ESANN 2024 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 9-11 October 2024, i6doc.com publ., ISBN 978-2-87587-090-2. Available from http://www.i6doc.com/en/.

Continual Learning with Graph Reservoirs: Preliminary experiments in graph classification

Domenico Tortorella and Alessio Micheli

University of Pisa - Department of Computer Science
Largo B. Pontecorvo 3, 56127 Pisa - Italy

Abstract. Continual learning aims to address the challenge of catastrophic forgetting in training models where data patterns are non-stationary. Previous research has shown that fully-trained graph learning models are particularly affected by this issue. One approach to lifting part of the burden is to leverage the representations provided by a training-free reservoir computing model. In this work, we evaluate for the first time different continual learning strategies in conjunction with Graph Echo State Networks, which have already demonstrated their efficacy and efficiency in graph classification tasks.

1 Introduction

Many real-world scenarios involve both non-stationary input data and learning tasks that change through time. This situation poses a challenge to traditional learning models developed under the assumptions of fixed data distribution and fixed target task. Continual learning (CL) is concerned with the design of models and techniques able to overcome the catastrophic forgetting of past knowledge while learning new information [1]. So far CL research has mainly focused on flat data, images, and more recently sequences [2]. Graphs provide a useful structure to represent relations between entities, such as paper citations or web page networks. A plethora of neural models have been proposed to solve graph-, edge-, and node-level tasks [3], most of them sharing an architecture structured in layers that perform local aggregations of node features. Research has shown that graph learning models particularly suffer from catastrophic forgetting [4], which may also be due to the interplay with inherent issues of learning on graphs [5]. Graph Echo State Networks (GESN) [6] are an efficient model within the Reservoir Computing (RC) paradigm. In RC, input data is encoded via a randomly-initialized reservoir, while only a readout classifier for the downstream task requires training. GESN has already been successfully applied to graph-level classification tasks [7]. The possibility of exploiting an RC model to provide data representations that do not require training has been already considered for sequence data [8]. In this paper, we evaluate for the first time GESN in conjunction with different CL strategies as a method for addressing the challenge of catastrophic forgetting. In this preliminary work, we focus our benchmarks on the class-incremental scenario for graph classification on social network data.

Research partly supported by PNRR, PE00000013 - “FAIR - Future Artificial Intelligence Research” - Spoke 1, funded by European Commission under the NextGeneration EU programme.

2 Reservoir computing for graphs

Reservoir computing is a paradigm for the efficient design of recurrent neural networks. Input data is encoded by a randomly initialized reservoir, while only the task prediction layer requires training. Graph Echo State Networks (GESNs) extended the reservoir computing paradigm to graph-structured data [6], and have already demonstrated their effectiveness in graph classification tasks [7].

Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be a graph with nodes $\mathcal{V}$ and edges $\mathcal{E}$. We denote by $\mathcal{N}(v)$ the neighborhood of node v, and by $\mathbf{A}$ the graph adjacency matrix. Each node may have associated input features $\mathbf{x}_v \in \mathbb{R}^X$. Node embeddings are recursively computed by the dynamical system

$$\mathbf{h}_v^{(k)} = \tanh\left(\mathbf{W}_{\text{in}}\, \mathbf{x}_v + \textstyle\sum_{v' \in \mathcal{N}(v)} \hat{\mathbf{W}}\, \mathbf{h}_{v'}^{(k-1)}\right), \quad \mathbf{h}_v^{(0)} = \mathbf{0}, \tag{1}$$

where $\mathbf{W}_{\text{in}} \in \mathbb{R}^{H \times X}$ and $\hat{\mathbf{W}} \in \mathbb{R}^{H \times H}$ are the input-to-reservoir and the recurrent weights, respectively (input bias is omitted). Equation (1) is iterated over k until the system state converges to fixed point $\mathbf{h}_v^{(\infty)}$, which is used as the embedding.

