Evaluating Catastrophic Forgetting in Neural Networks Trained with Continual Backpropagation

Justinas Jučas

Supervisors: Laurens Engwegen, Wendelin Böhmer

[1]EEMCS, Delft University of Technology, The Netherlands

Abstract

Continual Backpropagation (CBP) has recently been proposed as an effective method for mitigating loss of plasticity in neural networks trained in continual learning (CL) settings. While extensive experiments have been conducted to demonstrate the algorithm's ability to mitigate loss of plasticity, its susceptibility to catastrophic forgetting remains unexamined. This work addresses this gap by systematically evaluating the magnitude of catastrophic forgetting in models trained with CBP and comparing it to four baseline algorithms. We demonstrate that CBP suffers from significantly higher forgetting compared to all tested baselines, particularly in long-term and periodically revisited task scenarios. Moreover, we find that specific hyperparameters of the algorithm have significant influence on the stability-plasticity trade-off. We further analyze the internal dynamics of CBP, identifying strong correlations between forgetting and metrics such as activation drift. Finally, we evaluate three modifications to CBP: noise injection, layer-specific replacement, and partial neuron replacement, and show that the modifications reduce forgetting while maintaining high plasticity.

1 Introduction

Continual learning (CL) is a field of machine learning in which a model learns sequentially from a stream of data, instead of being trained once on a fixed dataset. Although significant progress has been made in this relatively new field, CL still faces two core challenges: **loss of plasticity** [1] and **catastrophic forgetting** [2, 3].

Catastrophic forgetting is a phenomenon where a neural network (NN) rapidly loses previously acquired knowledge when learning new information sequentially. This occurs due to the nature of standard training methods, such as stochastic gradient descent, which update model parameters to minimize the loss on the current objective without explicitly preserving knowledge from earlier stages [3]. As a result, information relevant to prior learning is gradually overwritten, sometimes almost entirely, as parameters critical to earlier solutions are modified.

The second problem, loss of plasticity, is a phenomenon in which a NN eventually becomes less capable of learning new trends when it is continuously trained on different data. This phenomenon occurs when certain neurons of the model eventually become specialized or "settled", as they begin to show very limited variability in their activation values when trained on any new data [1, 4]. Therefore, the model becomes less adaptable, as fewer parameters remain responsive to updates when exposed to new tasks.

Many algorithms have been shown to effectively mitigate either catastrophic forgetting or loss of plasticity separately [1, 5–12]. However, solving both catastrophic forgetting and loss of plasticity simultaneously remains the most fundamental problem in continual learning. The difficulty rises from the intrinsic trade-off, called **stability-plasticity dilemma** [13]. The stability-plasticity dilemma states that the more adaptable a model is, the faster it tends to forget prior knowledge. Alternatively, a highly stable model is naturally expected to be less plastic. Thus, any new online learning algorithm should ideally be assessed not only for its ability to prevent forgetting or maintain plasticity alone, but also for its effectiveness at balancing these two objectives.

As an attempt to solve the loss of plasticity problem, Dohare et al. [1] proposed an efficient approach, called Continual Backpropagation (CBP). The algorithm was shown to significantly decrease loss of plasticity for different NN architectures, across various experimental settings. The CBP algorithm works by reinitializing the incoming and outgoing weights of the neurons that have low utility scores, i. e. the neurons that presumably very minimally contribute to the final output. This reinitialization of neurons' weights has an effect of reactivating the mentioned "settled" neurons, in this way preserving model's adaptability. At the same time, performance is maintained, as the procedure aims to remove only parameters that contribute minimally to the network's output.

However, Dohare et al. [1] focused primarily on the plasticity of CBP, without assessing its susceptibility to catastrophic forgetting. While Silvestrin et al. [14] made a preliminary attempt to evaluate CBP's stability, their analysis was limited to a very simple experiment, which may not generalize to more complex CL settings. However, evaluation the forgetting of CBP might be crucial if it were further exploited in different settings, especially in applications where retaining prior knowledge is essential; for instance, in medical diagnosis systems that rely on historical patient data, or in lifelong language models. To address this gap, our work presents the **first** comprehensive evaluation of catastrophic forgetting in neural networks trained with CBP, and aims to extend the work of Dohare et al. [1] by proposing ways to improve stability-plasticity trade-off.

We systematically investigated CBS's effect on catastrophic forgetting. We began by evaluating whether CBP increases susceptibility to forgetting compared to different baselines, in particular, standard backpropagation, Shrink and Perturb (S&P) [6], L2 regularization, and the Adam optimizer [15]. Beyond standard performance metrics for forgetting and plasticity, we analyzed the internal dynamics of CBP-trained networks (i. e. weight and activation drift) and compared the found patterns with those observed under the baseline training algorithms. We next examined how CBP's and baseline algorithms' hyperparameters influence the stability-plasticity trade-off. Furthermore, in order to understand how well different algorithms manage to regain memory of lost information, we assessed algorithms' behavior when previously learned and then forgotten data was periodically reintroduced. Finally, we evaluated three distinct adjustments to CBP that were expected to improve the stability-plasticity trade-off.

Our results show that standard CBP suffers from significantly higher catastrophic forgetting compared to all tested baseline algorithms, both under single-pass and periodically revisited task setups. Furthermore, we found that specific hyperparameters of CBP strongly influence the balance between plasticity and stability. We also identified strong correlation between forgetting and model's activation drift, suggesting its potential as predictive indicator. Finally, we demonstrated that the three evaluated variants of CBP algorithm improve the stability of CBP, while only minimally reducing plasticity.

The remainder of this report is structured as follows: Section 2 reviews related work on catastrophic forgetting and loss of plasticity; Section 3 outlines the methodology and algorithms evaluated; Section 4 presents the experimental setup and results; Section 5 discusses key findings; and Section 6 and 7 conclude with remarks on responsible research and future directions.

2 Related Work

Continual learning aims to train neural networks on streaming data such that their performance matches that of models trained in a traditional offline setting. Two main challenges in this field are **catastrophic forgetting** [2, 3], where past knowledge is overwritten throughout continuous learning, and **loss of plasticity** [1], where the ability to learn new tasks is reduced as the learning proceeds. A multitude of methods have been proposed to solve each problem separately, while recent research has shifted towards developing methods that try to overcome both problems simultaneously.

2.1 Solving Catastrophic Forgetting

The notion of Catastrophic Forgetting was initially described in 1989 by McCloskey and Cohen [2]. Since then, different methods have been proposed to reduce the effect of the phenomenon.

Replay-based methods. *Replay-based* or *rehearsal* methods aim to mitigate catastrophic forgetting by storing the data of earlier tasks and mixing it together with the new data to form a replay buffer, from which the model is then re-trained [16–18]. While the approach is extremely simple and effective, it is resource-inefficient, as it requires storing potentially large amounts of past data. As an alternative, Rebuffi et al. [18] proposed a method called iCaRL, which, for image classification tasks, continually computes a mean feature vector of the past images, and uses it to mitigate forgetting. Additionally, *pseudo-rehearsal* methods were introduced, which, instead of storing the past data, generate synthetic samples that resemble the old data [5, 16]. An example of that is incrementally training an generative encoder [19], that is later used for older data samples generation, which in various cases has shown to be beneficial in CL settings [5, 20, 21].

Regularization methods. Regularization methods mitigate catastrophic forgetting by constraining updates to model parameters that are estimated to be important for retaining knowledge of previous tasks. A penalty is usually introduced to the loss function that discourages the model to move far away from already learned representation. The most simple example of that are L1 and L2 regularization methods, which are sometimes used as soft baselines for evaluating different algorithms [7–9]. Hinton and Plaut [10] were the first to propose a regularization-based approach to stabilize learning by introducing the concept of *fast* and *slow* weights, where fast weights adapt quickly to new information while slow weights retain long-term knowledge. Furthermore, Elastic Weight Consolidation (EWC) is a very effective and popular baseline, inspired by features of human brain [7]. It approximates sequential Bayesian learning using Fisher Information Matrix to estimate parameter importance and adds a quadratic penalty to discourage changes to the most important weights. However, while simple and effective, regularization methods rely on rough estimates of parameter importance, which can reduce learning accuracy for particular tasks.

