 2 output nodes provides an output that needs to be further disambiguated by knowing the task.

Average Continual Performance

Figure 3 displays the average performance of CloGAN when varying memory buffer size. Our method avoids catastrophic forgetting even with very small buffer sizes such as 0.08% (50 images) and 0.16% (100 images), for both MNIST and FASHION. For the more challenging E-MNIST and SVHN, buffer requirement becomes more demanding. Nonetheless, we obtain superior performance over the competing methods with still very reduced memory sizes: only 0.5% (576 images) and 1% (492 images).

Table 2 compares maximum average accuracies after training all tasks, for all methods tested. First, when no memory or GAN sampling is performed, the "FGD" condition which contains only fine-tuning with gradient descent, catastrophic forgetting occurs. Second, EWC accuracy rapidly declines, asymptotically reaching the catastrophic forgetting curve. EWC has already been shown to behave poorly in incremental single-headed paradigms [Kemker and Kanan, 2018; Farquhar and Gal, 2018]. To further confirm that this degradation of performance was not particular to our implementation, we replicated the permuted-MNIST experiment proposed in the original EWC paper ; and verified that in this learning paradigm EWC performs very well. This discrepancy between the experiments is likely due to the difference in output mapping, see supplementary.

Lastly, we report the accuracies for the deep replay methods, DGR and MeRGAN. For MNIST, both DGR and MeR-

