Continual learning project

Andrea Maria Di Palma

June 2024

The ongoing project focuses on generative deep replay methods for continual learning, aiming to enhance the model's ability to learn and retain information over time.
A Generative Adversarial Network (GAN) has been implemented as the generative method, providing a robust framework for creating synthetic data to facilitate the continual learning process. This approach helps mitigate the problem of catastrophic forgetting by generating and replaying previous experiences during training.

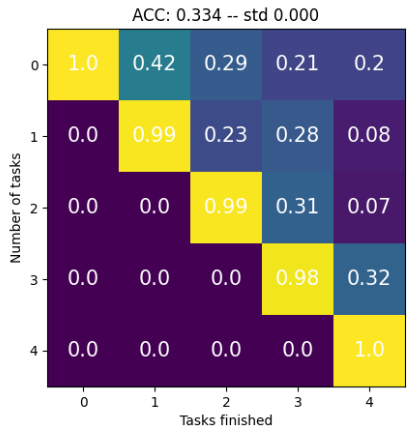
Results In order to maximise the reference metrics, the following choices of hyperparameters were made:

Parameter	**Value**
bs	128
epochs	8
GAN_epochs	10
num_tasks	5
dataset	MNIST
num_classes	10
in_size	28
n_channels	1
hidden_size	64
lr	2e-4
lr_cl	1e-3
latent_dim	128

Probably increasing the training epochs would have improved performance, but the low computing power available did not allow this.
The number of hyperparameter, in the case of GAN model, is $1.21 \cdot 10^6$, in the case of the only replay buffer methods is in the order of $5 \cdot 10^4$. Due to the fact that the generative method have to store a model for each iteration the computational cost is higher than vanilla replay methods.

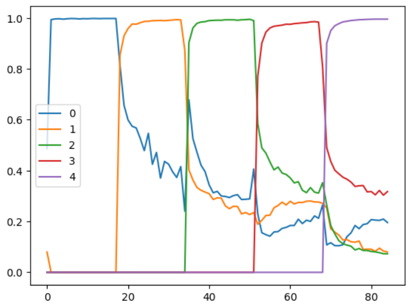
In the following table there are the metrics for the Generative methods:

Average accuracy	0.334
BWT	-0.826
FWT	-0.129



	0	1	2	3	4
0	1.0	0.42	0.29	0.21	0.2
1	0.0	0.99	0.23	0.28	0.08
2	0.0	0.0	0.99	0.31	0.07
3	0.0	0.0	0.0	0.98	0.32
4	0.0	0.0	0.0	0.0	1.0

(a) Confusion matrix



(b) Accuracy graph

Figure 1

Memory complexity The main reason for adopting generative methods for continual learning lies in the fact that a vanilla replay model, for each task must store data for previous iterations so complexity increases quadratically, in particular let d be the dimension of the subset for each task, n the number of tasks and c the dimension of the buffer of replay methods for each task:

$$d \times n + c \cdot \sum_{i=1}^{n} i = d \times n + \frac{n(n-1)}{2} \approx n^2 \tag{1}$$

Limitations The main limitations lies in the generative part, surely is not optimised for the specific task, maybe an approach using VAE could be more accurate because has less senstivity to hyperparameters and has more stability during train phase.