Other methods. French [3] has shown that catastrophic forgetting generally occurs when the newly created internal features of the model interfere with the ones that were learned previously. Therefore, many methods aim to reduce the *feature representational overlap*, and that can be done in many ways. For example, Yoon et al. [8] proposed a method called Learning Without Forgetting (LWF), which, once a model is trained on a new task, adds a fully-connected hidden layer within the model. While this method achieves great performance, if a model is faced with a high number of new tasks, it would lead to an increased memory consumption, and computational overhead. Another approach, called Hard Attention to the Task [22], proposes to learn, for each task, its specific attention mask, i. e. the exact neurons within each layer that should majorly contribute to the output. Then the parameter updates are restricted mainly to the chosen neurons, and in the future tasks, algorithm prevents modifying parameters identified by earlier masks, which preserves previously learned knowledge. Lastly, Farajtabar et al. [23] proposed to project the gradient of a new task in the orthogonal direction of the previous tasks gradient, which was shown to reduce representational interference.

2.2 Solving Loss of Plasticity

Significantly less research is conducted in search of methods that would mitigate loss of plasticity. Dohare et al. [1] proposed *Continual Backpropagation*, a method we examine in this work, which works by periodically reinitializing weights that correspond to the least useful neurons. Continual Backpropagation was shown to significantly decrease loss of plasticity under various experimental settings. In the same study, it is also shown that Shrink and Perturb [6], a method that continuously shrinks and adds noise to the weights of the network, is an effective method for increasing the plasticity of a model. Abbas et al. [4], similarly as Dohare et al. [1], identified that the neurons become inactive throughout learning, and as a solution proposed replacing standard ReLU activations with their introduced Concatenated ReLU (CReLU) activations. It resulted in preserved gradient flow and thus ensured that the neurons remain active throughout learning. A similar method to the one proposed by Yoon et al. [8] was introduced to mitigate plasticity instead of reducing forgetting: Lyle et al. [11] showed that dynamically increasing network capacity, allows the model to maintain adaptability over time. However, this approach is not efficient memory-wise. Lastly, a sophisticated regenerative regularization loss, as proposed by Kumar et al. [12], can be combined with the arbitrary loss in order to encourage models to preserve their ability to reconstruct past task inputs from compressed latent representations.

2.3 Stability-Plasticity Dilemma

The Stability-Plasticity Dilemma, first introduced by Carpenter and Grossberg in 1987 [13], describes a fundamental trade-off in learning systems: the better a model is at retaining past knowledge (stability), the less flexible it becomes in adapting to new information (plasticity), and vice versa. For many years, the two issues were addressed separately, with more focus placed on reducing forgetting than on improving adaptability. However, recently more effort has been directed towards developing methods that aim to balance both sides of the trade-off [24, 25]. In particular, Kim et al. [25] proposed a SOTA approach to train an independent network for a new task, and then integrate its outputs into a second network that preserves information of previous tasks. Additionally, Elsayed and Mahmood [24] proposed to use an efficient utility function for estimating the most important weights for past information retention: these weights are then modified less, in this way retaining stability, while less important weights are updated, consequently maintaining plasticity.

The stability of Continual Backpropagation (CBP) was not evaluated in the original paper. A partial analysis was later conducted by Silvestrin et al. [14], however, it was limited to a simple Bit-Flipping task and used a simple neural network with a single hidden layer of five neurons. While the study demostrated low stability of CBP, due to the small scale and simplicity of the setup, the findings may not generalize to more complex models or diverse experimental conditions.

3 Methodology

In this section we describe the CBP algorithm in detail, introduce the baseline methods used for comparison, and present and motivate the proposed modifications to CBP aimed at improving its stability-plasticity trade-off.

3.1 Continual Backpropagation

Continual Backpropagation is a modified version of the standard backpropagation (BP) algorithm specifically designed to address the loss of plasticity that occurs in neural networks during continual learning. Introduced by Dohare et al. [1], the algorithm was demonstrated to effectively preserve the learning capacity of the models in various CL settings.

The algorithm works by periodically resetting incoming and outgoing weights of neurons that are estimated to very minimally contribute to the output. This resetting of neurons allows the model to explore new representations, while only minimally reducing its performance, as only the presumably least-important parameters are discarded. The neurons that are replaced are selected based on the utility function:

$$\mathbf{u}_l[i] = \eta \times \mathbf{u}_l[i] + (1-\eta) \times |\mathbf{h}_{l,i,t}| \times \sum_{k=1}^{n_{l+1}} |\mathbf{w}_{l,i,k,t}|,$$

where η is the decay rate parameter, $\mathbf{u}_l[i]$ is the utility value of i$^{\text{th}}$ neuron in layer l, while $\mathbf{h}_{l,i,t}$ and $\mathbf{w}_{l,i,k,t}$ are corresponding activation and weight parameter values.

Continual Backpropagation depends on several key hyperparameters: the learning rate (α), the replacement rate (ρ), the decay rate (η), and the maturity threshold (m). Learning rate controls the magnitude at which the weights are updated during each training step, replacement rate controls the proportion of neurons that are replaced at each training step, decay rate establishes how much influence the previous utility of a neuron has in proportion to the newly computed value, and maturity threshold controls how long the reset neurons should not be reset again.

We expect learning rate to be the most influential parameter for managing the stability-plasticity trade-off, as it directly controls how adaptive the model is: increasing learning rate values should result in increased plasticity and faster forgetting. Among the algorithm-specific parameters, we theorize that the replacement rate should be critical for managing the stability-plasticity trade-off, as it influences how much previously acquired information is lost versus how much new variance is introduced into the network. We hypothesize that lower replacement rate values should notably reduce forgetting, as less information is lost due to neuron resetting.

3.2 Baselines

Four baseline algorithms were used to compare their effect on catastrophic forgetting with that of Continual Backpropagation.

1. **Backpropagation (standard SGD)**
   Backpropagation with stochastic gradient descent updates model parameters by computing the gradient of the loss function and applying it to the weights. It is the most standard offline NN training algorithm, and it does not include any mechanisms to retain previous knowledge, making it prone to forgetting. Additionally, Dohare et al. [1] showed that backpropagation displays high loss of plasticity.
2. **L2 regularization**
   L2 regularization is an extension of standard SGD, however it penalizes large weight magnitudes by adding the squared L2 norm of the weight vector to the loss function. This encourages the network to maintain smaller weight values which, in offline learning setting, usually prevents overfitting and improves generalization. In continual learning settings, L2 regularization can slightly mitigate forgetting by discouraging drastic changes to the learned parameters, though it does not explicitly preserve past knowledge. In Dohare et al. [1] work, L2 Regularization was shown to perform significantly better than regular backpropagation in terms of model's plasticity. With L2 Regularization, the total loss becomes:
   $$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{task}} + \lambda \|\theta\|_2^2,$$
   where θ is the weight vector, and λ controls the regularization strength.
3. **Adam optimizer** [15]
   Adam is an adaptive gradient-based optimizer that adjusts learning rates of BP, based on the first and second moments of past gradients. In the offline learning settings, it is an extremely popular method, which often results in faster convergence and improved performance. However, Adam does not inherently include any mechanisms to prevent catastrophic forgetting, as it optimizes purely for immediate task performance. While Dohare et al. [1] demonstrated that Adam introduces significant loss of plasticity, we theorize that the plasticity results could have been unrepresentative, as the used learning rate was very high for the simulated setting.
4. **Shrink and Perturb** [6]
   Shrink and Perturb is a continual learning technique, conceptually similar to L2 Regularization. After training on a single batch of data, the model's weights are marginally scaled down, and then small random noise is added. This process helps to maintain network plasticity by preventing the weights from becoming too rigid, thus mitigating loss of plasticity. Shrink and Perturb was shown to reduce loss of plasticity very close to the level of CBP by Dohare et al. [1]. Formally, the weight update after task completion is:
   $$\theta \leftarrow (1-\lambda)\cdot\theta + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2),$$
   where λ is the shrink factor controlling the amount of weight scaling, and ϵ is Gaussian noise with variance σ^2 added to encourage parameter diversity. We expected this method to successfully balance stability and plasticity, as the information contained in weights throughout learning is never completely discarded.

