Generative Adversarial Networks and Continual Learning*

Kevin J Liang[1], **Chunyuan Li**[1,2], **Guoyin Wang**[1] **& Lawrence Carin**[1]
[1]Duke University [2]Microsoft Research
{kevin.liang, chunyuan.li, guoyin.wang, lcarin}@duke.edu

Abstract

There is a strong emphasis in the continual learning literature on sequential classification experiments, where each task bares little semblance to previous ones. While certainly a form of continual learning, such tasks do not accurately represent many continual learning problems of the real-world, where the data distribution often evolves slowly over time. We propose using Generative Adversarial Networks (GANs) as a potential source for generating potentially unlimited datasets of this nature. We also identify that the dynamics of GAN training naturally constitute a continual learning problem, and show that leveraging continual learning methods can improve performance. As such, we show that techniques from both continual learning and GAN, typically studied separately, can be used to each other's benefit.

1 Introduction

The ability to learn new things continually while retaining previously acquired knowledge is a desirable attribute of an intelligent system. Humans and other forms of life do this well, but neural networks are known to exhibit a phenomenon known as *catastrophic forgetting* [12, 19]: the gradients that adapt a neural network's parameters to perform a new task tend to also clobber the model's ability to perform old ones. Because of its broad importance to the general field of machine learning, recent years have seen increased interest in approaches that enable *continual learning* (e.g. [9, 27, 11, 16, 20, 25]). These methods focus on improving the model architecture, objective, or training procedure to preserve knowledge of prior tasks while still enabling learning of new ones.

However, many of these works tend to conduct experiments that focus on learning a sequence of disparate tasks, which while certainly a continual learning task, does not capture the dynamics of a setting in which the data slowly evolves over time, as opposed to making abrupt discontinuous jumps. Such situations are common in many real-world applications, as deployed systems must maintain performance in an ever-evolving environment. It is therefore desirable for experiments in the literature to reflect this setting, but datasets that evolve over time are not readily available, which makes applying continual learning methods to such circumstances difficult.

On the other hand, recent years have seen an enormous amount of progress made in generative models, specifically with the advent of Generative Adversarial Networks (GANs) [3]. GANs have demonstrated the ability to learn impressively complex distributions [8, 1] from data samples alone. Interestingly, since GANs are capable of learning conditional distributions [14], and because the distribution of the generator's outputs smoothly evolves as training progresses, GANs represent an opportunity for producing a labeled dataset that varies through time.

Importantly though, the implications of the generator's distribution varying through time go beyond the potential for new sequential task benchmarks for continual learning. GANs are known to be somewhat challenging to train, with mode collapse a common problem. Inspection of a collapsed

*Part of submission to the International Conference on Learning Representations (ICLR) 2019

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

(a) Iteration 11960 (b) Iteration 12000 (c) Iteration 12160 (d) Iteration 12380

Figure 1: Real samples from a mixture of eight Gaussians in red; generated samples in blue. (a) The generator is mode collapsed in the bottom right. (b) The discriminator learns to recognize the generator oversampling this region and pushes the generator away, so the generator gravitates toward a new mode. (c) The discriminator continues to chase the generator, causing the generator to move in a clockwise direction. (d) The generator eventually returns to the same mode as (a). Such oscillations are common while training a vanilla GAN. Best seen as a video: https://youtu.be/91a2gPWngo8.

generator over subsequent training iterations reveal that rather than converging to a stationary distribution, mode-collapsed generators tend to oscillate wildly, oftentimes revisiting previous locations of the data space—modes that the discriminator presumably had previously learned to recognize as fake (see Figure 1). We conjecture this phenomenon is at least in part enabled by catastrophic forgetting in the discriminator: during training, synthesized fakes are presented to the discriminator in a sequential manner reminiscent of the way tasks are learned in continual learning literature. Since the discriminator is typically not refreshed with earlier synthesized samples, it loses its ability to recognize them, allowing the generator to oscillate back to previous locations.

With these perspectives in mind, we make the following observations and contributions:

- Experiments in continual learning focus on sequences of disjoint tasks and do not cover the more realistic scenario where a model encounters an evolving data distribution. GANs represent an opportunity to fill this gap by synthesizing datasets that have the requisite time component.
- The training of a GAN discriminator is a continual learning problem. We show that augmenting GAN models with continual learning methods improves performance on benchmark datasets.

