Experimental Evaluation of Time-Series Class-Incremental Learning with TSCIL Framework

Tyler Marino
Université Paris-Saclay
tylermarino.research@gmail.com

João Miguel Figueiredo
Universidade do Porto
up202108829@up.pt

ABSTRACT

As progress is made and evolution of ideas and objects prevail through time, models of these instances must keep pace. Current Methods in Time-Series Class-Incremental Learning (TSCIL) struggle to improve in real time and recall the past events that have informed previous instances. For this reason, we demonstrate how 6 model structures of TSCIL preform on classification tasks, addressing common problems such as catastrophic forgetting and resource efficiency.

1 INTRODUCTION

1.1 Context and Motivation

Continual learning (CL) is designed for models to adapt parameters and understanding of context as more data is generated. Time series data is a natural input to a continual learning model however it has not been as extensively explored as text and image learning. Temporal dependencies, variable sequence lengths and missing data have challenged researchers and models alike to gain a most realistic understanding of the data as it is now and how it may be in the future. Real world applications such as healthcare, environmental and urban systems, and financial markets have continuously evolving data relations as new interactions occur between objects within and around the systems in study. There is too much value in the past to forget, but to retrain an entire model at each timestep is too computationally expensive, thus making continual learning for time series data and exciting path forward for frugal AI researchers.

Two architectural flaws that inhibit greater success of these models include (1) normalization strategies which affect the existing stability of feature spaces and (2) the amount of information persisting from one learning phase to the next(replay buffer size). Therefore understanding how these two components direct model ability is of great importance to train resource-conscious learning systems.

1.2 Research Question

This work investigates the following research question:

> *How do design choices such as replay memory, normalization strategy, and encoder architecture affect the trade-off between stability and plasticity in continual learning for time-series data?*

We study this question using the **TSCIL** framework [4], which provides a standardized benchmark for class-incremental learning on time-series datasets. Our goal is to gain empirical insight into how architectural variation of agents, encoders, epochs, and memory budgets—impact the trade-off between knowledge retention(stability) and adaptation to new tasks (plasticity).

1.3 Task Definition

The task under investigation is *class-incremental time-series classification.* Inputs consist of multivariate temporal signals $x_t \in \mathbb{R}^{T\times D}$ collected sequentially over time, and outputs are discrete class labels $y \in \{1, ..., C\}$. At each incremental step k, **the model receives samples from a subset of new classes C_k, with previous data being partially or fully inaccessible.** The model must classify the union of all classes that existed at each timestep.

2 RELATED WORK

2.1 Overview of Existing Approaches

Early studies in continual learning primarily focused on preventing catastrophic forgetting through three major families of methods: (i) *replay-based*, (ii) *regularization-based*, and (iii) *dynamic-architecture* approaches [2, 5, 7]. Replay methods store a small subset of exemplary class data to input into the current timestep of training data so as to not lose memory of past tasks. iCaRL [5] is an example case using Experience Replay (ER) which relates new data to that which already exists while still being open to learning new classes. Depending on the size of the subset or buffer of past data, this approach provides a strong means concerning stability and computational efficiency. Regularization-based methods like Elastic Weight Consolidation (EWC) [2] position important nodes so that they are resistant to diminishing weights over time, weights that may be overwritten as timesteps progress. Dynamic methods, such as Progressive Networks [7], add new modules per task, avoiding forgetting at the cost of growing model size.

In recent years, researchers have explored how *normalization* and *buffer management* influence continual learning dynamics. Pham et al. [3] proposed *Continual Normalization*, highlighting Batch Normalization's increasing biased statistics over time and worsening forgetting. They demonstrated that changing the BN with a more generalist normalization that can persist over time, retention increases with continual learning.

Meanwhile, studies like Bhat et al. [1] analyzed small-bufferd, revealing that consistency regularization and strategic buffer sampling can yield sustainable accuracy even with constrained memory. Zhao et al. [9] further showed that forgetting affects specific network layers disproportionately, suggesting normalization layers play a critical role in stabilizing intermediate representations. Yin et al. [8] introduce Temporal Teacher Distillation (TTD), a method that uses attentive recurrent neural networks where distillation includes a loss to enforce non-forgetfulness. This approach however does not explore as in depth various buffer sizes and normalization strategies,

2.2 Summary of the TSCIL Framework

Qiao et al. [4] introduced the **Time-Series Class-Incremental Learning (TSCIL)** benchmark, a framework for evaluating continual learning methods on time-series data. Providing a benchmark example of class incremental learning using classic time series classification where consistent metrics of accuracy, forgetting and computational cost can be utilized across future trials. These metrics include average accuracy across all tasks, performance on tasks from previous timesteps, memory footprint and runtime. They test naive model (standard fine tuning) and compare other models including properties such as replay with a small buffer, regularization of weight changes and computationally expensive dynamic models that introduce new features as the time goes. This study helps lay foundation for future studies regarding buffer size and normalization affects.