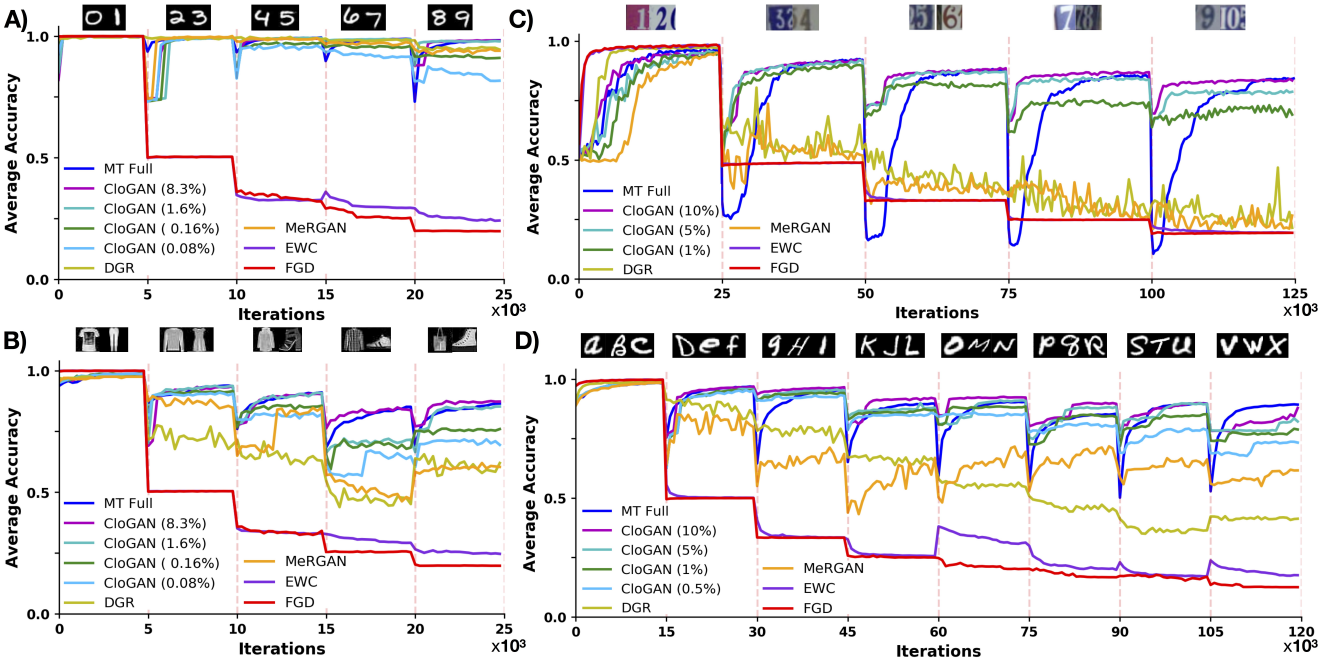

Figure 3: Maximum Average Accuracies across continually learned tasks. A) MNIST; B) Fashion; C) SVHN; D) E-MNIST. The dashed lines indicate start of a new task represented by a disjoint set of classes. We illustrate the performance CloGAN as memory sizes are varied (memory allotment is in parenthesis). In red we show catastrophic forgetting when fine-tuning by gradient descent. In salmon we show multi-task training until convergence with the full datasets starting from scratch at every task switch (MT-Full). We also show results for EWC, DGR and MeRGAN.

Method	Mnist	Fashion	Svhn	Emnist
MT-full	98.29	86.48	84.43	89.41
CloGAN	98.03 (1.6%)	85.25 (1.6%)	79.30 (5%)	83.50 (5%)
CloGAN	92.26 (0.16%)	76.15 (0.16%)	73.08 (1%)	79.14 (1%)
MeRGAN	98.25	65.62	31.94	61.92
DGR	94.90	62.11	46.83	42.35
EWC	29.19	26.52	22.55	23.76
FGD	19.97	20.22	19.56	14.28

Table 2: Performance after continual learning of all tasks. For CloGAN, memory allotment is in parenthesis

GAN perform very well, reaching 94.9 % and 98.25% whereas CloGAN achieves accuracies of 92.26% with memory of 0.16% and 98.03 with (1.6%). However, for all other datasets, which are significantly harder than MNIST, DGR and MeRGAN both underperform CloGAN by significant amounts. For SVHN, the most challenging, both DGR and MeRGAN display degraded performance after the first task. This behavior likely has cause in a persistent degradation of generated image quality throughout training. Both methods represent old data exclusively by replayed images from an intermediate generator copy. If the generator cannot produce images which represent the original distribution with high fidelity, the gap in representation capacity can be enlarged and propagated through successive GAN transfer (copy) operations. CloGAN alleviates GAN representation degeneration because it is trained from an extended set containing both replay images from the generator and real images in the buffer. The real images never degenerate and act as an anchor to keep smoothness and quality in the subsequent generated images. DGR has another disadvantage over CloGAN: it does not generate conditioned images, requiring a separate classifier to produce old image labels during training. If that classifier does not have perfect performance, it will inevitably misslabel some images, contributing to error propagation. In the supplementary we tested a new copy-CloGAN that copies the generator at each task switch. However, the copy operation did not provide an advantage if comparing memory usage.

Stochastic Up-Sampling

We confirm that CloGAN performs an upsampling of the memory buffer selection by comparing our method to two variations in which the AC-GAN is trained only from a memory buffer, both in continual (Frozen-CloGAN) and multi-task settings (MT). For the latter two conditions there is no closed-loop replay of GAN samples. Furthermore, in the MT setting we re-start training at each task switch. The results reported in figure 4 correspond to the maximum accuracies achieved for each task for all 3 variations. We verify that stochastic generation in CloGAN provides an upsampling of the buffer and achieves superior performance to Frozen-CloGAN and MT. We show results for the more challenging datasets, E-MNIST and SVHN.