2 Methods

2.1 GAN-generated datasets for continual learning

Consider distribution $p_{\text{real}}(\boldsymbol{x})$, from which we have data samples $\mathcal{D}^{\text{real}}$. We seek to learn a mapping from an easy-to-sample distribution $p(\boldsymbol{z})$ (e.g. standard normal) to a data distribution $p_{\text{gen}}(\boldsymbol{x})$, which we want to match $p_{\text{real}}(\boldsymbol{x})$. This mapping is parameterized as a neural network $G_{\boldsymbol{\phi}}(\boldsymbol{z})$ with parameters $\boldsymbol{\phi}$, termed the *generator*. The synthesized data are drawn $\boldsymbol{x} = G_{\boldsymbol{\phi}}(\boldsymbol{z})$, with $\boldsymbol{z} \sim p(\boldsymbol{z})$. In the GAN [3] set-up, we simultaneously learn another neural network $D_{\boldsymbol{\theta}}(\boldsymbol{x}) \in [0, 1]$ with parameters $\boldsymbol{\theta}$, termed the *discriminator*, which provides feed-back to $G_{\boldsymbol{\phi}}(\boldsymbol{z})$. Trained by a min-max objective in conjunction with the generator, the generator gradually evolves: initial generations resemble random noise, but eventually grow to resemble $\mathcal{D}^{\text{real}}$. At any point during training, an unlimited number of samples can be drawn from $G_{\boldsymbol{\phi}}(\boldsymbol{z})$. Therefore, at any training iteration t, we can generate a dataset $\mathcal{D}_t^{gen}$, and because $p_{\text{gen}}(\boldsymbol{x})$ smoothly evolves with t, so does the sequence of datasets $\mathcal{D}_1^{gen}, ..., \mathcal{D}_T^{gen}$.

As an example, we can train a DCGAN [18] on MNIST and generate an entire "fake" dataset of 70K samples every 50 training iterations of the DCGAN generator. We propose performing learning on each of these generated datasets as individual tasks for continual learning. Selected samples are shown in Figure 3 of Appendix A from the datasets $\mathcal{D}_t^{gen}$ for $t \in \{5, 10, 15, 20\}$, each generated from the same 100 samples of z for all t. By conditioning the GAN [14] on randomly generated labels, we have a mechanism for generating *labeled* datasets. With the success of large-scale GANs [1], a similar method can be used to generate time-varying ImageNet datasets.

2.2 Continual learning for GAN discriminators

The traditional continual learning methods like Elastic Weight Consolidation (EWC) [9] or Intelligent Synapses (IS) [27][1] are designed for certain canonical benchmarks, commonly consisting of a small number of clearly defined tasks (e.g., classification datasets in sequence). In GANs, the discriminator is trained on dataset $\mathcal{D}_t = \{\mathcal{D}^{\text{real}}, \mathcal{D}_t^{\text{gen}}\}$ at each iteration t. However, because of the evolution of the generator, the distribution $p_{\text{gen}}(x)$ from which $\mathcal{D}_t^{\text{gen}}$ comes changes over time.

[1]Summary of both of these methods can be found in Appendix B

As such, we argue that different instances in time of the generator should be viewed as separate tasks. Specifically, in the parlance of continual learning, the training data are to be regarded as $\mathcal{D} = \{(\mathcal{D}^{\text{real}}, \mathcal{D}_1^{\text{gen}}), (\mathcal{D}^{\text{real}}, \mathcal{D}_2^{\text{gen}}), ...\}$. Thus motivated, we would like to apply continual learning methods to the discriminator, but doing so is not straightforward for the following reasons:

- **Definition of a task**: EWC and IS were originally proposed for discrete, well-defined tasks. For GAN, there is no such precise definition as to what a "task" is, and as discriminators are not typically trained to convergence at every iteration, it is also unclear how long a task should be.
- **Computational memory**: While Equations 3 and 5 are for two tasks, they can be extended to K tasks by adding an additional loss term for each of the $K-1$ prior tasks. As each loss term requires saving both a historical reference term $\boldsymbol{\theta}_k^*$ and either a diagonal Fisher Information matrix F_k or importance weights $\boldsymbol{\omega}_k$ (all of which are the same size as the model parameters $\boldsymbol{\theta}$) for each task k, employing these techniques naively quickly becomes impractical for bigger models when K gets large, especially if K is set to the number of training iterations T.
- **Continual *not* learning**: Early iterations of the discriminator are likely to be non-optimal, and without a forgetting mechanism, EWC and IS may forever lock the discriminator to a poor initialization. Additionally, the unconstrained addition of a large number of loss terms will cause the continual learning regularization term to grow unbounded, which can disincentivize any further changes in $\boldsymbol{\theta}$.

