Advances in Continual Graph Learning for Anti-Money Laundering Systems

Bruno Deprez[1,2][0000−0002−0817−0501], Wei Wei[2][0000−0001−5651−8712], Wouter Verbeke[1][0000−0002−8438−0535], Bart Baesens[1,3][0000−0002−5831−5668], Kevin Mets[2][0000−0002−4812−4841], and Tim Verdonck[2,1][0000−0003−1105−2028]

[1] KU Leuven, 3000 Leuven, Belgium
{bruno.deprez,wouter.verbeke,bart.baesens}@kuleuven.be
[2] University of Antwerp - imec, 2020 Antwerpen, Belgium
{wei.wei,kevin.mets,tim.verdonck}@uantwerpen.be
[3] University of Southampton, Southampton SO17 1BJ, UK

Keywords: Continual graph Learning · Anti-Money Laundering

1 Introduction

Criminal activities generate income that needs to be *laundered* to make illicit money appear legitimate. The laundered money is used to expand criminal activity and finance terrorism (Levi and Reuter, 2006). Therefore, regulators have put stringent anti-money laundering (AML) requirements on financial institutions to mitigate money laundering (ML) risk.

Financial institutions face enormous volumes of financial transactions that require continuous monitoring, so detection models need regular fine-tuning. This fine-tuning is constrained since (1) there is only limited computational budget, (2) regulations prohibit retaining client data indefinitely and (3) data drift due to changing ML tactics leads to catastrophic forgetting (French, 1999; De Lange et al., 2022).

Continual learning provides a potential solution as it (1) limits computation time by fine-tuning existing models, (2) only requires the most recent data, and (3) preserves previously learned knowledge (Goodfellow et al., 2015).

Despite these benefits, research on continual graph learning for AML is rare. Furthermore, benchmarks lack analyses on the underlying assumptions of the applied methods. Therefore, this paper evaluates the use of continual learning for AML and the impact of hyperparameter settings on the performance of state-of-the-art continual learning methods.

2 Preliminaries

2.1 Deep Learning on Graphs

A graph $G(V, E)$ is defined using V and E, which represent the nodes and edges, respectively. Each node is also assigned a feature vector $x_i \in \mathcal{R}^m$.

Deep learning on graphs is often conducted using graph neural networks (GNNs) which construct the representation of node i at layer l, $h_i^{(l)}$, as $h_i^{(l)} = \phi^{(l-1)}\left(h_i^{(l-1)}, \sum_j \hat{A}_{ij}\psi^{(l-1)}\left(h_i^{(l-1)}, h_j^{(l-1)}\right)\right)$, with $\phi^{(l)}$ and $\psi^{(l)}$ layer-dependent functions, and $\hat{A}_{ij}$ the normalized adjacency matrix. The initial embedding is set equal to the node features, $h_i^{(0)} = x_i$.

2.2 Continual (Graph) Learning

In continual learning (CL), a model needs to sequentially learn disjoint tasks $\mathcal{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_K\}$ (Kirkpatrick et al., 2017; De Lange et al., 2022), where access to data of previous tasks is limited.

Three CL settings are discerned depending on the available information of the tasks (van de Ven et al., 2022), i.e., task-incremental (distinct tasks with task labels provided), class-incremental (growing number of classes without task labels provided) and domain-incremental learning (same problem, e.g., fraud detection, but the distribution of tasks shift).

CL methods can be broadly classified into three categories, i.e., replay (preserving some historical observations), regularisation-based (limiting the updating of important weights) and architecture-based (reserve specific weights for specific tasks) (De Lange et al., 2022).

For network data, one of the following two categories of node interactions across tasks can be present. First, separate networks are considered per task with no links to nodes in previous tasks. Second, the network grows across tasks, meaning that some nodes in the new task have connections to nodes of previous tasks.

3 Experimental Setup

The experiments are implemented in the BeGin framework (Ko et al., 2024), with code available on github[4]. The full pipeline is visualised in Figure 1.