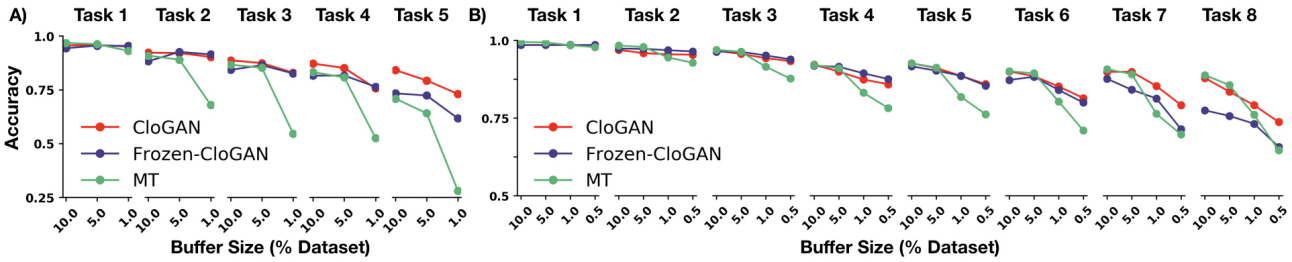

Figure 4: Stochastic Up-Sampling of CloGAN. A) SVHN. B) E-MNIST. The contribution of upsampling is indicated by a positive gap between CloGAN and Frozen-CloGAN as more tasks are learned. We also compare to the MT condition in which training is re-started at each task, eliminating forward-transfer of possible shared task features.

Upsampling is indicated when a positive gap between CloGAN and Frozen-CloGAN increases as more tasks are added. For SVHN, the last task shows clear gaps between CloGAN Frozen-CloGAN as well as MT (maximum gap of 11.39% at task 5). Similarly, E-MNIST shows a clear gap in the last two tasks, 7 and 8 (maximum gap of 10.41% at task 8). Additionally, we show that MT under-performs starting in early tasks due to lack of forward transfer since the networks are re-started from scratch at each task switch. Similar upsampling behavior was observed in MNIST and FASHION, with maximum gaps of 6.84% and 9.35% respectively. Additional figures can be found in the supplementary link.

Per Task Performance

In figure 5A,B), we exhibit per task accuracies along time. Here, CloGAN is shown to produce stable performance throughout consecutive tasks. For both E-MNIST and SVHN all past tasks maintain high accuracies consistently throughout learning of new classes. For example in EMNIST, task 1 preserves its accuracy at 84.33% despite the learning of 7 other tasks in succession. Likewise, SVHN task 1 has an accuracy of 83.87 %. The results are significantly higher when compared to the baseline of catastrophic forgetting and EWC. Moreover, we also display performance for MeRGAN. In E-MNIST, MeRGAN accuracies for tasks 1 and 2 are clearly underperforming CloGAN at the end of training, likely due to image degradation from GAN to GAN transfer.

In figure 5C), we show generated images by CloGAN and MeRGAN for both SVHN and E-MNIST. We list images taken after training of all tasks. For SVHN we list all classes cumulatively learned. For E-MNIST, since there are 24 classes, we limit the display to the two first tasks as well as the last task (8th). For CloGAN, we find that images are sharp even when using small memory sizes, 1% - SHVN and 0.5% - EMNIST. This is true for beginning tasks as well as latter tasks. In contrast, in MeRGAN former taks are sharply more degenerated than latter ones. In EMNIST this can be seen by an overall darkening of letters a through f.