To address these issues, we build upon EWC and IS by proposing several changes:

Number of tasks as a rate: We choose the total number of tasks K as a function of a constant rate α, which denotes the number of iterations before the conclusion of a task, as opposed to arbitrarily dividing the GAN training iterations into some set number of segments. Given T training iterations, this means a rate α yields $K = \frac{T}{\alpha}$ tasks.

Online Memory: Seeking a way to avoid storing extra $\boldsymbol{\theta}_k^*$, F_k, or $\boldsymbol{\omega}_k$, we observe that the sum of two or more quadratic forms is another quadratic, which gives the classifier loss with continual learning the following form for the $(k+1)^{\text{th}}$ task:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_{k+1}(\boldsymbol{\theta}) + \mathcal{L}^{\text{CL}}(\boldsymbol{\theta}), \quad \text{with} \quad \mathcal{L}^{\text{CL}}(\boldsymbol{\theta}) \triangleq \frac{\lambda}{2} \sum_i S_{k,i}(\theta_i - \bar{\theta}_{k,i}^*)^2 , \tag{1}$$

where $\bar{\theta}_{k,i}^* = \frac{P_{k,i}}{S_{k,i}}$, $S_{k,i} = \sum_{\kappa=1}^{k} Q_{\kappa,i}$, $P_{k,i} = \sum_{\kappa=1}^{k} Q_{\kappa,i}\theta_{\kappa,i}^*$, and $Q_{\kappa,i}$ is either $F_{\kappa,i}$ or $\omega_{\kappa,i}$, depending on the method. We name models with EWC and IS augmentations EWC-GAN and IS-GAN, respectively.

Controlled forgetting: To provide a mechanism for forgetting earlier non-optimal versions of the discriminator and to keep $\mathcal{L}^{\text{CL}}$ bounded, we add a discount factor γ: $S_{k,i} = \sum_{\kappa=1}^{k} \gamma^{k-\kappa} Q_{\kappa,i}$ and $P_{k,i} = \sum_{\kappa=1}^{k} \gamma^{k-\kappa} Q_{\kappa,i}\theta_{\kappa,i}^*$. Together, α and γ determine how far into the past the discriminator remembers previous generator distributions, and λ controls how important memory is relative to the discriminator loss. Note, the terms S_k and P_k can be updated every α steps in an online fashion:

$$S_{k,i} = \gamma S_{k-1,i} + Q_{k,i}, \qquad P_{k,i} = \gamma P_{k-1,i} + Q_{k,i}\theta_{k,i}^* \tag{2}$$

This allows the EWC or IS loss to be applied without necessitating storing either Q_k or $\boldsymbol{\theta}_k^*$ for every task k, which would quickly become too costly to be practical. Only a single variable to store a running average is required for each of S_k and P_k, making this method space efficient.

Note that the training of the generator remains the same. Here we have shown two methods to mitigate catastrophic forgetting for the original GAN; however, the proposed framework is applicable to almost all of the wide range of GAN setups. Similarly, while we focus on EWC and IS here, any continual learning method can be applied in a similar way.

3 Related work

There has been previous work investigating continual learning within the context of GANs. Improved GAN [21] introduced historical averaging, which regularizes the model with a running average of parameters of the most recent iterations. Simulated+Unsupervised training [23] proposed replacing half of each minibatch with previous generator samples during training of the discriminator, as previous generations should always be considered fake. However, this necessitates a historical buffer of samples and halves the number of current samples that can be considered. Continual Learning

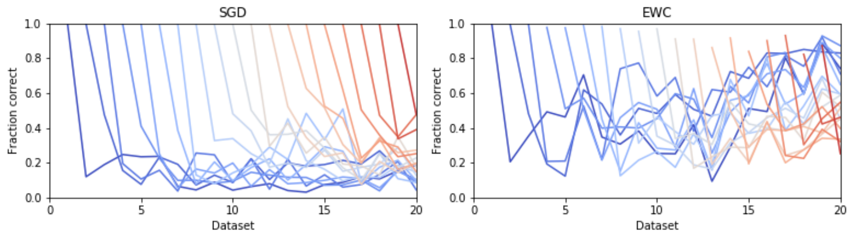


Figure 2: Each line represents the discriminator's test accuracy on the fake GAN datasets. Note the sharp decrease in the discriminator's ability to recognize previous fake samples upon fine-tuning on the next dataset using SGD (left). Forgetting still occurs with EWC (right), but is less severe.

