# DEEP GENERATIVE VIEW ON CONTINUAL LEARNING: COURSE ASSIGNMENT

**Vittoria Vineis**
vittoria.vineis@uniroma1.it

**Stefania Fiandrino**
stefania.fiandrino@uniroma1.it

## Introduction

This report showcases the key results obtained by implementing a generative replay method to address a class-incremental continual learning problem, using the MNIST dataset. For the generative method, we chose to use VAE[1] because we experienced a greater balance between computational efficiency and effective generative performance compared to GANs in our use case.

The key results and the values of evaluation metrics are reported in the next section, while in the last section comparisons with the vanilla replay method implemented in the third lab session and potential improvements are described.

## Results and discussion

Figure 1 showcases how the average accuracy varies according to different number of generative samples created (8,000 and 11,000), number of training epochs for the VAE (10 and 30) and number of epochs of the MLP (3 and 5). As shown, greater accuracy across tasks is achieved at higher sample size and higher number of epochs for the generative model, indicating that both the quality and quantity of the generated samples referring to the previous tasks influences the overall performance.
In Table 1, the effectiveness and efficiency metrics obtained with the most promising hyperparameter settings - that is learning rate equal to $1 \times 10^{-3}$ and batch size equal to 128 for both MLP and VAE; 11,000 generated samples at each incremental task; 30 and 5 training epochs respectively for VAE and MLP models - are reported.
Figure 2 and 3, instead, report the accuracy matrix and the average accuracy for each task throughout the incremental learning process. As can be noticed, despite not having significantly mitigated the problem of forgetting past tasks, the model showcases good performance in predicting the initial classes also at the latest stages of the continual learning process.

As verified, the generative replay method is more memory-efficient than vanilla replay method, because it avoid storing raw data samples from all past tasks, which can lead to impractically high memory usage as data volumes increase. In contrast, generative replay trains a model to recreate past data samples as needed. Given the proposed setup, increasing the number of classes/tasks make our solution scales almost linearly, given the fact that samples are not stored. On the other hand, given that samples are stored with the vanilla replay method, the latter showcases a greater progressive increase in computational requirements.

Regarding the downsides of our method, we were unable to completely solve the problem of forgetting previous classes. In general, when using generative replay methods, the accuracy across tasks heavily depends on the quality of the generated samples, and it can be challenging to achieve high-quality generative outputs for more complex inputs. Even when good generative performance is attainable, it can increase the model complexity, negatively impacting computational cost and processing time.

For further improvements, more computationally efficient architectures could be explored. Implementing knowledge distillation could be beneficial, allowing a smaller model to learn from a larger, more complex model, thereby

[1]The code for the VAE was adapted from the implementation available at https://github.com/lyeoni/pytorch-mnist-VAE/blob/master/pytorch-mnist-VAE.ipynb

maintaining performance while reducing computational costs. Quantization, which reduces the precision of the model's weights, and pruning, which removes redundant or less significant parameters, can also be effective in making the model more efficient. However, it is important to consider that each of these strategies may introduce trade-offs between performance and efficiency.

| Metric | Value |
|---|---|
| Average Accuracy | 0.753 |
| Forward Transfer | -0.119 |
| Backward Transfer | -0.286 |
| Computational Efficiency | 0.200 |

Table 1: Performance and efficiency metrics

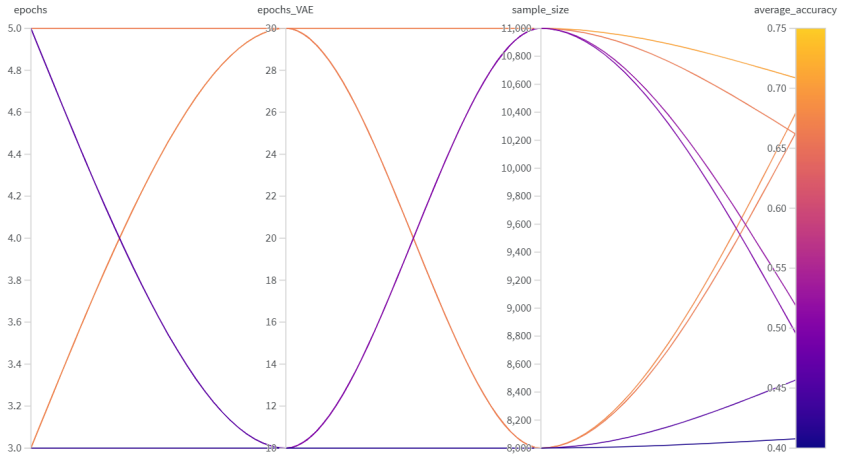


Figure 1: Sensitivity analysis

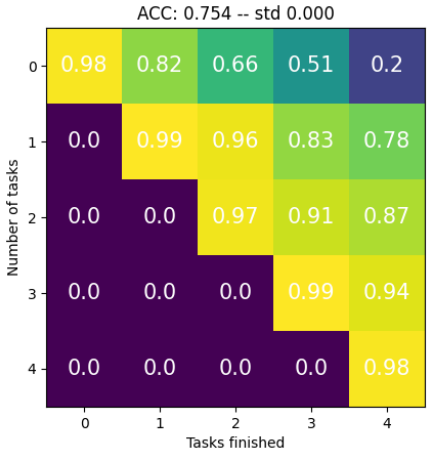


Figure 2: Accuracy matrix

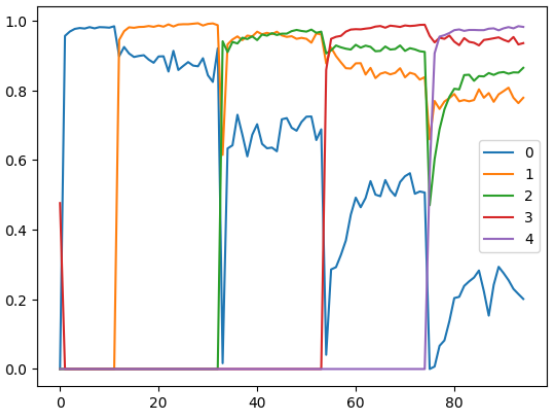


Figure 3: Accuracy over time