The existence of a fixed point is guaranteed by the Graph Embedding Stability (GES) property [7], which also guarantees independence from the system's initial state $\mathbf{h}_v^{(0)}$. A necessary condition [9] for the GES property is $\rho(\hat{\mathbf{W}}) < 1/\rho(\mathbf{A})$, where $\rho(\mathbf{A})$ denotes the graph spectral radius, i.e. the largest absolute eigenvalue of its adjacency matrix $\mathbf{A}$. This condition also provides the best estimate of the system bifurcation point, i.e. the threshold beyond which (1) becomes asymptotically unstable. Reservoir weights are randomly initialized from a uniform distribution in $[-1, 1]$, and then rescaled to the desired input scaling and reservoir spectral radius, without requiring any training.

In graph-level tasks node features are aggregated to provide global embeddings by a permutation-invariant pooling function such as $\mathbf{h}_{\mathcal{G}} = \sum_{v \in \mathcal{V}} \mathbf{h}_v^{(\infty)}$. This representation is then used to train a classifier for the downstream task. In reservoir computing this usually consists in a linear readout $\mathbf{y}_{\mathcal{G}} = \mathbf{W}_{\text{out}}\, \mathbf{h}_{\mathcal{G}} + \mathbf{b}_{\text{out}}$, where the weights $\mathbf{W}_{\text{out}} \in \mathbb{R}^{C \times H}$, $\mathbf{b}_{\text{out}} \in \mathbb{R}^C$ are trained by ridge regression on one-hot encodings of target class $y_{\mathcal{G}} \in 1, ..., C$.

3 Continual learning

In a general continual learning setting the models receive a stream of experiences $\mathfrak{e}_1, \mathfrak{e}_2, ...$ where each experience contains data sampled from its own distribution. Between experiences there may happen a shift in probability distributions, or even new learning tasks may present themselves. When trained sequentially on multiple experiences, neural networks tend to forget previous knowledge, catastrophically reducing their performance on previously seen data and tasks.

In this work, we focus on the *class-incremental* continual learning setting. Here each experience coincides with a classification task $\mathcal{T}_i$ on newly introduced classes. The performance of CL methods is evaluated with the average accuracy metric

$$\textsc{Acc}_T = \frac{1}{T}\sum_{i=1}^{T} \textsc{Acc}_{T,i} \tag{2}$$

where $\textsc{Acc}_{T,i}$ is the accuracy on task $\mathcal{T}_i$ after having experienced all tasks $\mathcal{T}_1, ..., \mathcal{T}_T$.

In our experiments, we evaluate the following continual learning strategies, which are agnostic with respect to the type of input data:

Naive The simplest (and least effective) strategy is just incrementally fine-tuning a single model without employing any method to contrast the catastrophic forgetting of previous knowledge.

Replay A memory buffer holds M patterns randomly sampled in previous experiences, which are added to the training set in the current experience to be 'replayed' [10]. This is one of the most effective strategies for fully-trained GNNs [4].

LwF Learning without Forgetting [11] uses distillation to regularize the current loss with soft targets taken from a previous version of the model on current experience patterns.

EWC Elastic weight consolidation [12] adds a penalty to the loss function in order to prevent large changes in model parameters deemed important in previous experiences. Parameter importance is computed as the average gradient norm on all experience patterns after training is complete, as proxy to Fisher information.

SLDA Streaming LDA [13] performs the incremental training of a linear readout layer by computing the online linear discriminant analysis on representations produced by a fixed input data encoder. As the LDA is computed exactly and efficiently by updating first- and second-order statistics with a single pass on input data, this CL strategy is the best-suited to leverage the embeddings efficiently computed by a training-free reservoir [8].

4 Experiments and discussion

We adapt two graph classification benchmarks from the widely-adopted collection [14] as class-incremental CL scenarios. Graph data represents online discussions between users in different sub-reddits, which correspond to the class to predict. Reddit-5K consists in 4,999 samples over 5 classes, while Reddit-12K has 11,929 samples over 11 classes. The first classification task $\mathcal{T}_1$ has 3 classes in each CL version of the benchmarks, while the remained classes are partitioned in pairs as the remaining tasks. We partition the datasets in 10-fold selection/test streams, holding out a fraction of each selection fold as validation stream, for a training/validation/test ratio of 80:10:10. We generically follow the experimental setup of [8]. The number of reservoir units for GESN is fixed at $H = 256$,