Table 1: Image generation quality on CelebA and CIFAR-10

	CelebA	CIFAR-10	
Method	FID $\downarrow$	FID $\downarrow$	ICP $\uparrow$
DCGAN	12.52	41.44	6.97 ± 0.05
DCGAN + EWC	10.92	34.84	7.10 ± 0.05
WGAN-GP	-	30.23	7.09 ± 0.06
WGAN-GP + EWC	-	29.67	7.44 ± 0.08
SN-DCGAN	-	27.21	7.43 ± 0.10
SN-DCGAN + EWC	-	25.51	7.58 ± 0.07

GAN [22] applies EWC to GAN, as we have, but uses it in the context of the class-conditioned generator that learns classes sequentially, as opposed to all at once, as we propose. [24] independently makes a similar observation on the continual learning nature of GAN training, but propose momentum and gradient penalty solutions instead and restrict themselves to experiments on toy examples.

4 Experiments

4.1 Sequential discrimination

While Figure 1 implies catastrophic forgetting in a GAN discriminator, we can show this concretely. Using the DCGAN-generated MNIST datasets $\mathcal{D}_1^{gen}, ..., \mathcal{D}_T^{gen}$ described in Section 2.1, we now train a discriminator to convergence on each $\mathcal{D}_t^{\text{gen}}$ in sequence. Importantly, we do *not* include samples from $\mathcal{D}_{<t}^{\text{gen}}$ while fine-tuning on $\mathcal{D}_t^{\text{gen}}$. After fine-tuning on the train split of dataset $\mathcal{D}_t^{\text{gen}}$, the percentage of generated examples correctly identified as fake by the discriminator is evaluated on the test splits of $\mathcal{D}_{\leq t}^{\text{gen}}$, with and without EWC (Figure 2). The catastrophic forgetting effect of the discriminator trained with SGD is clear, with a steep drop-off in discriminating ability on $\mathcal{D}_{t-1}^{\text{gen}}$ after fine-tuning on $\mathcal{D}_t^{\text{gen}}$; this is unsurprising, as $p_{\text{gen}}(x)$ has evolved specifically to deteriorate discriminator performance. While there is still a dropoff with EWC, forgetting is less severe. On the other hand, there is certainly room to improve, demonstrating the value of considering such kinds of datasets for continual learning methods.

4.2 Augmenting GAN with continual learning

We augment the discriminators of various popular GANs implementations with EWC to preserve recognition of previously seen generations, testing on two image datasets, CelebA and CIFAR-10. Comparisons are made with the TTUR [6] variants of DCGAN [18] and WGAN-GP [4], as well as an implementation of a spectral normalized [15] DCGAN (SN-DCGAN). Without modifying the learning rate or model architecture, we show results with and without the EWC loss term added to the discriminator for each. Performance is quantified with the Fréchet Inception Distance (FID) [6] for both datasets. Since labels are available for CIFAR-10, we also report ICP for that dataset. Best values are reported in Table 1. In each model, we see improvement in both FID and ICP from the addition of EWC to the discriminator. Additional experiments improving GAN with continual learning can be found in Appendix C.

5 Conclusion

We have identified the connections between GANs and continual learning: the training dynamics of GAN naturally form a continual learning problem. This perspective allows us to show that (1) GAN-generated datasets provide opportunities to form more realistic continual learning benchmarks; (2) Existing continual learning methods can be adjusted to improve GAN training. Extensive experimental results have demonstrated the proposed observations and solutions.

References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. *arXiv preprint*, 2018.

[2] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dolí, and C Lawrence Zitnick. Microsoft COCO Captions: Data Collection and Evaluation Server. *arXiv preprint*, 2015.

[3] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. *Advances In Neural Information Processing Systems*, 2014.

[4] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved Training of Wasserstein GANs. *Advances In Neural Information Processing Systems*, 2017.

[5] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long Text Generation via Adversarial Training with Leaked Information. *AAAI Conference on Artificial Intelligence*, 2018.

[6] 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. *Advances In Neural Information Processing Systems*, 2017.

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

[8] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. *International Conference on Learning Representations*, 2018.

[9] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming Catastrophic Forgetting in Neural Networks. *Proceedings of the National Academy of Sciences*, 2017.

[10] Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, and Ming-Ting Sun. Adversarial ranking for language generation. *Advances in Neural Information Processing Systems*, 2017.

[11] David Lopez-Paz and Marc ' Aurelio Ranzato. Gradient Episodic Memory for Continual Learning. *Advances In Neural Information Processing Systems*, 2017.

[12] Michael McCloskey and Neal J Cohen. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem. *The Psychology of Learning and Motivation*, 1989.

[13] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. Unrolled Generative Adversarial Networks. *International Conference on Learning Representations*, 2017.

[14] Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. *arXiv preprint*, 2014.

[15] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. *International Conference on Learning Representations*, 2018.

[16] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational Continual Learning. *International Conference on Learning Representations*, 2017.