3.3 Variants of Continual Backpropagation

In this section, we present three modifications to the original Continual Backpropagation algorithm. We hypothesize that all three variants of CBP can effectively mitigate forgetting while preserving the model's plasticity.

1. **Noise injection** [26].
   In the standard version of CBP, when a low-utility neuron is reinitialized, its incoming weights are reset to random values drawn from a predefined distribution, while its outgoing weights are set to zero. This procedure results in the complete loss of any information previously encoded in the neuron's weights, which may still be relevant to past or even current tasks.
   Urbonavičiūtė et al. [26] proposed a variant of this neuron resetting process, which uses noise injection combined with weight rescaling instead of full reinitialization. They demonstrate that noise injection variant of CBP maintains a level of plasticity comparable to standard reinitialization. However, for the proposed algorithm, the weights of a reset neuron are not completely discarded, therefore, a proportion of the its prior information is retained. Based on this, we hypothesize that noise injection can serve as an effective mechanism for reactivating low-utility neurons without entirely discarding potentially useful representations from earlier tasks.
   In this variant of CBP, both the incoming and outgoing weights (θ) of the selected neurons are updated according to the following rule:
   $$\theta \leftarrow (1-\lambda)\cdot\theta + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2),$$
   where λ is the shrink factor and σ^2 is the variance of the injected Gaussian noise.
2. **Layer-specific replacement** [27].
   Jučas et al. [27] has shown that reinitializing the neurons exclusively in the first hidden layer for CBP algorithm results in close to equivalent performance compared to reinitializing across all layers. However, in the proposed version of CBP, significantly less information is lost, as weights in deeper layers are completely preserved. This suggests that limiting reinitialization only to the first hidden layer may help retain information about previous tasks, and thus reduce forgetting.
3. **Partial neuron replacement**.
   We hypothesize that reinitializing all incoming and outgoing weights of a neuron may not be necessary to restore its plasticity. Instead, it may be sufficient to reinitialize only a fraction of the incoming and outgoing weights of the neurons that are chosen to be reset by CBP. To achieve this, we introduce a ratio $r \in [0;1]$, which defines the probability that each individual incoming or outgoing weight of a neuron, selected for reinitialization, is reset. In particular, with probability r, a given weight is either reinitialized using a random value drawn from the standard initialization distribution, or it remains unchanged. We expect this approach to introduce enough randomness to preserve model's plasticity, while maintaining more information from previous tasks, compared to full reinitialization of a neuron.

4 Experimental Setup and Results

In this section, we introduce the experimental setup used to evaluate CBP and the baseline algorithms, define the metrics that quantify stability and plasticity, and present the results.

4.1 Experimental Setup: Online Permuted MNIST Benchmark

MNIST [28] is a widely used, publicly available dataset consisting of handwritten digit images (0–9). We use this dataset to simulate a continual learning setting, in which the model is sequentially exposed to a series of tasks. In each task, the model is trained on the full MNIST dataset (10,000 grayscale images of size 28×28), where the pixels of all images for that task are permuted using a fixed random permutation (an example is shown in Figure 1). Once the model is trained on the entire dataset for a given task, a new random pixel permutation is applied to all the images, and the model continues training on the newly permuted dataset, representing the next task. Furthermore, the batch size for all of the experiments is set to 1, which means that the model trains on each image one by one. We call this benchmark Online Permuted MNIST (OPMNIST).

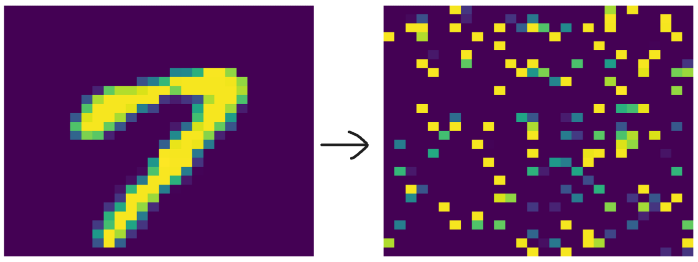


Figure 1: An illustration of the pixel permutation process used in the OPMNIST benchmark. The left image shows an original MNIST digit. The right image shows the same digit after a fixed reordering (permutation) has been applied to its pixels. In OPMNIST, all images within the same task share the same pixel permutation, which is randomly generated for each task.

A simple MLP architecture was chosen as a classifier, with three fully-connected hidden layers, each containing 400 neurons, and ReLu activation function. Note that all tasks within the OPMNIST are considered to be of the same level of difficulty, since the classification model that we use only comprises of simple fully connected layers, thus no spacial representations of the pixels are taken into account.

Furthermore, for each experiment, we evaluated the algorithms on two different forgetting scenarios.

1. **Initial exposure recall phase**. The model is initially trained sequentially on 100 distinct tasks, all of which it encounters for the very **first time**. This phase is meant to evaluate model's internal dynamics and remembering capabilities once it faces completely new data, while the effects of plasticity loss have not yet manifested.
2. **Recurrent task recall phase**. The model is then trained on an additional 2400 tasks. Every 100th task in this phase is a repetition of the very first task from the initial phase. In this way, the experiment is split to 25 *periods* . This phase is meant to evaluate how well different algorithms can recover and maintain knowledge about previously learned and then reintroduced information.

If not stated otherwise, for all of the tested algorithms we used a learning rate of $\alpha = 0.003$, while the default values of CBP hyperparameters were $\rho = 10^{-5}, \eta = 0.99, m = 100$. The exact hyperparameter values used for all the baseline algorithms are detailed in Appendix B.1.

Lastly, for each experiment, seven runs were performed and the average performance and standard error were reported in the plots. All of the experiments were run on Delft AI Cluster [29].

4.2 Evaluation Metrics

In this section, we describe a set of metrics for evaluating plasticity, forgetting, stability-plasticity trade-off and the internal dynamics of the models.

Metrics for Evaluating Plasticity

The most commonly used metric for evaluating plasticity is the evolution of model's performance, typically measured through accuracy. Elsayed and Mahmood [24] demonstrated that long-term accuracy is highly correlated with model's actual adaptability; moreover, accuracy directly reflects the algorithm's effectiveness at solving the target task. Therefore, we evaluated the plasticity of a model the following way: after each task, the resulting model is again evaluated on the data of that task, and the accuracy is computed. A decline in this accuracy over the course of learning new tasks of the same difficulty indicates loss of plasticity. Therefore, to summarize the model's ability to maintain adaptability throughout training, we define the plasticity as the average accuracy over the **final** 50 tasks of the experiment. Specifically, since all experiments run for 2500 tasks, plasticity is computed as the average accuracy across tasks 2451 to 2500.

Metrics for Evaluating Forgetting

To evaluate **catastrophic forgetting**, we use different metrics for the two training phases, that are introduced in Section 4.1.

1. **Evaluating initial exposure recall phase**. After training on the initial 40 tasks, the model's accuracy is evaluated on the full combined data of these tasks. This measures how well the model retains information from tasks it has only seen once. Higher accuracy indicates better retention. We call this metric **initial recall accuracy**.
2. **Evaluating recurrent task recall phase**. To evaluate forgetting for this phase, we use two complementary metrics.
   (a) **Recurrent accuracy curve**. After each task, accuracy on the very first task's (task 1) data is measured. In this way, we can track how well the model remembers a particular data distribution throughout training. Due to the nature of the recurrent task recall phase, accuracy on the initial task is expected to increase significantly every 100 tasks (a total of 25 times), as the initial data is reintroduced at this interval. The purpose of this metric is to track model's memory retention **evolution** as it is periodically re-exposed to previously seen data.
   (b) **Memory retention duration**. Given a *recurrent accuracy curve*, the number of consecutive new tasks that are processed before accuracy on the initial task's data falls below 30% is computed. This is measured starting from the most recent reintroduction of the initial task, during the final *period* of the experiment (i. e. tasks 2401-2500). Higher values of this metric indicate slower forgetting. The purpose of this metric is to quantify how effectively the model consolidates periodically reintroduced information.

Metrics for Evaluating Models' Internal Dynamics

Furthermore, in order to gain intuition why certain algorithms result in increased forgetting, we introduced a set of metrics to analyze the internal dynamics of CBP and the baseline algorithms.