	Reddit-5K		**Reddit-12K**	
	GESN+Linear	GESN+MLP	GESN+Linear	GESN+MLP
Joint	48.23 ± 2.86	51.75 ± 2.69	35.35 ± 1.94	40.85 ± 2.55
Naive	38.95 ± 1.90	21.00 ± 1.79	13.38 ± 1.69	12.43 ± 4.61
Replay	41.51 ± 1.71	24.02 ± 7.48	17.87 ± 3.59	19.15 ± 4.54
LwF	41.47 ± 1.08	22.79 ± 1.97	19.12 ± 2.93	21.61 ± 0.23
EWC	41.47 ± 1.55	25.36 ± 6.60	18.39 ± 3.32	16.49 ± 5.67
SLDA	43.65 ± 4.07	N/A	30.43 ± 5.28	N/A
Baseline	20.00 ± 0.01		21.73 ± 0.03	

Table 1: Final classification accuracy, average and deviation over 10 folds. (SLDA requires a linear readout, and thus cannot be applied to an MLP.)

while the hyper-parameters of each CL strategy (e.g. the memory buffer size M in the replay strategy) are selected by grid search according to the accuracy on the validation stream. As the classifier readout we adopt both a linear layer and an MLP with one hidden layer of 256 units and ReLU activation function. Both are trained with cross-entropy as classification loss via Adam optimizer, with learning rate and weight decay as part of the grid search parameters. The code for our experiments exploits the PyTorch implementation of the CL framework of the Avalanche library [15].

In Table 1 we report the average task accuracy metric Acc_T after the continual learning model has experienced all tasks, averaged over the 10 dataset folds and 3 reservoir initializations. We additionally report the join training on all experiences and the majority-class predictor baseline as references. We notice that in the joint setting the use of an MLP as readout boosts the classification accuracy by 3.5–5.5 points. However, in the class-incremental setting having the additional learnable parameters of the hidden layer accentuates the issue of catastrophic forgetting. The accuracy of CL strategies on the linear readout are generally above the accuracy with an MLP classifier. The best-performing CL strategy is SLDA, which by its nature requires a linear classifier and thus cannot be applied to an MLP, while we do not notice a consistent ranking in the remaining strategies that perform significantly worse. We notice a particular effect of catastrophic forgetting in Reddit-12K: in many cases the final accuracy is significantly lower than the majority-class baseline, a sign that knowledge on this class was present in past tasks and has been forgotten at the end of the experience stream. In Fig. 1 we report the progress of average task accuracy on seen tasks as new experiences are presented to the CL model. By examining the different trends we observe the trade-off between accuracy and resilience to catastrophic forgetting in SLDA. Both in Reddit-5K and Reddit-12K an LDA classifier performs worse in the first task $\mathcal{T}_1$. However, as the number of experiences progresses, SLDA results more resilient to catastrophic forgetting than other CL strategies, consistently performing better than the other strategies af-

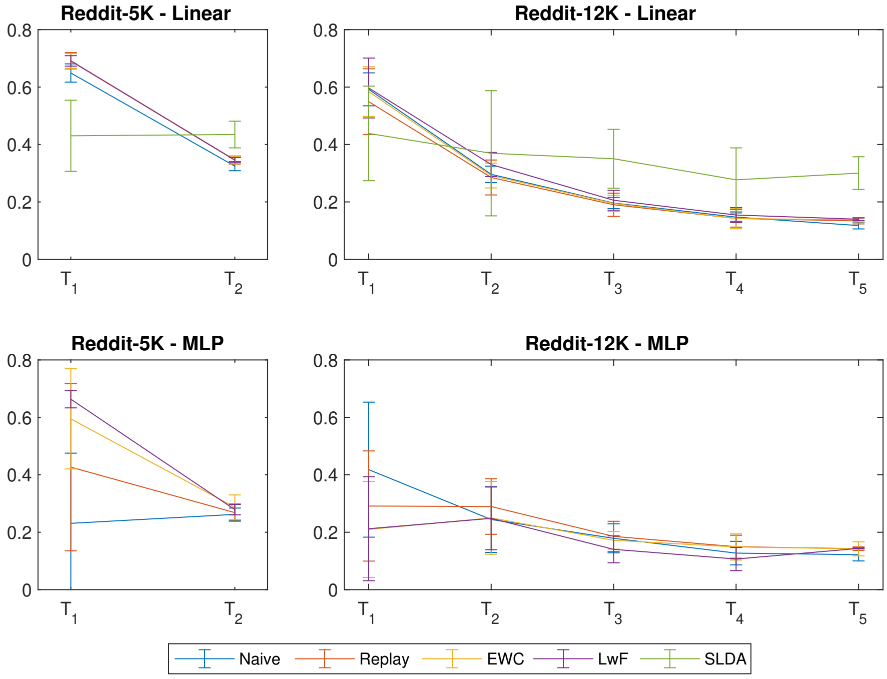

Fig. 1: Progress of average task accuracy and standard deviation on seen tasks.