3 EVALUATION

This project explores the problem of **continual learning for time-series classification**, where a model must learn new classes sequentially without retraining from scratch. Such a setting captures realistic data-stream conditions but also introduces the well-known challenge of *catastrophic forgetting*, in which performance on previously learned tasks degrades as new ones are introduced.

3.1 Motivation and Framing

Following the ideas presented in **TSCIL (Time-Series Continual Incremental Learning)** [4], we aimed to empirically explore how architectural and algorithmic design choices influence continual learning behaviour in constrained environments. Building on the research question introduced previously, our objective was to evaluate how different configurations, such as replay memory size, normalization strategy, and encoder architecture affect the trade-off between stability and plasticity.

From this motivation, we formulated three working hypotheses:

- **H1:** Experience Replay (ER) mitigates catastrophic forgetting compared to Sequential Fine-Tuning (SFT).
- **H2:** Increasing replay memory improves stability, but with diminishing returns beyond a certain capacity.
- **H3:** Layer Normalization (LN) offers more consistent results than Batch Normalization (BN) for small datasets.

3.2 Experimental Setup and Constraints

We used the **UCI Human Activity Recognition (HAR)** dataset [6], which contains smartphone accelerometer and gyroscope recordings from 30 individuals performing six daily activities such as walking, sitting, and standing. Each sample represents multivariate time-series data collected from wearable sensors, making the dataset well-suited for studying temporal recognition under incremental learning.

All experiments were conducted using the TSCIL framework to handle task creation, model initialization, and evaluation. Due to compatibility issues between CUDA and the framework's dependencies, all runs were executed on **CPU only**, using an **Intel Core Ultra 9 275HX** processor. Despite the lack of GPU acceleration, each full experiment (20 epochs per task) completed within 10–20 minutes, demonstrating that meaningful continual learning studies can be performed on consumer hardware.

The hardware constraint introduces variability in runtime performance due to factors such as system load and thermal throttling. Therefore, our discussion of efficiency focuses on **expected computational trends**: for example, how larger memory buffers or more complex encoders would theoretically affect latency or energy consumption—rather than absolute time measurements.

3.3 Task Configuration and Evaluation Protocol

In the TSCIL framework, continual learning is simulated by dividing the dataset's activity classes into multiple disjoint subsets, each representing a separate **task**. The UCI-HAR dataset contains six human activities:

Class ID	Activity Label
0	Walking
1	Walking Upstairs
2	Walking Downstairs
3	Sitting
4	Standing
5	Laying

Under the **class-incremental learning (CIL)** scenario with `-stream_split exp`, these six classes are divided into three sequential tasks, each introducing two new activities. In our experiment, the class order was randomized as `[5, 2, 1, 3, 0, 4]`, resulting in the following task structure:

Task	Class IDs	Activities
Task 1	[5, 2]	Laying, Walking Downstairs
Task 2	[1, 3]	Walking Upstairs, Sitting
Task 3	[0, 4]	Walking, Standing

During *Task 1*, the model learns to distinguish between the first two activities. When *Task 2* begins, the model receives data for two new classes while retaining its parameters from the previous stage—without direct access to prior training data. Finally, in *Task 3*, the remaining two activities are introduced, completing the full six-class classification problem.

Each task was trained for **20 epochs**, following the same optimization schedule and learning rate strategy across all experiments to ensure consistency. After training on each task, the model was evaluated on the test sets of all tasks seen so far:

`Task 1` $\longrightarrow$ `Task 2` $\longrightarrow$ `Task 3`, with evaluations after each stage: `E(1)`, `E(1,2)`, `E(1,2,3)`.

This setup enables the analysis of both **knowledge retention** (accuracy on earlier tasks) and **adaptation** (accuracy on newly learned tasks). It reflects a realistic continual learning scenario in which models must integrate new information over time while avoiding catastrophic forgetting.