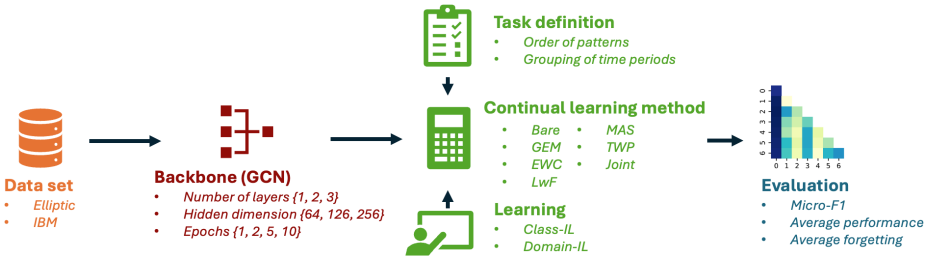


Fig. 1: Pipeline of experiments, with values of the methods and hyperparameters.

[4] https://github.com/VerbekeLab

We use two open-source ML datasets, the IBM AML (Altman et al., 2023) and the Elliptic dataset (Elliptic; Weber et al., 2019). The IBM dataset is a synthetic dataset consisting of 515,080 nodes—entities—and 5,078,345 edges—transactions. Transactions are labelled according to eight ML patterns, each consisting of only a couple hundred transactions. The Elliptic dataset contains real-world Bitcoin transactions, grouped in 49 time intervals. The network consists of 20,3769 nodes—transactions—and 234,355 edges—money flow—having just 4,545 illicit transactions.

The backbone GCN (Kipf and Welling, 2017) is combined with state-of-the-art CL methods covering replay (GEM (Lopez-Paz and Ranzato, 2017)), regularisation-based (GEM (Lopez-Paz and Ranzato, 2017), EWC (Kirkpatrick et al., 2017), MAS (Aljundi et al., 2018), TWP (Liu et al., 2021)) and architecture-based (LwF (Li and Hoiem, 2018)) methods. We compare the CL methods with the *bare* model—naive fine-tuning on only the current tasks—and the *joint* model—training on all data from the current and previous tasks. Most of these methods are general purpose CL methods, while TWP (Liu et al., 2021) is designed for network data.

Class-incremental learning is applied on the IBM dataset, where patterns are learned one-by-one using multi-class classification. For the Elliptic dataset, we apply domain-incremental learning, since it presents a binary classification problem with distribution shift over time.

The performance is measured using the micro-F1 metric, and the retention of previous knowledge is measured using average performance (AP) and average forgetting (AF). The latter two are based on the performance matrix M, with entries

$$M_{i,j} = \begin{cases} \text{Performance on task } i \text{ after training on task } j & \text{if } i \leq j \\ 0 & \text{otherwise} \end{cases},$$

and are defined as $\text{AP} = \sum_{i=1}^{k} \frac{M_{k,i}}{k}$ and $\text{AF} = \sum_{i=1}^{k-1} \frac{M_{i,i} - M_{k,i}}{k-1}$.

4 Results and Discussion

We discuss the results using the IBM AML dataset. Detailed discussions on the Elliptic dataset can be found in the full version of our work (Deprez et al., 2025).

Figure 2 contains the evolution of the AF during training and illustrates that forgetting stays low for a few epochs, but shoots up suddenly, resulting in catastrophic forgetting. Only GEM seems to be able to keep forgetting low, where it even improves when tasks are added. In general, however, special care is needed when selecting the number of epochs in our case.

A detailed view on the effect of the hyperparameter choices is shown in Figure 3. The results are presented according to the depth and width of the model. We see strong forgetting across experiments, which is probably caused by the strong similarity among the tasks (Wang et al., 2024).

We notice that more epochs lead to more forgetting, while there is a limit on the performance the model can obtain. Next, higher dimensions lead to an

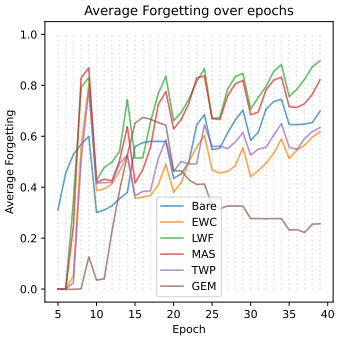


Fig. 2: AF when using five epochs per task.