[17] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. *Annual Meeting of the Association for Computational Linguistics*, 2002.

[18] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *International Conference on Learning Representations*, 2016.

[19] Roger Ratcliff. Connectionist Models of Recognition Memory: Constraints Imposed by Learning and Forgetting Functions. *Psychology Review*, 1990.

[20] Hippolyt Ritter, Aleksandar Botev, and David Barber. Online Structured Laplace Approximations For Overcoming Catastrophic Forgetting. *Advances In Neural Information Processing Systems*, 2018.

[21] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. *Advances In Neural Information Processing Systems*, 2016.

[22] Ari Seff, Alex Beatson, Daniel Suo, and Han Liu. Continual Learning in Generative Adversarial Nets. *arXiv preprint*, 2018.

[23] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russ Webb. Learning from Simulated and Unsupervised Images through Adversarial Training. *Conference on Computer Vision and Pattern Recognitionn*, 2017.

[24] Hoang Thanh-Tung, Truyen Tran, and Svetha Venkatesh. On catastrophic forgetting and mode collapse in Generative Adversarial Networks. *arXiv preprint*, 2018.

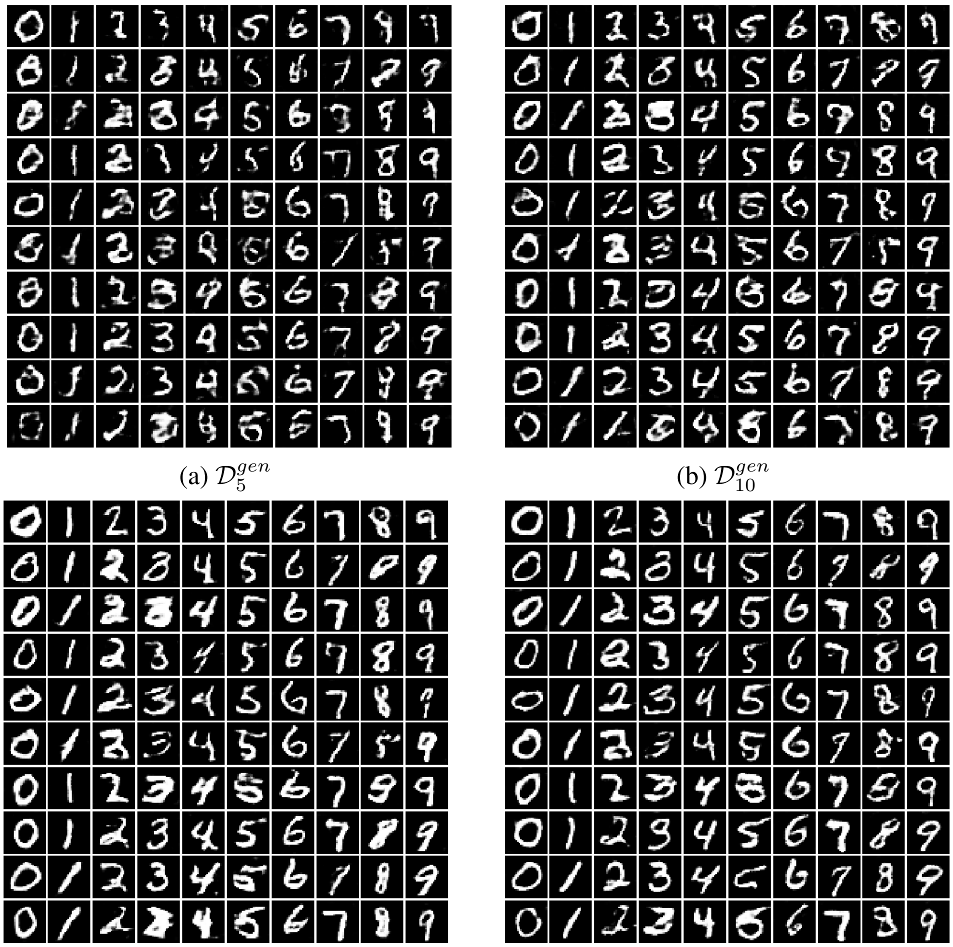
[25] Ju Xu and Zhanxing Zhu. Reinforced Continual Learning. *Advances In Neural Information Processing Systems*, 2018.

[26] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. *AAAI Conference on Artificial Intelligence*, 2017.

[27] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual Learning Through Synaptic Intelligence. *International Conference on Machine Learning*, 2017.

[28] Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. Adversarial feature matching for text generation. *ICML*, 2017.

