Supplementary Material for 'Benchmarking Continual Learning on Netlists with Circuit-Targeted Graph Neural Networks'

Rupesh Raj Karn, Johann Knechtel, Ozgur Sinanoglu
Center for Cyber Security, New York University, Abu Dhabi, UAE.
Email: {rupesh.k, johann, ozgursin}@nyu.edu

I. Graph Classification for Circuit Functional Analysis

In graph classification, each netlist is treated as a single graph with a label indicating circuit function or type (e.g., multiplier, adder, sine). This is essential for hierarchical EDA tasks and functional verification. Given a graph $G = (V, E, X)$, we compute:

$$\mathbf{h}_G = \text{READOUT}\left(\left\{\mathbf{h}_v^{(L)} \mid v \in V\right\}\right) \quad (1)$$

$$\hat{y}_G = \text{softmax}(W \cdot \mathbf{h}_G + b), \quad (2)$$

where `READOUT` is typically a mean or sum pooling operation.

A. Graph Classification: Accuracy Across All CL Tasks

Due to the uniqueness of each circuit function in the ISCAS85 and EPFL datasets—where each graph represents a distinct circuit appearing only once—we cannot directly apply a standard multi-class graph classification approach. Instead, we design a set of four binary classification tasks based on structural properties of the graphs, specifically their node counts. To do this, we compute the 25th, 50th, and 75th percentiles of graph sizes (in terms of node count) across the entire dataset and assign each graph to one of four size-based quartiles: $q \in \{0, 1, 2, 3\}$.

Each task is framed as a binary classification problem where the model must determine whether a given graph belongs to a specific quartile (positive class) or to any of the remaining three quartiles (negative class). For example, graphs in quartile 0 (i.e., the smallest 25% of circuits) are labeled positive, while all others are labeled negative. This formulation enables us to evaluate how well the model retains knowledge of circuits with distinct size and topological characteristics across tasks. The tasks are defined as follows:

- ▷ *Task 1:* Classify graphs in *quartile 0* (smallest graphs) versus all others (quartiles 1–3).
- ▷ *Task 2:* Classify graphs in *quartile 1* versus all others (quartiles 0, 2, 3).
- ▷ *Task 3:* Classify graphs in *quartile 2* versus all others (quartiles 0, 1, 3).
- ▷ *Task 4:* Classify graphs in *quartile 3* (largest graphs) versus all others (quartiles 0–2).

TABLE I: Accuracies (%) per stage and task for Graph Classification.

Method	Stage	Graph Classification			
		Task$_1$	Task$_2$	Task$_3$	Task$_4$
EWC	1	83.33	–	–	–
	2	66.67	80.00	–	–
	3	66.67	80.00	66.67	–
	4	66.67	80.00	66.67	66.67
SI	1	83.33	–	–	–
	2	83.33	80.00	–	–
	3	66.67	80.00	66.67	–
	4	66.67	80.00	66.67	66.67
iCaRL	1	100.00	–	–	–
	2	66.67	80.00	–	–
	3	66.67	80.00	66.67	–
	4	66.67	80.00	66.67	66.67
A-GEM	1	33.33	–	–	–
	2	16.67	20.00	–	–
	3	0.00	20.00	33.33	–
	4	16.67	20.00	50.00	50.00
MER	1	50.00	–	–	–
	2	83.33	80.00	–	–
	3	100.00	80.00	66.67	–
	4	66.67	80.00	66.67	66.67
CoPE	1	66.67	–	–	–
	2	33.33	60.00	–	–
	3	66.67	60.00	66.67	–
	4	33.33	40.00	50.00	83.33

B. Tasks Accuracy Analysis

a) Best-Performing Methods and Their Advantages: Among the methods evaluated from Table I, MER and CoPE consistently perform better across multiple tasks. CoPE excels in graph classification—reaching up to 83.33% on Task$_4$—and shows strong task recall, particularly in structurally complex gates like `xor`/`xnor`. MER also demonstrates competitive performance, indicating effective memory replay and adaptability. These results suggest that hybrid approaches, which incorporate both memory and gradient alignment strategies, are well-suited for dynamic circuit environments.