3.4 Metrics

To ensure consistency with prior work, we adopted the same evaluation protocol as TSCIL. The main performance metrics were:

- **Average End Accuracy** — the mean accuracy across all tasks after the final incremental stage.

- **Average End Forgetting** — the average performance degradation on earlier tasks.
- **Average Current Accuracy** — the accuracy on the most recently learned task.

Together, these metrics capture the balance between **stability** (retention of past knowledge) and **plasticity** (adaptation to new data), the two competing objectives in continual learning. Approximate training time was also tracked as a qualitative proxy for computational efficiency.

To support reproducibility, we implemented a dedicated Python pipeline function, `run_experiment()`, which automates configuration handling, logging, and metric extraction. Each run was identified by its key parameters: agent type, encoder, normalization, memory budget, and number of epochs, allowing seamless comparison across experiments.

3.5 Feasibility and Research Scope

Because the experiments were limited to CPU execution and a compact dataset, our scope emphasizes **conceptual trends** rather than large-scale benchmarking. Certain continual learning strategies, may depend on hyperparameter tuning and computational stability that exceed what can be reliably achieved in a lightweight setting. Nonetheless, this configuration offers a valuable sandbox for exploring the theoretical trade-offs that define frugal continual learning.

4 EXPERIMENTS

To interpret our findings systematically, experiments were conducted incrementally, each building upon the last. This progressive approach reveals how individual design choices—replay memory, normalization, and model type—affect continual learning dynamics. The experimental sequence was as follows:

- **Experiment 1:** Baseline SFT with CNN + LayerNorm (no replay).
- **Experiment 2–3:** Experience Replay (ER) with 5% buffer, comparing LN and BN.
- **Experiment 4:** Regularization-based continual learning (EWC).
- **Experiment 5a–5b:** Transformer encoder variants (with and without replay).
- **Experiment 6:** Increasing memory budgets (10–50%) for stability analysis.

Each subsection reports the setup, quantitative results, and qualitative observations relevant to the above hypotheses.

4.1 Experiment 1 — SFT Baseline

We began with the simplest configuration: the **Sequential Fine-Tuning (SFT)** agent using a **CNN encoder** and **Layer Normalization (LN)**. This setup trains the model sequentially on new tasks without any replay buffer or explicit regularization, providing a reference point for observing catastrophic forgetting.

Evaluation and Results. Figure 1 illustrates the **Forgetting Curves** obtained for Experiment 1. Each line corresponds to the model's evaluation after completing a task, while the x-axis represents the task being evaluated. The plot therefore visualizes how accuracy on earlier tasks evolves as new ones are introduced.

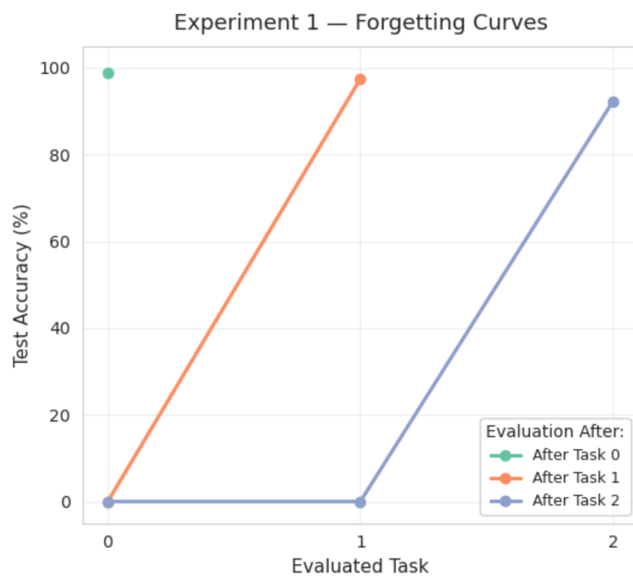

Figure 1: Experiment 1 — Forgetting Curves. Each line shows test accuracy on previously learned tasks after each incremental training step. The rapid accuracy drop (backwards) demonstrates catastrophic forgetting in the absence of replay or regularization.

As expected, performance degraded sharply on earlier tasks as new ones were introduced. The model retained nearly perfect accuracy on the most recent task but failed to preserve information about prior activities, confirming the instability of naive sequential training. This pattern exemplifies *catastrophic forgetting*, where parameter updates for new knowledge overwrite representations of older ones.