1. **Weight Drift**. For each hidden layer, normalized L2 distance between the model's current weight vector ($\mathbf{w}_{\text{current}}$), and the weight vector obtained after being most recently trained on the initial task ($\mathbf{w}_{\text{initial}}$), is computed:

$$\Delta w = \frac{\|\mathbf{w}_{\text{initial}} - \mathbf{w}_{\text{current}}\|_2}{\sqrt{N}},$$

where N is the number of neurons within the layer. The total weight drift is defined as the average weight drift across all hidden layers. A high weight drift may indicate that the model's parameters have deviated significantly from their original configuration, which can be associated with increased forgetting of earlier tasks [7].

2. **Activation Drift**. Increased forgetting in CL is linked to changes in model's internal representations [3]. As the model learns new tasks, the activations in hidden layers shift away from those formed during earlier tasks, which can potentially discard previously acquired knowledge: higher activation drift may imply increased forgetting. To quantify this representational drift, we compare hidden layer activations, obtained after training on the very first task of the experiment, to those observed during later tasks, using two different metrics:

   i. Centered Kernel Alignment (CKA) [30]. It is a similarity metric used to compare activation representations between neural networks or between different stages of training within the same network. In this research, CKA is used to quantify how much the internal representations of a neural network change over time.

   $$\text{CKA}(X, Y) = \frac{\|Y^\top X\|_F^2}{\|X^\top X\|_F \cdot \|Y^\top Y\|_F},$$

   where X and Y are the activation matrices of the same hidden layer, extracted at different points of the training. We define final CKA value as an average CKA over all hidden layers of the network.

   ii. Procrustes distance. It quantifies the geometric similarity between activation patterns by aligning two matrices through an optimal rotation and scaling. It is a reliable and widely used baseline for measuring activation drift [31].

   $$\text{PrSim}(X, Y) = \frac{\text{tr}(\Sigma_{X,Y})}{\|X\|_F \cdot \|Y\|_F},$$

   where X and Y are the activation matrices of the same hidden layer, extracted at different points of the training, and Σ is their covariance matrix. We define final Procrustes distance value as an average Procrustes distance over all hidden layers of the network.

   While CKA has been demonstrated to provide unreliable results [31], in all of the experiments we found the two metrics to follow almost identical trends in terms of activation drift (see Appendix B.5).

4.3 Results

In this section, we present results on the OPMNIST benchmark across different experiment groups. Specifically, we assess forgetting in CBP and baseline algorithms, analyze CBP's performance under various hyperparameter settings, examine the effectiveness of proposed CBP variants, and study the internal dynamics of the models.

The main findings are summarized in Figure 2a for the initial exposure recall phase and in Figure 2b for the recurrent task recall phase. In particular, both figures visualize the stability-plasticity trade-off for different groups of experiments that are presented in this section. In both visualizations, the horizontal axis corresponds to the plasticity metric, defined in Section 4.2—higher values of this metric indicate increased adaptability. The vertical axis corresponds to the *initial recall accuracy* metric for Figure 2a, and the *memory retention duration* metric for Figure 2b. Higher values of both metrics indicate increased memory retention.

Evaluating Forgetting on Plasticity-Optimized Algorithm Versions

Initially, we evaluated the forgetting on CBP and the baseline algorithms: regular backpropagation, L2 regularization, Shrink and Perturb and Adam. For all algorithms, excluding Adam, we used the hyperparameters optimized for plasticity for OPMNIST benchmark, as reported by Dohare et al. [1]. The exact parameter configurations for all algorithms and extended remarks can be found in Appendix B.1. For the Adam baseline, we report results using a more appropriate learning rate of $\alpha = 0.001$, as the default value of $\alpha = 0.003$ lead to consistently degraded and unrepresentative performance.

As shown in Figure 2a and Figure 2b, standard CBP exhibited the highest degree of forgetting among all evaluated baselines, both during the initial exposure recall and the recurrent task recall phases respectively. Furthermore, CBP failed to retain information of the first permutation, even after 25 reintroductions - no improvement in incorporating periodically reintroduced information was observed (see Figure 3). In contrast, Adam and standard backpropagation exhibited lower forgetting than CBP in both experimental settings, although they were significantly less plastic. Shrink and Perturb, as well as L2 regularization, performed comparably to each other, achieving a favorable balance between plasticity and forgetting for the initial exposure recall experiment. However, they were less effective than Adam and regular backpropagation at incorporating periodically reintroduced information over time (see Figure 3).

Evaluating Hyperparameter Influence on Forgetting

Furthermore, we evaluated how different hyperparameters influence forgetting for CBP algorithm. In particular, we tested CBP with different values of the **replacement rate** (ρ), **decay rate** (η), **maturity threshold** (m), and **learning rate** (α) parameters. In each experiment, only one hyperparameter was varied, while the others were held constant at their default values. Figure 2 summarizes replacement rate and decay rate parameters' influence on both stability and plasticity, while the extended results and analysis for the rest of the hyperparameters can be found in Appendix B.2.

First, it is evident that decreasing replacement rate values mitigate forgetting for both initial exposure recall and recurrent task recall settings (see Figure 2). This supports our hypothesis that the less frequently neurons are replaced, the less information regarding past tasks is lost. However, results obtained from periodic recall experiment indicate that even with low replacement rate values, the model is not able to efficiently improve upon the periodically reintroduced information (see Figure 4).

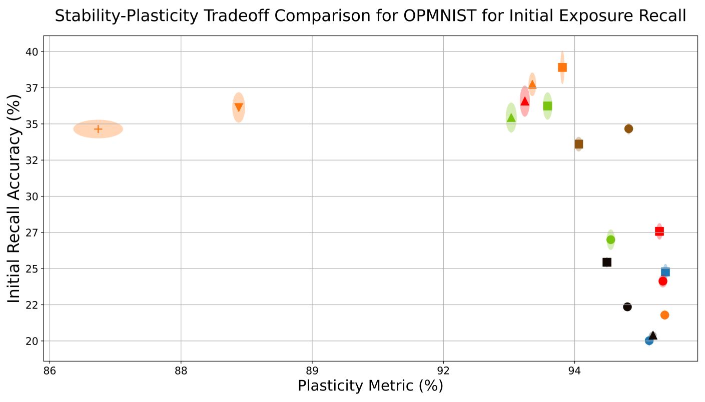
Interestingly, our experiments showed that lower decay rate values significantly reduced forgetting in the recurrent task recall setting, which enabled the model to effectively consolidate information from periodically reintroduced tasks (see Figure 4). Since the decay rate affects only the selection of neurons for reinitialization, these results imply that forgetting in CBP may not rise solely from the act of reinitializing neurons, but rather from reinitializing neurons that still encode useful information. This suggests potential for improving the utility estimation function to more effectively preserve past task-relevant representations.

Lastly, as shown in Appendix B.2, even with efficient values of the replacement rate parameter, the maturity threshold had negligible effect on both plasticity and forgetting. The learning rate, on the other hand, had significant impact on the stability-plasticity trade-off: low values of learning rate drastically increased the stability, however introduced loss of plasticity.

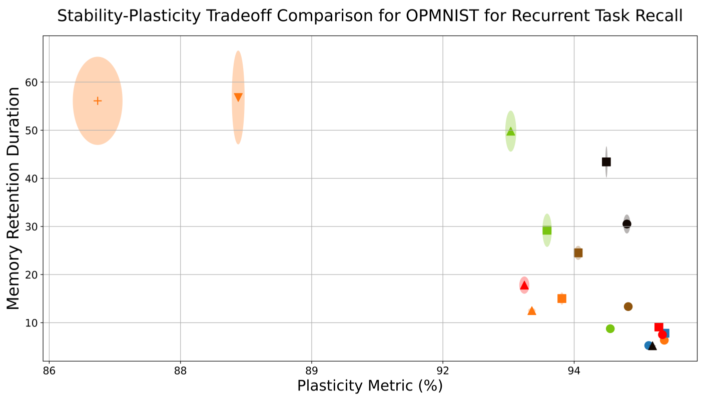
Evaluating Variants of Continual Backpropagation

We evaluated three variations of CBP, expected to better exploit the stability-plasticity tradeoff: **noise injection**, **layer-specific replacement** and **partial neuron replacement**. In this section, we show that all three variations of the algorithm reduce forgetting while only minimally losing plasticity.