b) Catastrophic Forgetting in Continual Learning: Continual learning systems often struggle with catastrophic forgetting, where learning new tasks degrades performance on previously seen tasks. This effect is evident in graph classification as well, though the degradation is less severe compared to finer-grained tasks. Some methods still show difficulty in retaining prior knowledge while adapting to new distributions, particularly when tasks involve structurally dissimilar gate types.

c) Quantifying Performance Drop Across Tasks: Sequential task learning results in performance degradation across earlier graph classification tasks, though the drop is less dramatic than in other problem types. For example, replay-based methods such as MER manage to sustain accuracy across

TABLE II: Drift vs forgetting across Graph classification tasks. The *from→to* column denotes the pair of training stages being compared: "$k \to t$" indicates that we compute drift and forgetting by comparing the model immediately after learning task T_k to its state after learning task T_t. In other words, it quantifies how parameter updates during tasks $k+1,\dots,t$ impact performance on the original task T_k.

Method	from→to	Graph Classification		
		$\|\Delta\theta\|$	drift	ΔAcc
EWC	1→2	0.1483	0.000006	+0.1667
	1→3	0.3026	0.000017	+0.1667
	2→3	0.1602	0.000001	0.0000
	1→4	0.4573	0.000025	+0.1667
	2→4	0.3197	0.000004	0.0000
	3→4	0.1614	0.000001	0.0000
SI	1→2	0.1503	3.56e-07	0.0000
	1→3	0.3031	1.57e-06	+0.1667
	2→3	0.1591	3.25e-06	0.0000
	1→4	0.4485	3.63e-06	+0.1667
	2→4	0.3098	1.20e-05	0.0000
	3→4	0.1540	4.80e-06	0.0000
iCaRL	1→2	1.1987	0.0966	+0.3333
	1→3	1.7311	35.1136	+0.3333
	2→3	0.9499	0.9218	0.0000
	1→4	1.8992	55.7087	+0.3333
	2→4	1.3804	5.5310	0.0000
	3→4	0.7888	1.8808	0.0000
A-GEM	1→2	0.1539	+0.8325	+0.1667
	1→3	0.3107	+0.7259	+0.3333
	2→3	0.1622	+0.9484	0.0000
	1→4	0.4633	+0.7090	+0.1667
	2→4	0.3197	+0.9388	0.0000
	3→4	0.1601	+0.9889	–0.1667
MER	1→2	0.0150	(meta)	–0.3333
	1→3	0.0470	(meta)	–0.5000
	2→3	0.0374	(meta)	0.0000
	1→4	0.0772	(meta)	–0.1667
	2→4	0.0682	(meta)	0.0000
	3→4	0.0309	(meta)	0.0000
CoPE	1→2	5.2621	0.50068	+0.33333
	1→3	6.7898	0.70651	0.00000
	2→3	5.9258	0.49176	0.00000
	1→4	7.3818	0.80119	+0.33333
	2→4	6.9671	0.62150	+0.20000
	3→4	4.3920	0.30842	+0.16667

Correlations:

Method	Graph Classification	
	corr($\|\Delta\theta\|$, ΔAcc)	corr(drift,ΔAcc)
EWC	+0.3941	+0.7727
SI	+0.7793	–0.3155
iCaRL	+0.7193	+0.6453
A-GEM	+0.4475	–0.9007
MER	+0.1849	–
CoPE	+0.0191	+0.1441

multiple stages, while regularization-based methods like EWC and SI show more noticeable declines.

d) Stability–Plasticity Trade-offs: Not all methods balance old and new knowledge equally. In graph classification, A-GEM virtually eliminates forgetting yet achieves limited forward transfer, often trailing replay methods on later tasks. In contrast, MER and CoPE show stronger adaptability, combining stability with the ability to learn new tasks effectively.