Table 1 summarizes the key metrics for this experiment, all computed on the **test set**. The **Average End Accuracy** of only 30.76% highlights severe loss of previously learned knowledge, while the **Average End Forgetting** reached 98.12%, indicating that nearly all information about earlier tasks was lost. However, the **Average Current Accuracy** remained high (96.18%), showing that the model can still learn new tasks effectively when not constrained by memory of the past.

Table 1: Experiment 1 — Test Set Summary Metrics

Metric	Value (%)
Average End Accuracy	30.76
Average End Forgetting	98.12
Average Current Accuracy	96.18

These results establish the lower bound for continual learning performance under our experimental settings. They clearly demonstrate the central problem that continual learning methods aim to solve: preserving knowledge stability while maintaining the ability to adapt.

4.2 Experiment 2 — Experience Replay (ER) with Small Buffer (5%)

Building upon the SFT baseline, **Experiment 2** introduces the **Experience Replay (ER)** mechanism, which reuses a small portion of past samples during training to mitigate forgetting. The model employs the same **CNN encoder** with **Layer Normalization (LN)**, but a **5% memory buffer** is now allocated to store examples from previously learned tasks.

This approach enables the model to periodically revisit older samples while learning new ones, thereby improving retention of earlier knowledge. The goal of this experiment is to observe how even a small replay memory influences stability and overall performance under identical computational constraints.

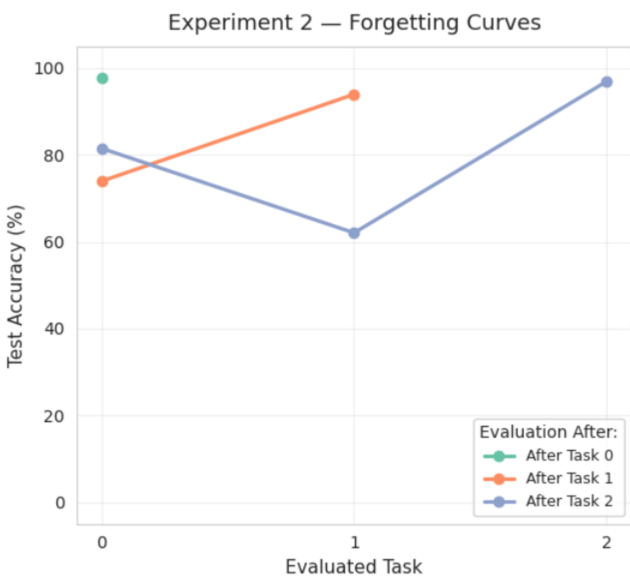

Figure 2: Forgetting curves for Experiment 2 — Experience Replay (5%) showing test accuracy after each task.

As shown in Figure 2, introducing Experience Replay substantially reduces catastrophic forgetting compared to the SFT baseline (Experiment 1). The model now retains far more information from earlier tasks, as reflected by the smoother and more consistent forgetting curves.

Quantitatively, the effect is clear: the **Average End Accuracy** increases from 30.76% to 83.74%, while the **Average End Forgetting** drops sharply from 98.12% to 24.15%. The **Average Current Accuracy** remains high at 96.19%, confirming that ER does not impair adaptation to new tasks.

Although the replay buffer is small, this experiment demonstrates that even limited memory can greatly enhance stability without a significant computational overhead. This aligns with findings from the TSCIL framework, where replay-based methods consistently outperform purely sequential approaches.

In subsequent experiments, we continue to use the 5% replay buffer while exploring the impact of other architecture choices.

4.3 Experiment 3 — ER with Batch Normalization (BN)

Continuing with the **Experience Replay (ER)** framework, **Experiment 3** explores the impact of using **Batch Normalization (BN)** instead of **Layer Normalization (LN)** in the CNN encoder.

Batch Normalization standardizes activations across each mini-batch, stabilizing and accelerating training by keeping feature distributions consistent. Unlike Layer Normalization, which normalizes across features within a single sample, BN relies on batch-level statistics—making it more sensitive to distributional changes between tasks.

In the context of continual learning, this distinction is crucial: while BN often improves convergence speed and training efficiency on stationary datasets, it can also introduce instability when the input distribution shifts significantly between tasks. This experiment therefore examines whether BN supports or hinders ER's ability to retain knowledge over sequential learning stages.

