Generative Memory for Continual Learning

Ivan Skorokhodov

November 26, 2019

Contents

Deep Generative Replay (DGR) [1]

Main idea (1/3)

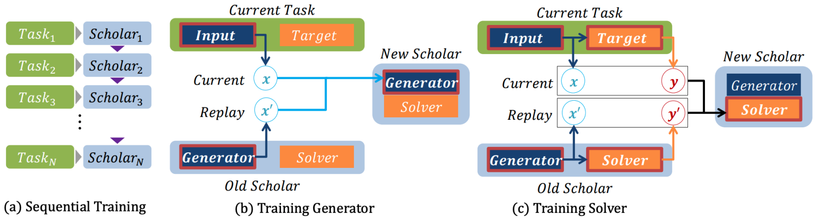
- Train a generator G_1, train a classifier C_1 for task #1
- For task $t > 1$ generate images with G_{t-1}, generate labels with C_{t-1} to obtain a dataset $\hat{D}_{:t}$
- Train on both $\hat{D}_{:t}$ and D_t (real data for task t) jointly
- Note: it's not clear from the paper if they trained C_t on the logits of C_{t-1} or its one-hot predictions
- Note: it's a bit odd that they do not train conditional generator

$$\mathcal{L} = r \mathop{\mathbb{E}}_{(\boldsymbol{x},\boldsymbol{y}) \sim D_i} \left[L\left(C_t(\boldsymbol{x}), \boldsymbol{y} \right) \right] + (1 - r) \mathop{\mathbb{E}}_{\boldsymbol{x}' \sim G_{t-1}} \left[L(C_t(\boldsymbol{x}'), C_{t-1}(\boldsymbol{x}')) \right]$$

[1] "Continual Learning with Deep Generative Replay" by Shin et al., NeurIPS 2017

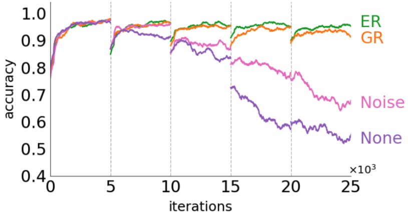
Deep Generative Replay (DGR)

Illustration (2/3)



Deep Generative Replay (DGR)

Results on permuted MNIST (3/3)



- 5 tasks
- ER — Joint Multi-Task baseline
- Noise — feeding random noise instead of images into C_{t-1} to distill knowledge

Memory Replay GAN [2]

Main idea (1/3)

- Idea is simple: train a generative memory G_t, save its snapshot before each new task and distill its knowledge into a new one G_{t+1}
- There are two ways to distill the knowledge
  - Generate synthetic data and mix it into a new one S'_t (Joint Retraining)
  - Perform real knowledge distillation (Replay Alignment):

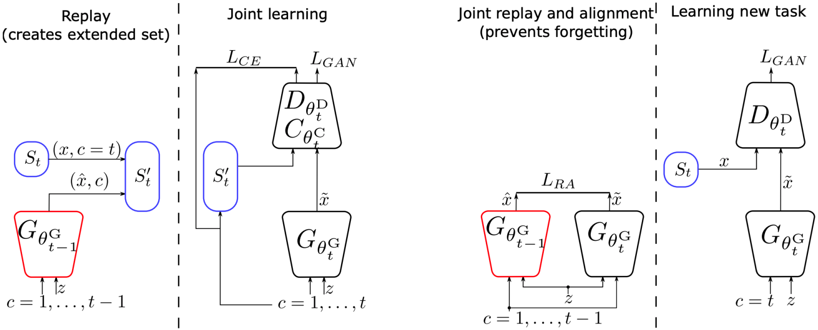
$$\mathcal{L}_G = L_G(\theta_t, S_t) + \lambda \mathop{\mathbb{E}}_{z \sim p_z, c \sim U(0, t-1)} \left[\| G_t(z, c) - G_{t-1}(z, c) \|^2 \right]$$

- Also train a classifier on top of GAN

[2] "Memory Replay GANs: Learning to Generate New Categories without Forgetting" by Wu et al., NeurIPS 2018

Memory Replay GAN (MeRGAN)

Illustration (2/3)

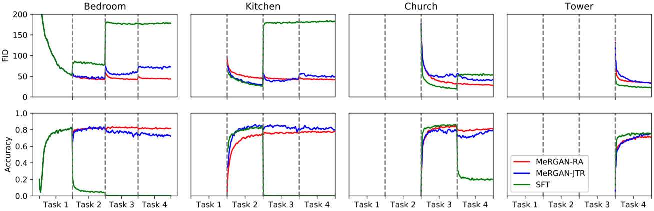


(a) Joint retraining with replay

(b) Replay alignment

Memory Replay GAN (MeRGAN)

Results (3/3)



- SFT (sequential fine tuning) is no tricks at all
- Replay Alignment tends to work better
- Authors are not clear about how they have measured the accuracy, as far as I got — they have trained a classifier on real data and measured its performance on the fake data.