Noise injection We evaluated the noise injection variant of CBP using different values for the shrink rate λ and the standard deviation σ^2 of the noise component, across various replacement rate



(a) Initial exposure recall experiment trade-off comparison



(b) Recurrent task recall experiment trade-off comparison

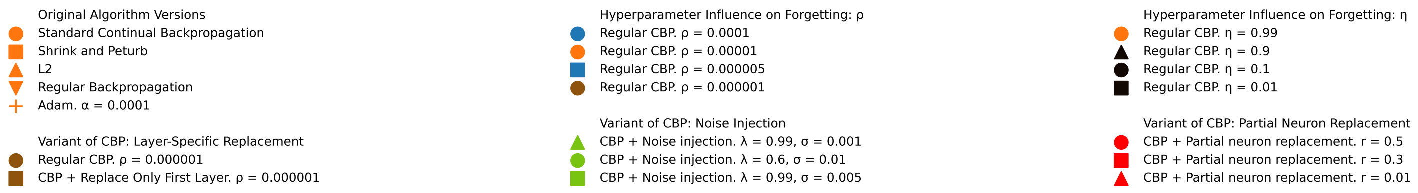


Figure 2: Stability-plasticity trade-off comparison for different versions of algorithms, for two forgetting scenarios: initial exposure recall (a) and recurrent task recall (b). The markers indicate the average value of the metric, and the shaded area corresponds to the standard error. As described in Section 4.2, the plasticity metric for both plots is the average accuracy of the final 50 tasks of the whole experiment. The forgetting metric for a corresponds to *initial recall accuracy*, while forgetting metric for 2b corresponds to *memory retention duration*. In both figures, higher values of the forgetting metric indicate increased retention, and higher values of the plasticity metric indicate increased plasticity.

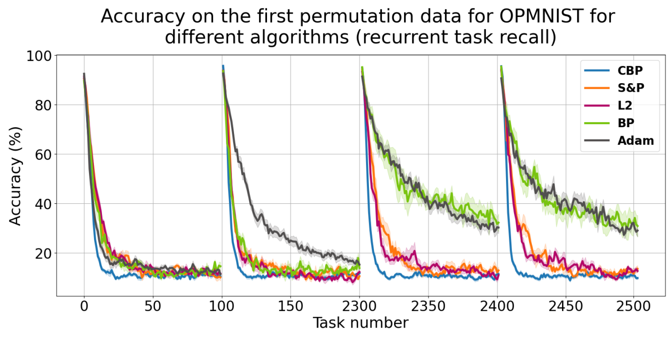


Figure 3: Comparison of *recurrent accuracy curves* of five algorithms (Continual Backpropagation (CBP), regular backpropagation (BP), L2 Regularization (L2), Adam and Shrink and Perturb (S&P)), for the recurrent task recall phase. The initial task is reintroduced every 100 tasks for 25 times. For simplicity, only the initial two and the final two periods are shown. Results are averaged over seven independent runs; solid lines represent the mean accuracy, and shaded areas denote the standard error.

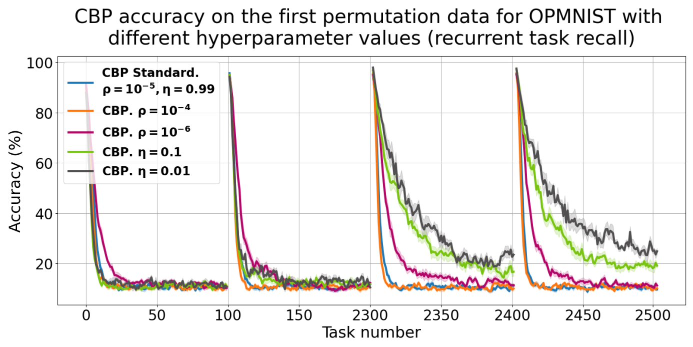


Figure 4: Comparison of *recurrent accuracy curves* of CBP with different values of the **replacement rate** and **decay rate** parameters, for the recurrent task recall phase. The initial task is reintroduced every 100 tasks for 25 times. For simplicity, only the initial two and the final two periods are shown. Results are averaged over seven independent runs; solid lines represent the mean accuracy, and shaded areas denote the standard error.

settings. Our results indicate that low noise magnitude combined with relatively high shrink rates allows models to retain memory substantially better than regular CBP for both forgetting evaluation scenarios (see Figures 2 and 5). In particular, Figure 2 shows that the noise injection variant was one of the few methods that achieved high memory retention in both the initial exposure recall and recurrent task recall settings, relative to other algorithms. However, we also found that this comes at the cost of a noticeable loss in plasticity. Finally, as shown in Figure 2b, noise injection with $\lambda = 0.99$ and $\sigma^2 = 0.001$ was extremely effective in improving retention for periodically reintroduced information.

A more detailed analysis of how different parameters of noise injection variant affect stability-plasticity trade-off is presented in Appendix B.3.

Layer-specific replacement We evaluated the stability and plasticity across varying replacement rate values for the proposed strategy, in comparison to original CBP algorithm. However, we found that differences in forgetting only become apparent at low replacement rate values. Specifically, as illustrated in Figure 6, for a replacement rate of $\rho = 10^{-6}$, the proposed strategy demonstrated improved retention of periodically reintroduced information, whereas the standard CBP approach remained unaffected. However, the improved memory retention for the recurrent task recall phase came at the cost of a slight reduction in plasticity (see Figure 2b). Moreover, the difference in forgetting for the initial exposure scenario was negligible (see Figure 2a).

Partial neuron replacement Lastly, we hypothesized that it may not be necessary to reinitialize all of a neuron's incoming and outgoing weights to restore its plasticity. Our findings support this hypothesis: we showed that reinitializing up to $r = 0.5$ of a neuron's weights fully preserves model's plasticity while slightly reducing forgetting in the initial exposure recall setting (see Figure 2a). However, this reduction in memory loss was only observed in the initial exposure recall setting, since for $r > 0.5$, no clear improvement in memory retention was seen, when information is periodically reintroduced (see Figure 2b). Nevertheless, reinitializing as little as 1% of the weights noticeably reduced forgetting in the recurrent task recall experiment (see Figure 6), though it came

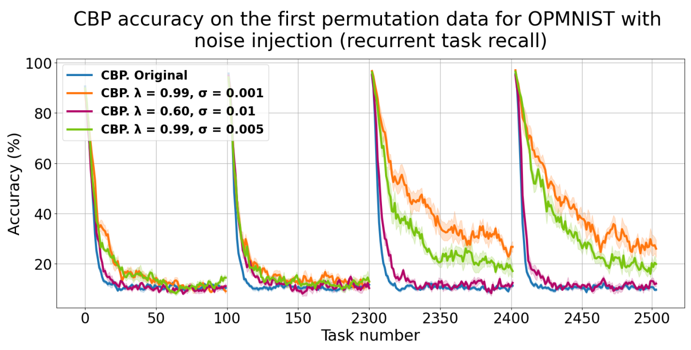


Figure 5: Comparison of *recurrent accuracy curves* of variant of CBP with **noise injection**, with different values of the shrink rate and noise magnitude parameters, for the recurrent task recall phase. The initial task is reintroduced every 100 tasks for 25 times. For simplicity, only the initial two and the final two periods are shown. Results are averaged over seven independent runs; solid lines represent the mean accuracy, and shaded areas denote the standard error.

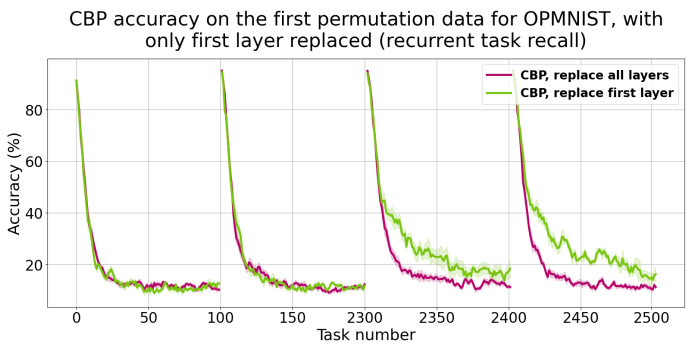


Figure 6: Comparison of *recurrent accuracy curves* of variant of CBP with **layer-specifc replacement**, with $\rho = 10^{-6}$, for the recurrent task recall phase. The initial task is reintroduced every 100 tasks for 25 times. For simplicity, only the initial two and the final two periods are shown. Results are averaged over seven independent runs; solid lines represent the mean accuracy, and shaded areas denote the standard error.

at the cost of reduced plasticity. Appendix B.4 includes an extended analysis of this variant, covering results for a wider range of r values.