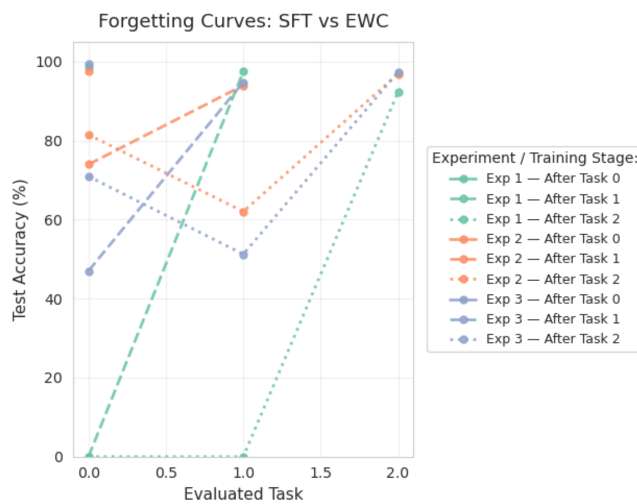

Figure 3: Forgetting curves for Experiment 3 — ER with Batch Normalization (BN) compared to previous runs.

As shown in Figure 3, switching to Batch Normalization resulted in noticeably lower stability compared to the LayerNorm-based ER configuration (Experiment 2). The forgetting curves reveal a steeper decline in performance on earlier tasks, suggesting that batch-dependent normalization may amplify distributional shifts across incremental steps.

Quantitatively, the **Average End Accuracy** decreased to 73.19%, and **Average End Forgetting** increased to 35.98%, confirming weaker retention despite similar overall accuracy on the most recent task. These results indicate that, although BN is theoretically more efficient for large-scale or stationary data due to improved gradient flow, its reliance on mini-batch statistics makes it less robust for continual learning on small or non-stationary datasets.

Given these observations, we chose to continue using **Layer Normalization (LN)** in subsequent experiments to ensure consistent and reliable comparisons. Nonetheless, in larger or more data-rich continual learning settings, **Batch Normalization** could still be advantageous from a theoretical efficiency standpoint, where larger and more stable batches would provide more reliable statistics, and it is the best option for real-world scenarios.

4.4 Experiment 4 — Elastic Weight Consolidation (EWC)

In **Experiment 4**, we explore a fundamentally different approach to continual learning based on **Elastic Weight Consolidation (EWC)**. Unlike Experience Replay, which relies on storing past samples, EWC is a *regularization-based method* that preserves previous knowledge by selectively constraining updates to parameters deemed critical for earlier tasks.

After each training phase, the model estimates the **Fisher Information Matrix**, which quantifies how important each parameter was for the previous task. When learning a new task, a penalty term is added to the loss function, preventing large deviations in these important weights. This mechanism helps balance **stability** (retaining learned information) and **plasticity** (adapting to new tasks) without needing a replay buffer.

However, our results revealed that this approach performed poorly under our setup, showing clear signs of **catastrophic forgetting**. Despite its theoretical appeal, the model almost completely lost performance on earlier tasks, resulting in an**Average End Forgetting** of 98.48%. This outcome likely stems from an imbalanced regularization strength (λ): overly strong penalties can restrict adaptation to new tasks, while too weak ones fail to preserve prior knowledge.

While EWC can be highly effective with careful tuning or in domains with well-separated tasks, our time constraints made it less suitable. For the remainder of this study, we therefore **return to the Experience Replay (ER)** framework, which showed more reliable performance, interpretability, and consistency under the same experimental conditions.

4.5 Experiment 5 — ER with Transformer Encoder

In this experiment, we aim to enhance the Experience Replay (ER) approach by incorporating a **Transformer Encoder** into the model architecture. Transformers have demonstrated strong performance in sequence modeling due to their ability to capture long-range dependencies and contextual information, which could, in principle, improve continual learning by enabling better reuse of past experiences.

Experiment 5a introduced a Transformer encoder under the Sequential Fine-Tuning (SFT) setup to establish a new baseline. The results were nearly identical to those obtained with the CNN-based SFT model (within a ±1% margin of accuracy), still exhibiting pronounced **catastrophic forgetting**. This indicates that while Transformers can capture temporal structure more effectively, they do not inherently solve the problem of knowledge retention across tasks, as expected.

Building on this, **Experiment 5b** evaluated the Transformer within the Experience Replay framework and compared its performance to the CNN-based ER model (Experiment 2). As shown in Figure 4, the CNN consistently achieved better results in both accuracy and stability. The Transformer-based ER reached an **Average End Accuracy** of 70.18% and an **Average End Forgetting** of 37.76%, compared to 80.15% and 24.06% respectively for the CNN model.