ter subsequent tasks are presented to the model. Incidentally, we notice also a large dispersion of first experience accuracy, as hyper-parameter configurations are selected according to the overall performance after the final experience.

5 Conclusion and future directions

In this work we have provided the first experimental evaluation of GESNs with readout classifiers trained with popular continual learning methods in class-incremental scenarios. Our experiments have shown a trade-off between accuracy and resilience to catastrophic forgetting, with SLDA performing significantly better than other CL strategies. As this continual learning strategy relies upon a fixed input data encoder, it cannot be properly exploited by end-to-end trained graph learning models. Graph reservoirs on the other hand are able to efficiently provide effective graph representations, being best suited to be applied jointly with SLDA. The combination of GESN and SLDA thus can provide a CL method for graphs that is both more effective and more efficient than the fully-trained counterparts, avoiding the catastrophic forgetting that is typical in this context [4]. Building on these results, in future works we will exploit streaming approaches to ridge regression [16] to improve upon SLDA and to extend its approach also to regression tasks. We will also explore different graph learning settings, such as node classification in a single graph with evolving topology or learning dynamical graphs [17], with the resulting challenges for reservoir design.

References

[1] L. Wang, X. Zhang, H. Su, and J. Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(8):1–20, 2024.

[2] A. Cossu, A. Carta, V. Lomonaco, and D. Bacciu. Continual learning for recurrent neural networks: An empirical evaluation. *Neural Networks*, 143:607–627, 2021.

[3] D. Bacciu, F. Errica, A. Micheli, and M. Podda. A gentle introduction to deep learning for graphs. *Neural Networks*, 129:203–221, 2020.

[4] A. Carta, A. Cossu, F. Errica, and D. Bacciu. Catastrophic forgetting in deep graph networks: A graph classification benchmark. *Frontiers in Artificial Intelligence*, 5(824655):1–7, 2022.

[5] F. G. Febrinanto, F. Xia, K. Moore, C. Thapa, and C. Aggarwal. Graph lifelong learning: A survey. *IEEE Computational Intelligence Magazine*, 18(1):32–51, feb 2023.

[6] C. Gallicchio and A. Micheli. Graph echo state networks. In *The 2010 International Joint Conference on Neural Networks*, pages 3967–3974, 2010.

[7] C. Gallicchio and A. Micheli. Fast and deep graph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):3898–3905, 2020.

[8] A. Cossu, D. Bacciu, A. Carta, C. Gallicchio, and V. Lomonaco. Continual learning with echo state networks. In *ESANN 2021 Proceedings*, pages 275–280, 2021.

[9] D. Tortorella, C. Gallicchio, and A. Micheli. Spectral bounds for graph echo state network stability. In *The 2022 International Joint Conference on Neural Networks*, 2022.

[10] A. Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

[11] Z. Li and D. Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018.

[12] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences of the United States of America*, 114(13):3521–3526, 2017.

[13] T. L. Hayes and C. Kanan. Lifelong machine learning with deep streaming linear discriminant analysis. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 887–896, 2020.

[14] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann. TUDataset: A collection of benchmark datasets for learning with graphs. In *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020)*, 2020.

[15] A. Carta, L. Pellegrini, A. Cossu, H. Hemati, and V. Lomonaco. Avalanche: A PyTorch library for deep continual learning. *arXiv*, 2302.01766, 2023.

[16] T. Zhang and B. Yang. An exact approach to ridge regression for big data. *Computational Statistics*, 32(3):909–928, 2017.

[17] A. Micheli and D. Tortorella. Discrete-time dynamic graph echo state networks. *Neurocomputing*, 496:85–95, 2022.