e) Trade-offs Between Retention and Transfer: Our evaluation reveals fundamental trade-offs between knowledge retention and forward transfer. In graph classification, A-GEM offers good backward retention but limited forward generalization, where its performance remains low even after multiple stages. MER strikes a more favorable balance, showing both decent retention and the ability to learn new tasks without significant interference.

f) Task-Specific Insights on Gate-Type Pairs: Performance across different gate-type tasks also reveals interesting insights. Tasks involving `xor`/`xnor` (Task$_3$) tend to be more challenging due to structural asymmetry and less frequent occurrences, leading to greater forgetting in regularization-based methods like EWC and SI. In contrast, replay-based approaches such as MER manage to sustain performance on these tasks by explicitly storing past samples. Tasks involving structurally similar gates (e.g., `nand`/`nor`) generally see better retention, likely due to shared topological patterns. CoPE shows strong recall in these scenarios, indicating its strength in capturing gate-specific connectivity nuances.

g) Comparing GNN Paradigms: Graph classification appears more forgiving—likely due to the broader granularity of entire-circuit features—which explains the consistent performance across methods like EWC, SI, and iCaRL. Nonetheless, CoPE and MER again emerge as favorable, demonstrating adaptability regardless of graph abstraction level.

h) Method Suitability for Dynamic Netlist Learning: Overall, our results suggest that replay-based methods such as MER are best suited for dynamic netlist learning, especially when dealing with high task variability and structural diversity. MER, in particular, offers a strong balance between plasticity and stability, making it ideal for real-world EDA pipelines. In contrast, regularization-based methods like EWC and SI are more appropriate in resource-constrained environments where memory buffers may be limited, but their susceptibility to forgetting limits their practical use in highly dynamic design settings. CoPE also proves competitive, especially in graph-level reasoning, and could be useful for tasks requiring global circuit-level classification.

C. Catastrophic Forgetting Metric Insights

a) Parameter Drift Predicts Forgetting in Graph Classification: The Pearson correlation between the ℓ_2 parameter drift $\|\Delta\theta\|$ and the drop in accuracy ΔAcc varies across methods as shown in Table II. Under EWC, we observe a moderate positive correlation for Graph Classification ($r \approx 0.39$), indicating that larger overall parameter updates actually coincide with less forgetting. This suggests that, in graph-level tasks, parameter drift can sometimes reflect beneficial adaptation rather than harmful forgetting.

b) Method-Specific Drift Outperforms Raw Parameter Drift: When we replace $\|\Delta\theta\|$ with each algorithm's internal drift metric (e.g., Fisher-weighted updates for EWC, importance-weighted SI drift, distillation-based drift for iCaRL), correlations strengthen dramatically. EWC's method drift exhibits a strong positive correlation in Graph Classification ($r \approx 0.77$), demonstrating that penalizing critical parameters effectively anchors past wiring rules and reduces forgetting.

c) Structural Similarity Modulates Drift Impact: Gate-type pairs with shared subgraph motifs (AND/OR, NAND/NOR) show smaller drifts and milder accuracy drops under most methods, which explains why correlations there are weaker yet consistently positive. By contrast, transitions into the exclusive-OR domain incur larger drifts (e.g., EWC's

$\|\Delta\theta\| \approx 9.7$ for Stage 1$\rightarrow$3) and more severe drops, amplifying the predictive power of method-specific drift metrics in those harder scenarios.

d) Role of GNN Paradigms in Graph Classification: In graph classification, methods like EWC and iCaRL exhibit resilience to forgetting, indicating that their drift facilitates rather than hinders knowledge consolidation. Replay-based methods such as MER also sustain performance effectively, while CoPE demonstrates strong recall in structurally complex gate-type tasks. Overall, drift metrics provide meaningful signals for understanding stability–plasticity trade-offs in graph-level continual learning.