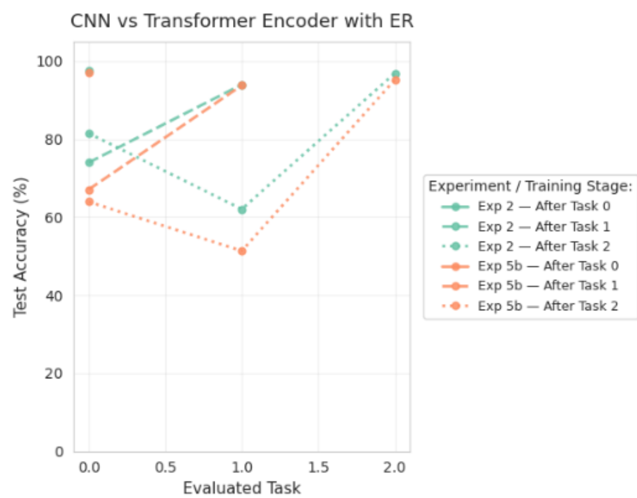

Figure 4: Experiment 5b — Comparison of CNN and Transformer encoders under Experience Replay. CNN achieved higher retention, with simpler computation.

Overall, the **Transformer-based encoder did not outperform the CNN** at all. The added architectural complexity did not lead to measurable gains in continual learning stability, likely due to the small dataset size and limited sequence length, where attention mechanisms offer little additional benefit. Furthermore, the CNN's lower computational footprint makes it better aligned with the objectives of **frugal AI**. For this reason, we continue with the CNN encoder for the final experiment, focusing on memory efficiency.

4.6 Experiment 6 — ER with Varying Memory Budgets

In this experiment, we investigate how different replay memory budgets influence the performance of the **Experience Replay (ER)** agent. By varying the amount of memory allocated for storing past samples, we aim to identify an optimal configuration that balances accuracy, retention, and computational efficiency — key aspects of frugal continual learning.

The results, shown in Figure 5, reveal that performance remains relatively stable across different memory budgets, with only minor variations. The **Average End Accuracy** ranges between 87–91%, and **Average Forgetting** stays low (7–13%), demonstrating that even small buffers are sufficient to retain much of the prior knowledge.

Interestingly, a subtle **parabolic trend** emerges, with the best results observed around a **40% memory budget**. At this point, the

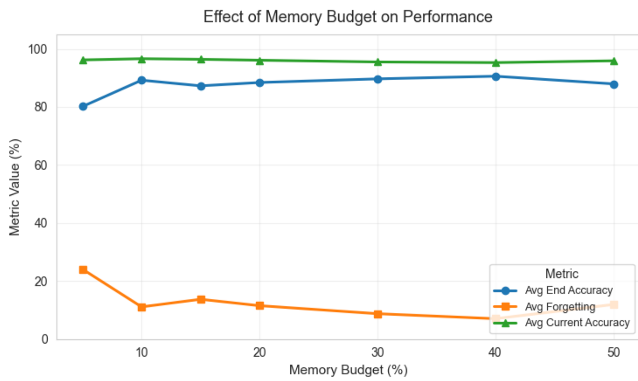

Figure 5: Effect of memory budget on performance metrics. Increasing the buffer improves stability up to a moderate point before diminishing returns appear.

model achieves an **Average End Accuracy** of 90.58% and the lowest **Average Forgetting** of 7.03%. However, despite being the strongest performance on these two metrics, it exhibited the lowest **Average Current Accuracy** among all configurations, highlighting the **stability–plasticity trade-off**. Beyond this point, performance slightly declines, suggesting that larger buffers may introduce redundancy or interfere with learning dynamics, while smaller ones may not capture enough representative samples. This trend reflects the inherent balance between **stability** (retaining past information) and **plasticity** (adapting to new tasks).

Although our evaluation remains relatively simple, these findings align with theoretical expectations: a **moderate replay memory** offers the most balanced trade-off between resource efficiency and continual learning stability. From a frugality perspective, allocating excessive memory yields diminishing returns, while moderate replay buffers achieve comparable accuracy with reduced computational and storage costs.

To further illustrate the relationship between memory budget and performance, we compared two representative configurations: **ER with 10%** and **ER with 40%** memory buffers.

