Deep Generative view on continual learning Assignment

author

donatella.genovese@uniroma1.it

Date: 15/06/2024

`https://github.com/DonatellaGenovese/ContinualLearningAssignment`

Contents

1 Introduction

In this assignment, we introduce a continual learning model for the class incremental scenario [5], where we use a generative neural network (specifically a Variational Autoencoder (VAEs) [1]) to mitigate the problem of catastrophic forgetting [2], [9]. In the class incremental learning scenario, the model is progressively exposed to new classes, making it crucial to retain knowledge of earlier classes while learning new ones. By employing a VAE, we generate synthetic data representative of past classes, which is then used alongside new data to train the model.

2 Dataset and Model

2.1 Dataset

We implement our experiments on the MNIST dataset (http://yann.lecun.com/exdb/mnist/) in the Class incremental scenario [6], in particular the original MNIST dataset is split into five tasks, where each task was a two-way classification. The standard training/test-split is used, resulting in 60,000 training images and 10,000 test images.

2.2 Models

The basic architecture is formed by a classifier that is used to classify inputs of each task. We choose an MLP network with 2 hidden layers with size of 50 which provides enough capacity to learn essential features of MNIST while keeping the model relatively simple. Then we implement also a generator (a VAEs) that is used to generate examples of previous task and add them on the current one in order to avoid cathastrofic forgetting. The VAE is composed by an encoder and a decoder network with 2 fully connected hidden layers of 400 units, a stochastic latent variable layer of size 100 and a standard normal distribution as prior as implemented in [4] Except for the first task, both the classifier and the generator are trained using replay data. The replay data is generated by sampling inputs from a copy of the generator. These inputs are then labeled with the most likely class as predicted by a copy of the classifier. The versions of the generator and classifier used to produce the replay data are copies of both models after finishing training on the previous task. Following the approach detailed in [4], for both the classifier and the generator, the total loss is a weighted sum of the loss on the data from the current task and the loss on the replayed data:

$$\mathcal{L} = \frac{1}{N_{\text{task so far}}}\mathcal{L}_{\text{current}} + (1 - \frac{1}{N_{\text{task so far}}})\mathcal{L}_{\text{replay}}$$

For the MLP model both $\mathcal{L}_{\text{current}}$ and $\mathcal{L}_{\text{replay}}$ are the cross entropy loss, while for VAE, we implement the loss as prescribed in [1]. In each iteration, the number of replayed samples

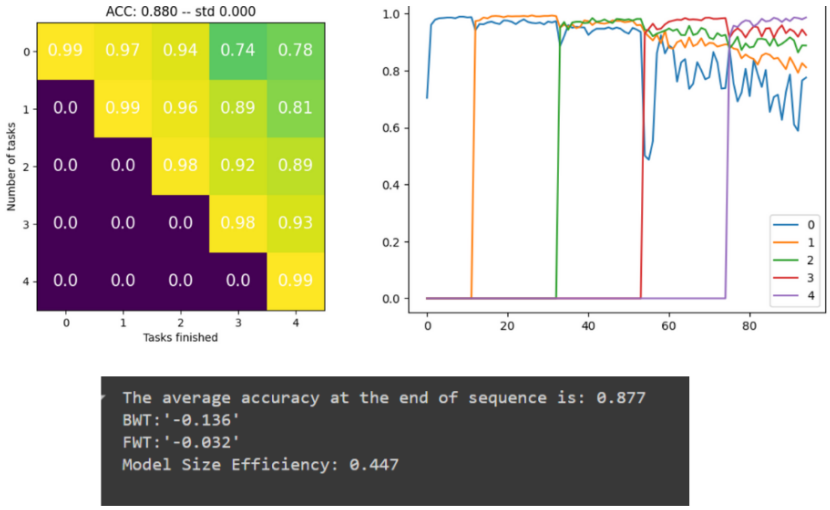

Figure 3.0.1: Results in term of accuracy, FT and BT and MS of a Generative Replay model

is equal to the number of samples from the current task. We train the MLP model over 5 epochs with a batch size of 128 and a learning rate of 1e-3 which is a standard choice for training a MNIST dataset. While the VAE model is trained on 50 epochs over a batch size of 128 as well. We also add a weight decay of 1e-5 and a dropout with probability of 0.1, since the model is not too complex and the dataset is not too large to require high values for regularization.

3 Results

In this section we show the results of the computation in terms of: accuracy, forward transfer (FT), backward transfer (BT) and model size efficiency (MS) [3]. As we can see in 3 we achieve good results in term of accuracy, FT and BT and our results are comparable with the ones obtained by a greedy buffer based GDUMB implementation[7], which are depicted in 3. While for the model size efficiency we do not have so high value which suggest that the model could be improved for achieve a better evaluation for this metric.