D. Continual Learning in Circuits Versus Other Domains

While continual-learning research spans image classification, speech recognition, reinforcement learning, IoT anomaly detection and network intrusion, the circuits domain introduces unique challenges and opportunities. Below we compare our experimental findings (from Graph tasks) to those typical of other application areas.

a) Heterogeneous Task Definitions: In vision benchmarks (e.g. class-incremental CIFAR [1] or MNIST [2], [3]), each task shares the same input space and differs only by label subset. By contrast, our circuits experiments focus on quartile-based graph classification, which introduces its own structural inductive bias. This requires algorithms to reason over global circuit properties, a complexity rare in pixel or spectrogram inputs.

b) Severity and Shape of Forgetting: Image-domain studies often report a gradual accuracy decline of twenty to thirty points over ten or more tasks under simple regularization [4]. In circuits, we see steeper drops even in just four stages. Graph Classification is somewhat more forgiving—losing only about sixteen points—but still underscores how quickly structural rules can be overwritten when tasks vary widely.

c) Drift–Forgetting Correlation Patterns: In many non-graph domains, raw parameter drift ($\|\Delta\theta\|$) tracks forgetting consistently: larger updates almost always mean more catastrophic forgetting [4], [5]. Our circuits data break that pattern. Under EWC (ref Table II), Graph tasks exhibit a moderate positive correlation (around $r = 0.39$), indicating that larger overall parameter updates can coincide with less forgetting. This divergence highlights how graph-level reasoning interacts differently with drift compared to image or speech models [3], [6].

d) Algorithmic Trade-offs: Replay-based methods like MER and iCaRL often lead in vision and speech, but circuits amplify their distinctions. In Graph Classification, A-GEM's cosine-drift measure even flips sign ($r \approx -0.90$), a reversal rarely seen in standard benchmarks. MER and CoPE, by contrast, sustain stronger performance across stages, showing that replay and hybrid strategies are particularly effective at the graph level.

e) Structural Versus Statistical Drift: IoT anomaly detection and network intrusion systems face concept drift in streaming timeseries [7], [4], where abrupt changes degrade model quality. Circuits drift arises from entirely new subgraph patterns, not simply shifting data distributions. As a result, method-specific drift metrics—Fisher-weighted for EWC, distillation loss for iCaRL—prove more diagnostic than global parameter norms, whereas in many continuous-valued domains practitioners often rely on simpler L_2 drift measures.

f) Implications for Dynamic Netlist Learning: Our results suggest that circuit-specific structure demands both fine-grained drift control and flexible replay. MER, in particular, offers a strong balance between plasticity and stability in graph classification, while CoPE demonstrates competitive recall on structurally complex gate families. Meanwhile, in resource-limited IoT nodes or embedded FPGA monitors, EWC's lightweight Fisher penalty remains a viable fallback, echoing its low-overhead appeal in mobile vision applications but accepting a steeper drop when the next "task" is a wholly new gate family.

References

[1] G. Kim, C. Xiao, T. Konishi, Z. Ke, and B. Liu, "A theoretical study on solving continual learning," *Advances in neural information processing systems*, vol. 35, pp. 5065–5079, 2022.

[2] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[3] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *International conference on machine learning*. PMLR, 2017, pp. 3987–3995.

[4] R. R. Karn, P. Kudva, and I. M. Elfadel, "Learning without forgetting: A new framework for network cyber security threat detection," *IEEE Access*, vol. 9, pp. 137 042–137 062, 2021.

[5] R. R. Karn, J. Knechtel, and O. Sinanoglu, "Progressive learning with recurrent neural network for sequence classification," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, no. 3, pp. 1591–1595, 2023.

[6] Y. Zhang, Y. Xiong, D. Li, C. Shan, K. Ren, and Y. Zhu, "Cope: modeling continuous propagation and evolution on interaction graph," in *Proceedings of the 30th ACM international conference on information & knowledge management*, 2021, pp. 2627–2636.

[7] R. R. Karn, P. Kudva, and I. M. Elfadel, "Criteria for learning without forgetting in artificial neural networks," in *2019 IEEE International Conference on Cognitive Computing (ICCC)*. IEEE, 2019, pp. 90–97.