Evaluating Models' Internal Dynamics

We evaluated the internal behavior of the networks throughout the forgetting process. In particular, we observed that activation drift, measured both using Procrustes distance and CKA, strongly correlates with forgetting: higher drift values reliably indicated increased forgetting. Figure 8 shows how activation drift, measured via the Procrustes distance (see Section 4.2), evolved during the recurrent task recall experiment across different algorithms. The forgetting of the same experiment for corresponding algorithms is evaluated in Figure 3. A very strong alignment between these metrics is visible-algorithms that resulted in increased activation drift also displayed low memory retention. However, we did not observe consistent correlation between forgetting and weight drift of a model: detailed results of weight drift and further evaluation of activation drift metrics can be found in Appendix B.5. Nevertheless, the strong alignment between activation drift metrics highlights the potential of particular drift-based metrics as reliable indicators of forgetting.

5 Discussion

Our main observation is that CBP consistently performed poorly in terms of stability compared to all four baselines. This finding was consistent across both the initial exposure recall and the recurrent task recall experiment types. Despite its strong plasticity, CBP failed to retain task-relevant knowledge over time, and even

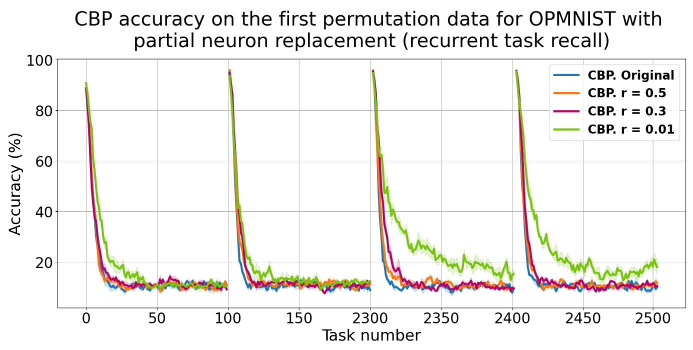


Figure 7: Comparison of *recurrent accuracy curves* of variant of CBP with **partial neuron replacement**, with different values of weights replacement ratio (r) parameter, for the recurrent task recall phase. The initial task is reintroduced every 100 tasks for 25 times. For simplicity, only the initial two and the final two periods are shown. Results are averaged over seven independent runs; solid lines represent the mean accuracy, and shaded areas denote the standard error.

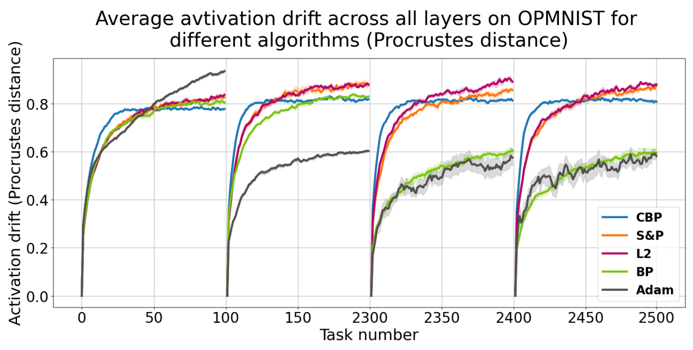


Figure 8: Evolution of activation drift, measured using Procrustes distance. The activation drift is computed relative to the features obtained after the most recent occurrence of the very first task, for five algorithms (Conitinual Backpropagation (CBP), regular backpropagation (BP), L2 Regularization (L2), Adam and Shrink and Perturb (S&P)), for the recurrent task recall phase. Lower values of the metric imply smaller activation drift. The initial task is reintroduced every 100 tasks for 25 times. For simplicity, only the initial two and the final two periods are shown. Results are averaged over seven independent runs; solid lines represent the mean accuracy, and shaded areas denote the standard error.

repeated exposure to previously learned tasks did not effectively restore the forgotten information. In fact, even algorithms such as standard SGD and Adam (both known to suffer from limited stability) showed better memory retention for OPMNIST experiment. This highlights that while CBP effectively addresses one side of the stability–plasticity dilemma (plasticity), it severely compromises the other (stability), making it unsuitable for tasks where long-term memory is important.

We believe that the reason for the poor stability observed in CBP lies within the core of the algorithm: its neuron resetting mechanism. Our results suggest that the utility-based neuron reinitialization process may be flawed in two fundamental ways. First, it may be too aggressive—discarding neurons that, while estimated to have low utility for current task, still encode representations that are useful for retaining past knowledge. As a result, previous task-relevant features may be lost, which degrades performance on past data. Second, the mechanism may be too imprecise in identifying truly necessary neurons. The utility function may not capture all aspects of a neuron's contribution, leading to preservation of the unimportant neurons, while reinitializing ones that are relevant for retaining past information. Together, these limitations can explain CBP's inability to retain long-term knowledge, despite its effectiveness at maintaining plasticity.

Our results obtained from experiments with the decay rate parameter of the utility function provide strong support for the second hypothesis: CBP's forgetting partially results from poor selection of neurons for reinitialization. Specifically, we found that lowering the decay rate, which increases the influence of past utility scores in the current utility calculation, lead to a substantial reduction in forgetting during the recurrent task recall phase. Since the decay rate only affects the selection of neurons, and not the reinitialization process itself, this suggests that forgetting in CBP may rise from misidentifying which neurons are actually important to preserve. More precisely, neurons that are still useful for retaining earlier knowledge may be marked as low-utility. This suggest that there exists room for developing more effective utility functions that capture past task relevance. For example, incorporating historical gradient or drift-based information could improve neuron selection and thus reduce forgetting without mitigating plasticity.

Furthermore, our analysis of CBP variants offers promising room for mitigating forgetting while retaining plasticity. We examined three modifications to CBP, all of which were effective in reducing forgetting in specific settings, with minimal or no loss of plasticity. All three evaluated approaches work by lowering the amount of randomness introduced to the model, and we find that these less destructive methods still allow the model to remain adaptable. This indicates that CBP algorithm could potentially be even further improved by searching for ways to less destructively reset low utility neurons; that could be done by either combining our proposed strategies or by exploring new methods.

We additionally found a strong correlation between forgetting and certain internal model dynamics. In particular, activation drift closely followed the forgetting trends, especially in the recurrent task recall scenario. This supports a well established idea that the representational changes over time are the underlying reason of forgetting. Our results also indicate that feature drift metrics may be useful in practice for monitoring and predicting when a model is likely to forget earlier tasks, and, as stated before, could potentially be utilized to improve the utility estimation. Finally, we found that weight drift, computed using L2 distance, is not a reliable indicator of forgetting.

Lastly, while our findings provide valuable insight into the behavior of CBP and its variants, our conclusions are drawn from experiments conducted exclusively on the Online Permuted MNIST benchmark using a simple multi-layer perceptron architecture for our models. This setting, although well-controlled and widely used in continual learning research, may not capture the full complexity of real world tasks. In particular, the effect of CBP on forgetting on high dimensional input data, such as high resolution images or natural language, remains unexplored. Moreover, our evaluation focuses primarily on classification tasks; the effects of CBP on other learning types, such as reinforcement learning, are not determined. Therefore, we advise to responsibly interpret and use our results in more complex continual learning scenarios.

6 Conclusions and Future Work

The goal of this research was to evaluate catastrophic forgetting in models trained using the Continual Backpropagation algorithm. To achieve this, we established a continual learning setting using MNIST dataset. We defined and evaluated forgetting under two distinct scenarios: (1) initial exposure recall, where forgetting is assessed on data encountered only once and for the first time, and (2) recurrent task recall, where certain data is reintroduced multiple times. Different metrics were used for these two scenarios to best capture their dynamics.

We compared CBP with four baseline algorithms: Shrink and Perturb, Adam, L2 regularization, and standard backpropagation. Across both initial exposure recall and recurrent task recall scenarios, CBP consistently exhibited significantly higher forgetting than all the baselines. Notably, CBP failed to benefit from periodic re-exposure to tasks, in contrast to the baselines, which showed improved retention over time.