As shown in Figure **??**, the larger buffer achieves slightly higher accuracy but at a substantial increase in memory usager: 4 times bigger buffer. Specifically, the 10% buffer reached an **Average End Accuracy** of 89.20% with an **Average Forgetting** of 11.10%, while the 40% buffer improved these values to 90.58% and 7.03%, respectively. However, the **Average Current Accuracy** decreased marginally from 96.60% to 95.27%, suggesting that the gain in long-term retention comes at a small cost to immediate adaptability.

This comparison underscores a key principle of continual learning under constrained resources: the **trade-off between memory capacity and performance efficiency**. While larger buffers can enhance stability by storing more representative samples, their benefits quickly diminish relative to their computational and memory cost. In practice, the 10% configuration already achieves near-optimal performance with far lower resource demands, making it possibly a more frugal and balanced choice for efficient continual learning.

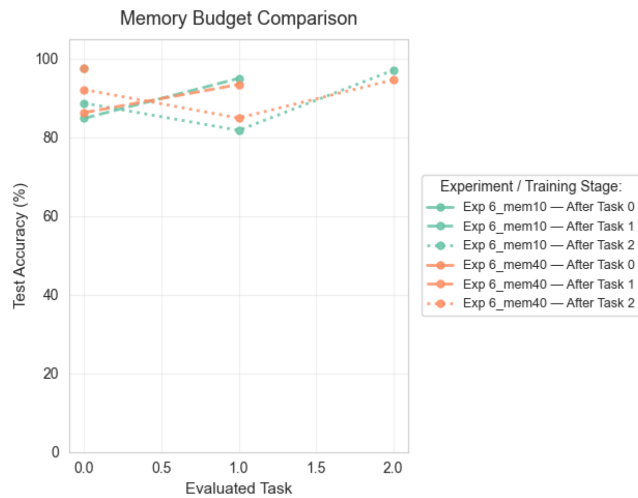

Figure 6: Visual comparison of performance curves for smaller (10%) and larger (40%) memory budgets.

5 DISCUSSION AND INSIGHTS

In this section, we revisit the research question and hypotheses outlined previously to synthesize the main findings from our experiments.

5.1 Revisiting the Research Question

Our central research question asked:

> *How do design choices such as replay memory, normalization strategy, and encoder architecture affect the trade-off between stability and plasticity in continual learning for time-series data?*

Across all experiments, our results confirmed that these factors significantly influence continual learning dynamics, even in small-scale, resource-limited environments. The observed trade-offs between stability (retaining past knowledge) and plasticity (adapting to new tasks) align closely with theoretical expectations discussed in the literature.

5.2 Hypothesis Evaluation

H1: Experience Replay mitigates catastrophic forgetting compared to Sequential Fine-Tuning. This hypothesis was strongly supported. Introducing a replay buffer (Experiment 2) drastically improved retention of previous tasks, raising the **Average End Accuracy** from 30.76% (SFT baseline) to over 80%, and reducing **Average Forgetting** from nearly 100% to around 24%. This confirms that even minimal memory replay is highly effective for maintaining stability in continual learning.

H2: Increasing replay memory improves stability, but with diminishing returns beyond a certain capacity. Our memory variation study (Experiment 6) validated this hypothesis as well. While larger buffers consistently reduced forgetting, the improvements flattened beyond a moderate size (around 40% memory). This suggests that once sufficient representative samples are stored, adding more memory yields only marginal gains, while increasing resource consumption, a key consideration for frugal AI systems.

H3: Layer Normalization offers more consistent results than Batch Normalization for small datasets. This hypothesis was also supported by Experiment 3, where substituting BN for LN led to degraded performance and slightly changed efficiency. BN's dependence on batch statistics makes it more sensitive to data shifts between tasks, while LN provided more stable results under our small sample conditions. In larger scale and real world experiments, however, BN may still prove more advantageous due to its faster convergence and stronger regularization effects.

5.3 Key Takeaways

Overall, these findings provide empirical evidence for the central premise of continual learning: achieving robustness over time requires balancing memory, stability, and adaptability. Replay-based approaches consistently outperformed others, demonstrating greater reliability under limited resources. Moreover, the experiments highlighted that **frugality does not necessarily imply lower accuracy**, but rather a more thoughtful distribution of computational and memory resources to maintain consistent learning performance.

6 CONCLUSIONS

This work presented a short yet insightful investigation into the behavior of continual learning methods for time-series data using the **TSCIL** framework. Conducted over a two-week period, the objective was not to achieve state-of-the-art performance, but rather to apply our theoretical understanding of continual learning to practical experimentation. By implementing, running, and analyzing multiple configurations, we gained hands-on experience into how replay memory, normalization, and model architecture influence the balance between **stability** and **plasticity**.