4 Conclusions

We note that one of the advantage of the implemented model is that, since at each iteration the number of the replayed samples is equal to the number of samples from the current task, the generation doesn't depend on the number of previous task and the computational costs per task doesn't increase with the number of tasks so far. This is an

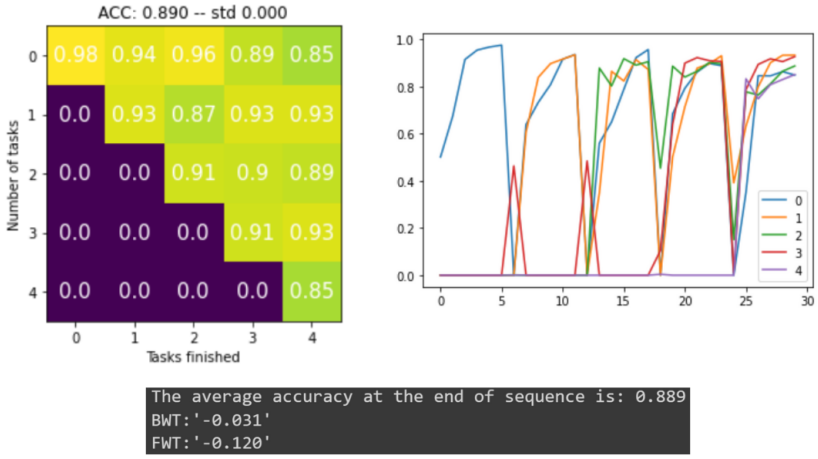

Figure 3.0.2: Results in term of accuracy, FT and BT of a Greedy Buffer

advantage wrt the full replay methods where some examples are stored in the memory and then add to the dataset of future class, while in our generative model we do not store sample but only the parameters of the VAE that are used to generate the sample for each task. Anyway we have also mentioned that when training a VAE alongside the classification model, we essentially have two distinct neural network models with their own sets of parameters that need to be optimized independently. This means that during each training iteration, we need to perform separate forward passes, compute gradients, and update parameters for both the VAE and the MLP model. To reduce the computational cost of generative replay it could be a good idea to integrate the generative model inside the main model (i.e. MLP model), in [4] this is done by equipping the main model with generative feedback connections.

Bibliography

[1] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2014. URL: http://arxiv.org/abs/1312.6114.

[2] Hanul Shin et al. "Continual Learning with Deep Generative Replay". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 2990–2999. URL: https://proceedings.neurips.cc/paper/2017/hash/0efbe98067c6c73dba1250d2beaa81f9-Abstract.html.

[3] Natalia Díaz Rodríguez et al. "Don't forget, there is more than forgetting: new metrics for Continual Learning". In: *CoRR* abs/1810.13166 (2018). arXiv: 1810.13166. URL: http://arxiv.org/abs/1810.13166.

[4] Michiel van der Ven and Andreas S. Tolias. "Generative replay with feedback connections as a general strategy for continual learning". In: *CoRR* abs/1809.10635 (2018). arXiv: 1809.10635. URL: http://arxiv.org/abs/1809.10635.

[5] Gido M. van de Ven and Andreas S. Tolias. "Three scenarios for continual learning". In: *CoRR* abs/1904.07734 (2019). arXiv: 1904.07734. URL: http://arxiv.org/abs/1904.07734.

[6] Gido M. van de Ven and Andreas S. Tolias. "Three scenarios for continual learning". In: *CoRR* abs/1904.07734 (2019). arXiv: 1904.07734. URL: http://arxiv.org/abs/1904.07734.

[7] Ameya Prabhu, Philip H. S. Torr, and Puneet K. Dokania. "GDumb: A Simple Approach that Questions Our Progress in Continual Learning". In: *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part II*. Ed. by Andrea Vedaldi et al. Vol. 12347. Lecture Notes in Computer Science. Springer, 2020, pp. 524–540. DOI: 10.1007/978-3-030-58536-5_31. URL: https://doi.org/10.1007/978-3-030-58536-5_31.

[8] Daniel Turner, Pedro J. S. Cardoso, and João M. F. Rodrigues. "Modular Dynamic Neural Network: A Continual Learning Architecture". In: *Applied Sciences* 11.24 (2021). ISSN: 2076-3417. DOI: 10.3390/app112412078. URL: https://www.mdpi.com/2076-3417/11/24/12078.

[9] Gido M. van de Ven, Zhe Li, and Andreas S. Tolias. "Class-Incremental Learning With Generative Classifiers". In: *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2021, virtual, June 19-25, 2021*. Computer Vision Foundation / IEEE, 2021, pp. 3611–3620. DOI: 10.1109/CVPRW53098.2021.00400. URL: https://openaccess.thecvf.com/content/CVPR2021W/CLVision/html/van_de_Ven_Class-Incremental_Learning_With_Generative_Classifiers_CVPRW_2021_paper.html.