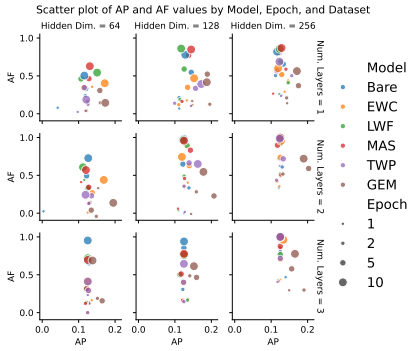


Fig. 3: AF (lower is better) against AP (higher is better) across the hyperparameter values.

increase in performance, but also correlate with stronger forgetting. Forgetting seems to be worse for wider models, probably because of overfitting on the current task.

We also note that the Bare model suffers strong forgetting, especially for the GCNs with three layers, again because of overfitting on the current task. For the other layers, Bare, MAS and LwF seem to consistently suffer more forgetting for similar performance compared to the other methods. From these results, we see that GEM (replay) performs best in terms of performance and forgetting. TWP is also able to keep forgetting low by incorporating network information.

5 Conclusion

This work extends the application of continual graph learning to anti-money laundering, with the aim to mitigate current business constraints. It provides an overview of the main CL and AML literature and complements this with a comprehensive experimental evaluation.

The results illustrate the impact of hyperparameter choices. Wider models as well as models trained with too many epochs are more prone to forgetting because of overfitting on the current task. We also show that replay methods and methods incorporating network topology are best at keeping forgetting low.

Acknowledgments. This work was supported by the Research Foundation – Flanders (FWO research project 1SHEN24N), by the BNP Paribas Fortis Chair in Fraud Analytics, and by the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme. The resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation - Flanders (FWO) and the Flemish Government.

Bibliography

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

Erik Altman, Jovan Blanuša, Luc von Niederhäusern, Beni Egressy, Andreea Anghel, and Kubilay Atasu. Realistic synthetic financial transactions for anti-money laundering models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 29851–29874. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/5f38404edff6f3f642d6fa5892479c42-Paper-Datasets_and_Benchmarks.pdf.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3366–3385, 2022. https://doi.org/10.1109/TPAMI.2021.3057446.

Bruno Deprez, Wei Wei, Wouter Verbeke, Bart Baesens, Kevin Mets, and Tim Verdonck. Advances in continual graph learning for anti-money laundering systems: A comprehensive review. *WIREs Computational Statistics*, 17(3):e70040, 2025. https://doi.org/https://doi.org/10.1002/wics.70040. URL https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.70040. e70040 EOCS-778.R1.

Elliptic. Elliptic. www.elliptic.co. Accessed: 2024-01-31.

Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135, 2025/02/25 1999. https://doi.org/10.1016/S1364-6613(99)01294-2. URL https://doi.org/10.1016/S1364-6613(99)01294-2.

Ian J. Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks, 2015. URL https://arxiv.org/abs/1312.6211.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13): 3521–3526, 2017. https://doi.org/10.1073/pnas.1611835114. URL https://www.pnas.org/doi/abs/10.1073/pnas.1611835114.

Jihoon Ko, Shinhwan Kang, Taehyung Kwon, Heechan Moon, and Kijung Shin. Begin: Extensive benchmark scenarios and an easy-to-use framework for graph continual learning, 2024. URL https://arxiv.org/abs/2211.14568.

Michael Levi and Peter Reuter. Money laundering. *Crime and justice*, 34(1): 289–375, 2006. https://doi.org/10.1086/501508.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018. https://doi.org/10.1109/TPAMI.2017.2773081.

Huihui Liu, Yiding Yang, and Xinchao Wang. Overcoming catastrophic forgetting in graph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(10):8653–8661, May 2021. https://doi.org/10.1609/aaai.v35i10.17049. URL https://ojs.aaai.org/index.php/AAAI/article/view/17049.

David Lopez-Paz and Marc' Aurelio Ranzato. Gradient episodic memory for continual learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/f87522788a2be2d171666752f97ddebb-Paper.pdf.

Gido M. van de Ven, Tinne Tuytelaars, and Andreas S. Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022. https://doi.org/10.1038/s42256-022-00568-3. URL https://doi.org/10.1038/s42256-022-00568-3.

Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(8):5362–5383, 2024. https://doi.org/10.1109/TPAMI.2024.3367329.

Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel Karl I. Weidele, Claudio Bellei, Tom Robinson, and Charles E. Leiserson. Anti-money laundering in bitcoin: Experimenting with graph convolutional networks for financial forensics, 2019. URL https://arxiv.org/abs/1908.02591.