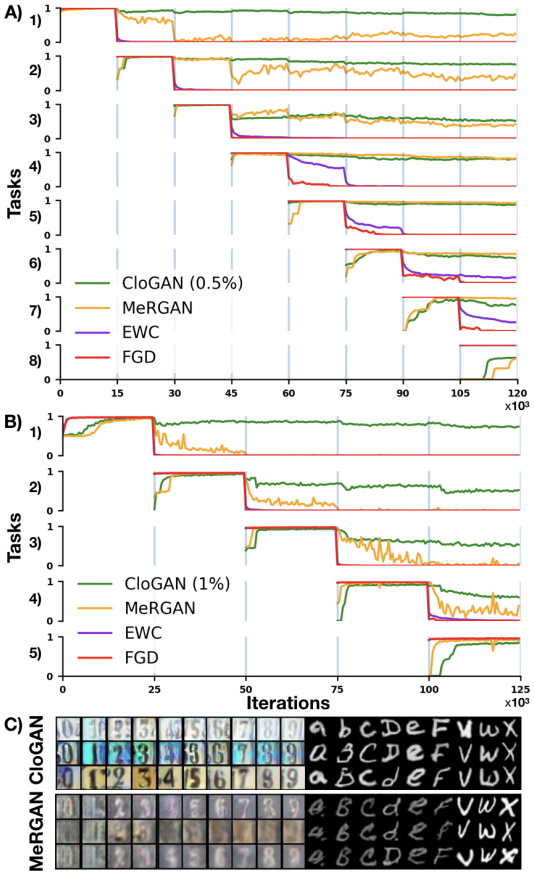

Figure 5: Accuracies per task for EMNIST (A) and SVHN (B). Generated images for SVHN and EMNIST (C). CloGAN preserves performance for early tasks throughout training. MeRGAN has degenerated performance and image generation for early tasks. In EMNIST, degradation is a darkening in the first tasks (a,...,f) in contrast to the last task (v,w,x).

5 Conclusion

In conclusion, we have shown how using very small buffers in conjunction with stochastic replay can give rise to superior performance compared to simple gradient descent, EWC or other replay methods. In our model, CloGAN, the memory buffer acts as an external regularization for the generator, counteracting image degradation through time. Our approach is relatively easy to implement and necessitates only low computation (no full retraining) and memory (small buffer), making it ideal to enable life-long learning on resource-constrained mobile (at the edge) devices.

Acknowledgments

This work was supported by the NSF (CCF-1317433, CNS-1545089), C-BRIC (SRC - DARPA) and Intel Corporation.

References

[Azadi *et al.*, 2018] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian Goodfellow, and Augustus Odena. Discriminator rejection sampling. 2018.

[Cohen *et al.*, 2017] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and André van Schaik. Emnist: an extension of mnist to handwritten letters. *arXiv preprint arXiv:1702.05373*, 2017.

[Farquhar and Gal, 2018] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733v1*, 2018.

[Fernando *et al.*, 2017] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet : Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*, 2017.

[Flesch *et al.*, 2018] Timo Flesch, Jan Balaguer, Ronald Dekker, Hamed Nili, and Christopher Summerfield. Comparing continual task learning in minds and machines. *Proceedings of the National Academy of Sciences*, 115(44):E10313–E10322, 2018.

[French, 1999] Robert M French. Catastrophic forgetting in connectionist networks. 6613(April):128–135, 1999.

[Goldberg and Deb, 1991] David E Goldberg and Kalyanmoy Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of genetic algorithms*, volume 1, pages 69–93. Elsevier, 1991.

[Kemker and Kanan, 2018] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. 2018.

[Kingma *et al.*,] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*.

[Kirkpatrick *et al.*, 2017] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 2017.

[LeCun *et al.*,] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11).

[Li and Hoiem, 2017] Zhizhong Li and Derek Hoiem. Learning without Forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[Lopez-paz and Ranzato, 2017] David Lopez-paz and Marc Aurelio Ranzato. Gradient episodic memory for continual learning. (Advances in Neural Information Processing Systems), 2017.

[MacKay, 2003] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[McCloskey and Cohen, 1989] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[Netzer *et al.*, 2011] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[Nguyen *et al.*, 2018] Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. In *International Conference on Learning Representations*, 2018.

[Odena *et al.*, 2017] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. *Proceedings of the 34th International Conference on Machine Learning, Sydney*, 2017.

[Parisi *et al.*, 2019] German Parisi, Ronald Kemker, Jose Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 2019.

[Rebuffi *et al.*, 2017] Sylvestre-alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl : Incremental classifier and representation learning. *Conference on Computer Vision and Pattern Recognition*, 2017.

[Shin *et al.*, 2017] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in Neural Information Processing Systems*, 2017.

[Wu *et al.*, 2018] Chenshen Wu, Luis Herranz, Xialei Liu, Yaxing Wang, Joost van de Weijer, and Bogdan Raducanu. Memory replay gans: learning to generate images from new categories without forgetting. *Advances in Neural Information Processing Systems*, 2018.

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

[Zenke *et al.*, 2017] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual Learning Through Synaptic Intelligence. *Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017*, 2017.