A Samples from generated MNIST datasets



(a) $\mathcal{D}_5^{gen}$ (b) $\mathcal{D}_{10}^{gen}$

(c) $\mathcal{D}_{15}^{gen}$ (d) $\mathcal{D}_{20}^{gen}$

Figure 3: Image samples from generated "fake MNIST" datasets

B Continual learning methods summary

B.1 Elastic weight consolidation (EWC)

To derive the EWC loss, [9] frames training a model as finding the most probable values of the parameters $\boldsymbol{\theta}$ given the data $\mathcal{D}$. For two tasks, the data are assumed partitioned into independent sets according to the task, and the posterior for Task 1 is approximated as a Gaussian with mean centered on the optimal parameters for Task 1 $\boldsymbol{\theta}_1^*$ and diagonal precision given by the diagonal of the Fisher information matrix F_1 at $\boldsymbol{\theta}_1^*$. This gives the EWC loss the following form:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_2(\boldsymbol{\theta}) + \mathcal{L}^{\mathrm{EWC}}(\boldsymbol{\theta}), \quad \text{with} \quad \mathcal{L}^{\mathrm{EWC}}(\boldsymbol{\theta}) \triangleq \frac{\lambda}{2} \sum_i F_{1,i} (\theta_i - \theta_{1,i}^*)^2 \,, \tag{3}$$

where $\mathcal{L}_2(\boldsymbol{\theta}) = \log p(\mathcal{D}_2|\boldsymbol{\theta})$ is the loss for Task 2 individually, λ is a hyperparameter representing the importance of Task 1 relative to Task 2, $F_{1,i} = \big(\frac{\partial \mathcal{L}_1(\boldsymbol{\theta})}{\partial \theta_i}\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_1^*}\big)^2$, i is the parameter index, and $\mathcal{L}(\boldsymbol{\theta})$ is the new loss to optimize while learning Task 2. Intuitively, the EWC loss prevents the model from straying too far away from the parameters important for Task 1 while leaving less crucial parameters free to model Task 2. Subsequent tasks result in additional $\mathcal{L}^{\mathrm{EWC}}(\boldsymbol{\theta})$ terms added to the loss for each previous task. By protecting the parameters deemed important for prior tasks, EWC as a regularization term allows a single neural network (assuming sufficient parameters and capacity) to learn new tasks in a sequential fashion, without forgetting how to perform previous tasks.

B.2 Intelligent synapses (IS)

While EWC makes a point estimate of how essential each parameter is at the conclusion of a task, IS [27] protects the parameters according to their importance along the task's entire training trajectory. Termed synapses, each parameter θ_i of the neural network is awarded an *importance measure* $\omega_{1,i}$ based on how much it reduced the loss while learning Task 1. Given a loss gradient $\boldsymbol{g}(t) = \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}_t}$ at time t, the total change in loss during the training of Task 1 then is the sum of differential changes in loss over the training trajectory. With the assumption that parameters $\boldsymbol{\theta}$ are independent, we have:

$$\int_{t^0}^{t^1} \boldsymbol{g}(t) d\boldsymbol{\theta} = \int_{t^0}^{t^1} \boldsymbol{g}(t)\boldsymbol{\theta}' dt = \sum_i \int_{t^0}^{t^1} g_i(t)\theta_i' dt \triangleq -\sum_i \omega_{1,i} \,, \tag{4}$$

where $\boldsymbol{\theta}' = \frac{d\boldsymbol{\theta}}{dt}$ and (t^0, t^1) are the start and finish of Task 1, respectively. Note the added negative sign, as importance is associated with parameters that decrease the loss.

The importance measure $\omega_{1,i}$ can now be used to introduce a regularization term that protects parameters important for Task 1 from large parameter updates, just as the Fisher information matrix diagonal terms $F_{1,i}$ were used in EWC. This results in an IS loss very reminiscent in form:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_2(\boldsymbol{\theta}) + \mathcal{L}^{\text{IS}}(\boldsymbol{\theta}), \quad \text{with} \quad \mathcal{L}^{\text{IS}}(\boldsymbol{\theta}) \triangleq \frac{\lambda}{2} \sum_i \omega_{1,i}(\theta_i - \theta_{1,i}^*)^2 \,. \tag{5}$$

Note that [27] instead consider $\Omega_{1,i} = \frac{\omega_{1,i}}{(\Delta_{1,i})^2+\xi}$, where $\Delta_{1,i} = \theta_{1,i} - \theta_{0,i}$ and ξ is a small number for numerical stability. We however found that the inclusion of $(\Delta_{1,i})^2$ can lead to the loss exploding and then collapsing as the number of tasks increases and so omit it. We also change the hyperparameter c into $\frac{\lambda}{2}$.