Through six experiments, we observed and confirmed several key patterns discussed in class:

- Sequential fine-tuning without replay leads to severe catastrophic forgetting.
- Even a small replay buffer significantly improves retention, though larger buffers yield diminishing returns.
- Normalization and architecture choices impact learning dynamics but do not guarantee better performance under constrained settings.

From a broader perspective, this study highlighted the fundamental trade-offs of **frugal continual learning**: balancing accuracy, retention, and efficiency within limited computational resources. Although our experiments were relatively simple and CPU-bound, they demonstrate that meaningful continual learning analysis is possible without access to high-end hardware.

With more time, several improvements could be made: we would explore a wider range of buffer management strategies, perform hyperparameter tuning for regularization-based methods like EWC, and extend the benchmark to more complex datasets and encoder architectures (e.g., lightweight Transformers). Additionally, integrating energy and latency measurements would provide a more complete view of the efficiency-performance trade-off central to frugal AI research.

Overall, this project was both enjoyable and informative. It allowed us to connect theoretical concepts from class to empirical results, reinforcing our understanding of continual learning dynamics and the practical compromises that arise when optimizing for efficiency, stability, and adaptability.

In essence, our experiments reaffirm that Frugal AI is not only about achieving high accuracy with fewer resources, but about understanding and managing the trade-offs that define sustainable, efficient learning over time.

ACKNOWLEDGMENTS

We acknowledge the use of large language models (ChatGPT [**?**]) to support parts of the writing, editing, and clarification of technical content in this report. It was used to assist in structuring sections and refining phrasing, but all research and synthesis were performed by the authors. LLMs were also used on the practical component, mainly to write markdown cells explaining each notebook

REFERENCES

[1] Prashant Bhat, Bahram Zonooz, and Elahe Arani. 2022. Consistency is the Key to Further Mitigating Catastrophic Forgetting in Continual Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*. https://proceedings.neurips.cc/paper_files/paper/2022/hash/d08622dc6e513518ec652a548d4ff80b-Abstract-Conference.html Experience replay and consistency regularization.

[2] 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. 2017. Overcoming Catastrophic Forgetting in Neural Networks. *Proceedings of the National Academy of Sciences (PNAS)* 114, 13 (2017), 3521–3526. https://doi.org/10.1073/pnas.1611835114 Regularization-based continual learning (EWC).

[3] Quang Pham, Chenghao Liu, and Steven C. H. Hoi. 2022. Continual Normalization: Rethinking Batch Normalization for Online Continual Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*. https://openreview.net/forum?id=0YUkAUkZgYb Normalization strategies for continual learning.

[4] Zhiwei Qiao, Rui Zhang, Yan Zhang, Wei Xie, and Xue Li. 2024. Class-Incremental Learning for Time Series: Benchmark and Evaluation. In *Proceedings of the 30th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. https://doi.org/10.1145/3637528.3659610 TSCIL framework benchmark.

[5] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. iCaRL: Incremental Classifier and Representation Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR.2017.587 Replay-based continual learning baseline.

[6] Jorge Luis Reyes-Ortiz, Davide Anguita, Alessandro Ghio, Luca Oneto, and Xavier Parra. 2013. *Human Activity Recognition Using Smartphones*. https://doi.org/10.24432/C54S4K Dataset.

[7] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive Neural Networks. *arXiv preprint arXiv:1606.04671* (2016). https://arxiv.org/abs/1606.04671 Dynamic architecture continual learning.

[8] Shao-Yu Yin, Qin Zhang, Yu Xiang, Yong Ai, Dong-Sheng Li, Yong Xu, and Jie-Lu Zeng. 2022. Continual learning with attentive recurrent neural networks for temporal data classification. *Neural Networks* 158 (2022), 171–187. https://doi.org/10.1016/j.neunet.2022.07.010 Temporal continual learning using attentive recurrent models.

[9] Haiyan Zhao, Yihan Zhang, Wei Zhang, Wei Chen, and Jiang Bian. 2023. Does Continual Learning Equally Forget All Parameters?. In *Proceedings of the 40th International Conference on Machine Learning (ICML)*. https://proceedings.mlr.press/v202/zhao23n.html Analysis of parameter sensitivity and buffer effects.