We further investigated how CBP's performance varies with changes in its hyperparameters: learning rate, replacement rate, decay rate and maturity threshold. Our results indicate that the learning rate, decay rate, and replacement rate significantly influence the stability-plasticity trade-off, whereas the maturity threshold showed negligible impact. In particular, our analysis of the decay rate highlights the potential for improving the utility estimation function to better preserve past task-relevant representations.

Three CBP variations were also examined—noise injection, layer-specific replacement and partial neuron replacement—hypothesizing that they could mitigate forgetting while maintaining high plasticity. All three methods were effective in reducing forgetting in specific settings. Remarkably, we showed that replacing up to 50% of a neuron's incoming and outgoing weights is sufficient to restore its plasticity without without compromising performance.

Moreover, we analyzed the internal dynamics of the networks, focusing on activation and weight drift in hidden layers over the course of learning. Activation drift proved to be a strong indicator of forgetting: models with higher feature drift exhibited faster forgetting, suggesting it as a useful indicator for model's stability.

Finally, we identify several promising directions for future investigation. First, our results suggest that the current utility function in CBP may not optimally preserve task-relevant information. Developing a more effective utility estimation method—potentially one that incorporates information from gradient history or drift patterns—could help reduce forgetting without significantly compromising plasticity. Furthermore, several variants of CBP, such as noise injection and partial neuron replacement, performed comparably well to the original version in terms of plasticity while also reducing forgetting. These findings indicate that there is room for systematic improvement of the CBP algorithm, either through combining these strategies or by exploring new approaches. Finally, future work could expand the evaluation of catastrophic forgetting to include more complex CL benchmarks, such as continual reinforcement learning, since our current conclusions are based solely on a single benchmark.

7 Responsible Research

7.1 Ethics

Our research does not involve human subjects, personal data, or any sensitive or confidential information. All experiments were conducted exclusively on publicly available datasets, such as MNIST, which do not raise direct ethical concerns.

However, CL methods evaluated and proposed in this study can potentially be applied in high-stake domains, such as medical diagnostics or autonomous driving. In these contexts, the reliability of the algorithms, particularly their ability to retain information over time, might be of significant importance. It is therefore essential that any system that uses the results of our research undergoes thorough testing before deployment.

7.2 Reproducibility and Replication of Results

Ensuring the replication of our findings has been a relevant part of the process throughout this project. All experiments were implemented using publicly available Python libraries, primarily leveraging PyTorch[1]. The codebase is a modified and extended version of the original implementation by Dohare et al. [1], structured to allow users to reproduce and extend experiments with minimal effort.

To support replication, every experiment can be executed via straightforward configuration file following a standardized and extendable format. Instructions on how to prepare and use these configuration files are included in the repository's documentation, which allows users to reproduce exact experimental setups and also adjust parameters for further exploration.

[1]pytorch.org

The entire codebase, which includes implementations of baseline algorithms, the Continual Backpropagation method, proposed modifications, and all custom evaluation metrics (such as weight drift and activation drift), is publicly available on GitHub[2]. The repository contains setup instructions, dependency requirements, and sample configuration files used to run experiments, described in this work.

Additionally, all hyperparameters used in the experiments, across both baselines and our proposed CBP variants, are reported in detail in Section 4.1 and Appendix A, ensuring transparency and allowing replication of results.

However, it must be mentioned that while we acknowledge the value of using fixed random seeds for strict reproducibility, we chose not to enforce them in our experiments. This decision was based on the observation that our results were highly consistent across multiple independent runs — each experiment was repeated seven times, and the variance in outcomes was minimal. This suggests that our findings are robust to initialization and random factors.

7.3 Usage of Large Language Models

In this research, the use of Large Language models was kept minimal. Copilot[3] was occasionally used for basic code generation, while Grammarly[4] and Overleaf[5] suggestions were used for minor language corrections.

ChatGPT[6], however, was used occasionally for code-related debugging, plot visualization-related questions and technical LaTex paper writing-related questions. The exact prompts and usage details are documented in Appendix A.

[2] https://github.com/jujucas21/research-project-repository
[3] copilot.microsoft.com
[4] grammarly.com
[5] overleaf.com
[6] chatgpt.com

References

[1] Shibhansh Dohare, J Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A Rupam Mahmood, and Richard S Sutton. Loss of plasticity in deep continual learning. *Nature*, 632(8026):768–774, 2024.

[2] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[3] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[4] Zaheer Abbas, Rosie Zhao, Joseph Modayil, Adam White, and Marlos C Machado. Loss of plasticity in continual deep reinforcement learning. In *Conference on lifelong learning agents*, pages 620–636. PMLR, 2023.

[5] Craig Atkinson, Brendan McCane, Lech Szymanski, and Anthony Robins. Pseudo-recursal: Solving the catastrophic forgetting problem in deep neural networks. *arXiv preprint arXiv:1802.03875*, 2018.

[6] Jordan Ash and Ryan P Adams. On warm-starting neural network training. *Advances in neural information processing systems*, 33:3884–3894, 2020.

[7] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[8] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

[9] 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)*, pages 139–154, 2018.

[10] Geoffrey E Hinton and David C Plaut. Using fast weights to deblur old memories. In *Proceedings of the ninth annual conference of the Cognitive Science Society*, pages 177–186, 1987.

[11] Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks. In *International Conference on Machine Learning*, pages 23190–23211. PMLR, 2023.

[12] Saurabh Kumar, Henrik Marklund, and Benjamin Van Roy. Maintaining plasticity in continual learning via regenerative regularization. *arXiv preprint arXiv:2308.11958*, 2023.

[13] Gail A Carpenter and Stephen Grossberg. Art 2: Self-organization of stable category recognition codes for analog input patterns. *Applied optics*, 26(23):4919–4930, 1987.

[14] Jacopo Silvestrin, Francisco S Melo, and Manuel Lopes. A comparative study of continual backpropagation. In *EPIA Conference on Artificial Intelligence*, pages 324–334. Springer, 2024.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

[17] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS

Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019.

[18] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017.

[19] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[20] Ragav Venkatesan, Hemanth Venkateswara, Sethuraman Panchanathan, and Baoxin Li. A strategy for an uncompromising incremental learner. *arXiv preprint arXiv:1705.00744*, 2017.

[21] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

[22] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International conference on machine learning*, pages 4548–4557. PMLR, 2018.

[23] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *International conference on artificial intelligence and statistics*, pages 3762–3773. PMLR, 2020.

[24] Mohamed Elsayed and A Rupam Mahmood. Addressing loss of plasticity and catastrophic forgetting in continual learning. *arXiv preprint arXiv:2404.00781*, 2024.

[25] Sanghwan Kim, Lorenzo Noci, Antonio Orvieto, and Thomas Hofmann. Achieving a better stability-plasticity trade-off via auxiliary networks in continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11930–11939, 2023.

[26] Urtė Urbonavičiūtė, Laurens Engwegen, and Wendelin Böhmer. Exploring alternatives to full neuron reset for maintaining plasticity in continual backpropagation. Manuscript in preparation, 2025.

[27] Augustinas Jučas, Laurens Engwegen, and Wendelin Böhmer. Layerwise perspective into continual backprop: Replacing the first layer is all you need. Manuscript in preparation, 2025.

[28] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

[29] Delft AI Cluster. The Delft AI Cluster (DAIC), RRID:SCR_025091, 2024. URL https://doc.daic.tudelft.nl/. https://doc.daic.tudelft.nl/.

[30] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR, 2019.

[31] Meagan Campol Haynes, Nessa Ryan, Mona Saleh, Abigail Ford Winkel, and Veronica Ades. Contraceptive knowledge assessment: validity and reliability of a novel contraceptive research tool. *Contraception*, 95(2):190–197, 2017.

A Use of Large Language Models

In this section, we provide main prompts that were used for querying LLMs. In particular, we used ChatGPT several times for code debugging purposes, LaTex-related quires and for plot visualizations. In order not to overwhelm the paper with code-related text, we simplify the prompts by only providing the human-language part of the queries. We now provide the three groups of queries that were used for ChatGPT, with examples of the questions that were asked.