C Additional Experiments

C.1 Mixture of eight Gaussians

We show results on a toy dataset consisting of a mixture of eight Gaussians, as in the example in Figure 1. Following the setup of [13], the real data are evenly distributed among eight 2-dimensional Gaussian distributions arranged in a circle of radius 2, each with covariance $0.02I$ (see Figure 4). We evaluate our model with Inception Score (ICP) [21], which gives a rough measure of diversity and quality of samples; higher scores imply better performance, with the true data resulting in a score of around 7.870. For this simple dataset, since we know the true data distribution, we also calculate the symmetric Kullback–Leibler divergence (Sym-KL); lower scores mean the generated samples are closer to the true data. We show computation time, measured in numbers of training iterations per second (Iter/s), averaged over the full training of a model on a single Nvidia Titan X (Pascal) GPU. Each model was run 10 times, with the mean and standard deviation of each performance metric at the end of 25K iterations reported in Table 2.

The performance of EWC-GAN and IS-GAN were evaluated for a number of hyperparameter settings. We compare our results against a vanilla GAN [3], as well as a state-of-the-art GAN with spectral normalization

Table 2: Iterations per second, inception score, and symmetric KL divergence comparison on a mixture of eight Gaussians.

Model						
Method	α	λ	γ	Iter/s ↑	ICP ↑	Sym-KL ↓
GAN	-	-	-	87.59 ± 1.45	2.835 ± 2.325	19.55 ± 3.07
GAN + ℓ_2 weight	1	0.01	0		5.968 ± 1.673	15.19 ± 2.67
GAN + historical avg.	1	0.01	0.995		7.305 ± 0.158	13.32 ± 0.88
GAN + SN	-	-	-	49.70 ± 0.13	6.762 ± 2.024	13.37 ± 3.86
GAN + IS	1000	100	0.8	42.26 ± 0.35	7.039 ± 0.294	15.10 ± 1.51
GAN + IS	100	10	0.98	42.29 ± 0.10	7.500 ± 0.147	11.85 ± 0.92
GAN + IS	10	100	0.99	41.07 ± 0.07	7.583 ± 0.242	11.88 ± 0.84
GAN + SN + IS	10	100	0.99	25.69 ± 0.09	7.699 ± 0.048	11.10 ± 1.18
GAN + EWC	1000	100	0.8	82.78 ± 1.55	7.480 ± 0.209	13.00 ± 1.55
GAN + EWC	100	10	0.98	80.63 ± 0.39	7.488 ± 0.222	12.16 ± 1.64
GAN + EWC	10	10	0.99	73.86 ± 0.16	7.670 ± 0.112	11.90 ± 0.76
GAN + SN + EWC	10	10	0.99	44.68 ± 0.11	7.708 ± 0.057	11.48 ± 1.12

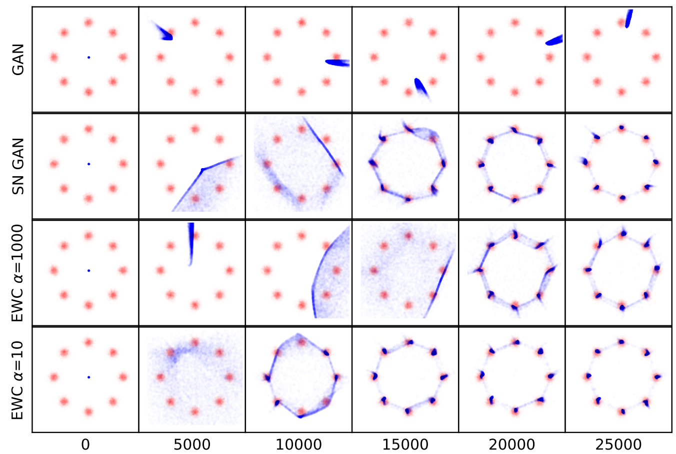


Figure 4: Each row shows the evolution of generator samples at 5000 training step intervals for GAN, SN-GAN, and EWC-GAN for two α values. The proposed EWC-GAN models have hyperparameters matching the corresponding α in Table 2. Each frame shows 10000 samples drawn from the true eight Gaussians mixture (red) and 10000 generator samples (blue).

(SN) [15] applied to the discriminator. As spectral normalization augments the discriminator loss in a way different from continual learning, we can combine the two methods; this variant is also shown.