Dynamic Generative Memory (DGM) [4]

Main idea (1/3)

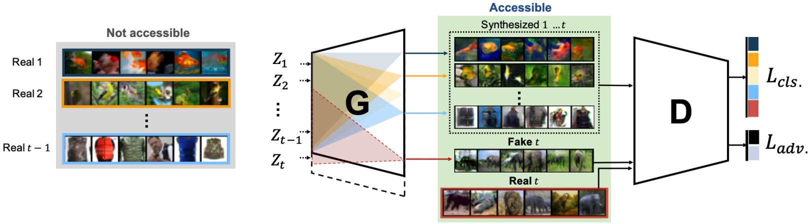
- Authors consider class-incremental learning (just as we do): classes arrive sequentially and we evaluate the performance on all the tasks
- They do not run any kind of knowledge distillation and follow HAT[3] approach instead
- More precisely, for task t, for layer l of the generator they train a binary mask m_l^t and multiply layer's weights on this mask
- Binary mask m_l^t is regularized to be sparse
- Previously learned weights are not updated in the future (but network can learn to ignore them by learning the corresponding mask)
- But compared to HAT, authors are cheating: they add new neurons to the generator after each task to preserve its capacity

[3] "Overcoming Catastrophic Forgetting with Hard Attention to the Task" by Serra et al., ICML 2018

[4] "Learning to Remember: A Synaptic Plasticity Driven Framework for Continual Learning" by Ostapenko et al., arxiv

Dynamic Generative Memory (DGM)

Illustration (2/3)



Dynamic Generative Memory (DGM)

Results (3/3)

	Method	MNIST (%) A_5	MNIST (%) A_{10}	SVHN(%) A_5	SVHN(%) A_{10}	CIFAR10(%) A_5	CIFAR10(%) A_{10}	ImageNet-50(%) A_{30}	ImageNet-50(%) A_{50}
	JT	99.87	99.24	92.99	88.72	83.40	77.82	57.35	49.88
Episodic memory	iCarl-S [22]	-	55.8	-	-	-	-	29.38	**28.98**
	EWC-S[9]	-	79.7	-	-	-	-	-	-
	RWalk-S[2]	-	82.5	-	-	-	-	-	-
	PI-S [34]	-	78.7	-	-	-	-	-	-
Generat. memory	EWC-M [28]	70.62	77.03	39.84	33.02	-	-	-	-
	DGR [30]	90.39	85.40	61.29	47.28	-	-	-	-
	MeRGAN [31]	98.19	97.00	80.90	66.78	-	-	-	-
	DGMw (ours)	98.75	96.46	**83.93**	**74.38**	**72.45**	**56.21**	**32.14**	17.82
	DGMa (ours)	**99.17**	**97.92**	81.07	66.89	71.91	51.75	25.93	15.16

- Here A_n is the performance on n previously seen classes
- For ImageNet-50 they train for 5 tasks, 10 classes per task
- They do not state it clearly, but as far as I got they use 5 and 10 tasks for other datasets

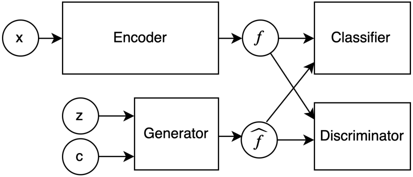
Latent Generative Memory

Main idea (1/2)

- Let's get rid of separate knowledge distillation step (it is questionable both biologically and practically)
- So let's train GM and Classifier jointly
- Since training GM in the visual space is tough, let's train it in the feature space
- Make the GM reside in "deep" layers of the Classifier and hallucinate

Latent Generative Memory

Illustration (2/2)



- For task $t = 1$ we train the model normally
- For task $t > 1$ generate a lot of fake memories with G_{t-1} of previously seen classes
- Train Classifier to correctly distinguish these fake memories
- Question #1: how to avoid knowledge distillation for Generator?
- Question #2: what if Encoder will start changing the embedding manifold? Then our fake memories will not correspond to actual embeddings. Maybe we can introduce prototypes to resolve this?