Note that a discounted version of discriminator historical averaging [21] can be recovered from the EWC and IS losses if the task rate $\alpha = 1$ and $Q_{k,i} = 1$ for all i and k, a poor approximation to both the Fisher information matrix diagonal and importance measure. If we also set the historical reference term $\bar{\theta}_k^*$ and the discount factor γ to zero, then the EWC and IS losses become ℓ_2 weight regularization. These two special cases are also included for comparison.

We observe that augmenting GAN models with EWC and IS consistently results in generators that better match the true distribution, both qualitatively and quantitatively, for a wide range of hyperparameter settings. EWC-GAN and IS-GAN result in a better ICP and FID than ℓ_2 weight regularization and discounted historical averaging, showing the value of prioritizing protecting important parameters, rather than all parameters equally. EWC-GAN and IS-GAN also outperform a state-of-the-art method in SN-GAN. In terms of training time, updating the EWC loss requires forward propagating a new minibatch through the discriminator and updating S and P, but even if this is done at every step ($\alpha = 1$), the resulting algorithm is only slightly slower than SN-GAN. Moreover, doing so is unnecessary, as higher values of α also provide strong performance for a much smaller time penalty. Combining EWC with SN-GAN leads to even better results, showing that the two methods can complement each other. IS-GAN can also be successfully combined with SN-GAN, but it is slower than EWC-GAN as it requires tracking the trajectory of parameters at each step. Sample generation evolution over time is shown in Figure 4.

C.2 Text generation of COCO Captions

We also consider the text generation on the MS COCO Captions dataset [2], with the pre-processing in [5]. Quality of generated sentences is evaluated by BLEU score [17]. Since BLEU-b measures the overlap of b consecutive words between the generated sentences and ground-truth references, higher BLEU scores indicate better fluency. Self BLEU uses the generated sentences themselves as references; lower values indicate higher diversity.

We apply EWC and IS to textGAN [28], a recently proposed model for text generation in which the discriminator uses feature matching to stabilize training. This model's results (labeled "EWC" and "IS") are compared to a Maximum Likelihood Estimation (MLE) baseline, as well as several state-of-the-art methods: SeqGAN [26], RankGAN [10], GSGAN [7] and LeakGAN [5]. Our variants of textGAN outperforms the vanilla textGAN for all BLEU scores (see Table 3), indicating the effectiveness of addressing the forgetting issue for GAN training in text generation. EWC/IS + textGAN also demonstrate a significant improvement compared with other methods, especially on BLEU-2 and 3. Though our variants lag slightly behind LeakGAN on BLEU-4 and 5, their self BLEU scores (Table 4) indicate it generates more diverse sentences. Sample sentence generations can be found in Table 5.

Table 3: Test BLEU ↑ results on MS COCO

Method	MLE	SeqGAN	RankGAN	GSGAN	LeakGAN	textGAN	EWC	IS
BLEU-2	0.820	0.820	0.852	0.810	0.922	0.926	0.934	0.933
BLEU-3	0.607	0.604	0.637	0.566	0.797	0.781	0.802	0.791
BLEU-4	0.389	0.361	0.389	0.335	0.602	0.567	0.594	0.578
BLEU-5	0.248	0.211	0.248	0.197	0.416	0.379	0.400	0.388

Table 4: Self BLEU ↓ results on MS COCO

Method	MLE	SeqGAN	RankGAN	GSGAN	LeakGAN	textGAN	EWC	IS
BLEU-2	0.754	0.807	0.822	0.785	0.912	0.843	0.854	0.853
BLEU-3	0.511	0.577	0.592	0.522	0.825	0.631	0.671	0.655
BLEU-4	0.232	0.278	0.288	0.230	0.689	0.317	0.388	0.364

Table 5: Sample sentence generations from EWC + textGAN

a couple of people are standing by some zebras in the background
the view of some benches near a gas station
a brown motorcycle standing next to a red fence
a bath room with a broken tank on the floor
red passenger train parked under a bridge near a river
some snow on the beach that is surrounded by a truck
a cake that has been perform in the background for takeoff
a view of a city street surrounded by trees
two giraffes walking around a field during the day
crowd of people lined up on motorcycles
two yellow sheep with a baby dog in front of other sheep
an intersection sits in front of a crowd of people
a red double decker bus driving down the street corner
an automobile driver stands in the middle of a snowy park
five people at a kitchen setting with a woman
there are some planes at the takeoff station
a passenger airplane flying in the sky over a cloudy sky
three aircraft loaded into an airport with a stop light
there is an animal walking in the water
an older boy with wine glasses in an office
two old jets are in the middle of london
three motorcycles parked in the shade of a crowd
group of yellow school buses parked on an intersection
a person laying on a sidewalk next to a sidewalk talking on a cell phone
a chef is preparing food with a sink and stainless steel appliances