1. Code debugging-related queries.
   (a) *"I get this error when fetching data from wandb:"*
   (b) *"In my history, I have been logging some metrics for the first 300 steps, and then stopped, while others were logged for additional 100 steps. Will the data frame contain the cut-out version of the metrics that I logged for longer? Because I get this error:"*
   (c) *"What does this piece of code do in PyTorch? I get this error:"* (most common query).
2. Graphical visualization-related question.
   (a) "*In matplotlib, how to make the legend split into two rows*"
   (b) "*How to plot the legend horizontally?*"
   (c) "*How to increase boldness in legend text?*"
3. Overleaf and Latex-related questions.
   - "*How to cite a paper that is not published*"
   - "*What is the difference between @article and @inproceedings in .bib files latex*"
   - "*Is it possible to set the height of a figure once the width is already set?*"

B Further Analysis of Results

B.1 Remarks on Baseline Algorithms

In Section 4.3, the results are presented for plasticity-optimized algorithms, based on the parameter settings reported by Dohare et al. [1]. We reproduced the results of that study under a reduced experimental setup, using a dataset of 10,000 samples instead of 60,000, and a smaller model architecture. Despite these changes, we found that the optimal hyperparameters for minimizing plasticity loss remained consistent across algorithms for a learning rate of $\alpha = 0.003$.

Specifically, the optimal regularization strength for L2 regularization was $\lambda = 0.0001$, while the best parameters for the Shrink and Perturb method were $\lambda = 0.0001$ and $\sigma^2 = 0.00001$. Continual Backpropagation achieved the highest plasticity with $m = 100$, $\rho = 10^{-5}$, and $\eta = 0.99$. As noted, all algorithms were trained with a learning rate of $\alpha = 0.003$.

However, we found that this learning rate was too high for the Adam optimizer. At $\alpha = 0.003$, Adam showed to have very minimal ability to learn in general. As shown in Figure 9, the models trained with this learning rate failed to properly learn initial task, achieving only around 70% accuracy at best. This performance declined rapidly over subsequent tasks due to reduced plasticity. However, slightly reducing the learning rate to $\alpha = 0.001$ resulted in significantly increased plasticity and forgetting (see Figure 9), suggesting that previous choice of learning rate was sub-optimal.

B.2 Extended Evaluation of Hyperparameter Influence for CBP

The standard hyperparameter values used in CBP were $\alpha = 0.003$, $m = 100$, $\rho = 10^{-5}$ and $\eta = 0.99$. Figures 11-14 illustrate how change in different hyperparameter values influence the stability-plasticity trade-off.

First, we observe that varying the replacement rate values results in consistent relative performance across both forgetting phases (see 11). This suggests that the influence of the replacement rate on the stability-plasticity trade-off manifests similarly across both forgetting scenarios. Furthermore, our results indicate

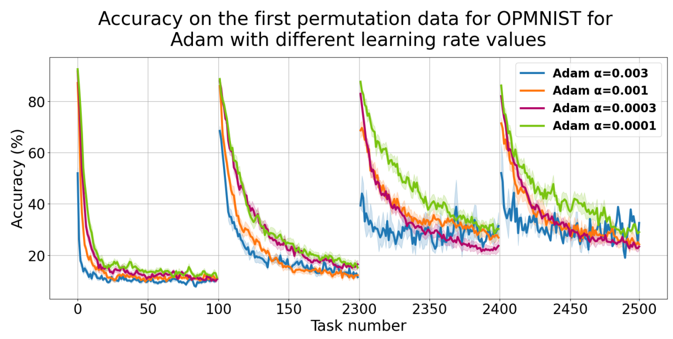


Figure 9: Comparison of *recurrent accuracy curves* of models trained with Adam with different learning rate values.

that lower values of the replacement rate parameter reduce forgetting.

Figure 12 shows that changing the replacement rate leads to consistent relative performance trends across both forgetting scenarios (initial exposure recall and recurrent task recall). This suggests that the effect of the replacement rate on the stability-plasticity trade-off is the same both when model is exposed to data for the first and, and when it is periodically introduced. Furthermore, the results clearly indicate that lower replacement rate values mitigate forgetting.

As hypothesized, low values of the learning rate parameter improve retention capabilities - this finding was consistent for both experimental phases (see Figures 10 and 13). However, while very low values of the learning rate parameter significantly decrease plasticity (for instance, $\alpha = 0.0001$), it is visible that for reasonably high values of the learning rate parameter (in our case $\alpha \in [0.01; 0.001]$), plasticity is maintained up to very similar level. Lastly, our results indicate that even with low learning rate values, the model is not able to improve upon periodically re-introduced information.

Lastly, maturity threshold was shown to have negligible impact for leveraging the stability-plasticity trade-off (see Figure 14). The obtained results do not provide distinctive insight between the performance of CBP trained with different values of the maturity threshold. Since maturity threshold's effect to CBP depends on what the replacement rate value is, we tested maturity threshold's influence on forgetting across various plasticity-optimal replacement rate values; however, the same conclusion was drawn from the obtained results.

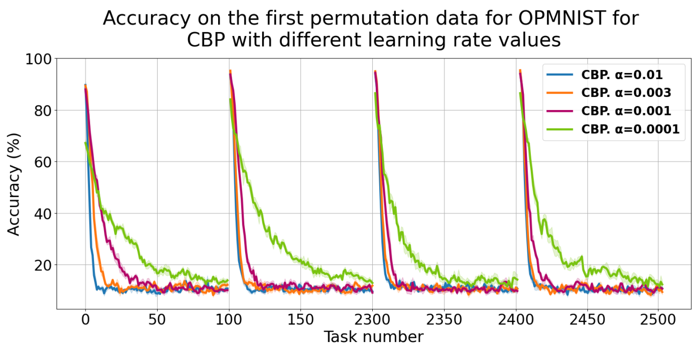


Figure 10: Comparison of *recurrent accuracy curves* of models trained with CBP with different learning rate values.

B.3 Extended Evaluation of Variant of CBP: Noise Injection

We evaluated how the forgetting for the recurrent task recall phase changes with different values of shrink rate (λ) and noise variance (σ^2) parameters. Results for varying shrink rate parameter with fixed $\sigma^2 = 0.001$ are provided in Figure 16, while changing noise variance influence on forgetting is visualized in Figure 15. It is evident from the plots that higher values of the shrink rate parameter decrease forgetting, while lower amount of noise added to the model results in less forgetting. These findings were robust across various values of fixed shrink rate and noise variance parameters.

Furthermore, we evaluated the stability-plasticity trade-off for both forgetting scenarios. This trade-off is visualized in Figure 17, and we used the same shrink rate and noise variance parameters as in the previous figures. It is clear that increasing values of the noise variance improve plasticity. However, change in shrink rate value for a fixed $\sigma^2 = 0.001$ does not have significant impact on plasticity.

B.4 Extended Evaluation of Variant of CBP: Partial Neuron Replacement

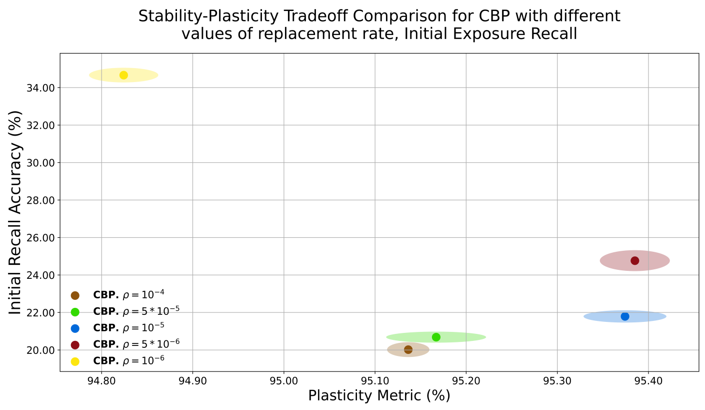
We evaluated the performance of the partial neuron replacement variant of CBP, using a an increased number of replacement ratio r values. Results for both the initial exposure recall and the recurrent task recall phases are provided in Figure 18. We find that the trade-off for different values of r is identical for both forgetting scenarios: higher r values indicate lower forgetting, but also reduce plasticity.

B.5 Extended Evaluation of Weight and Activation Drift

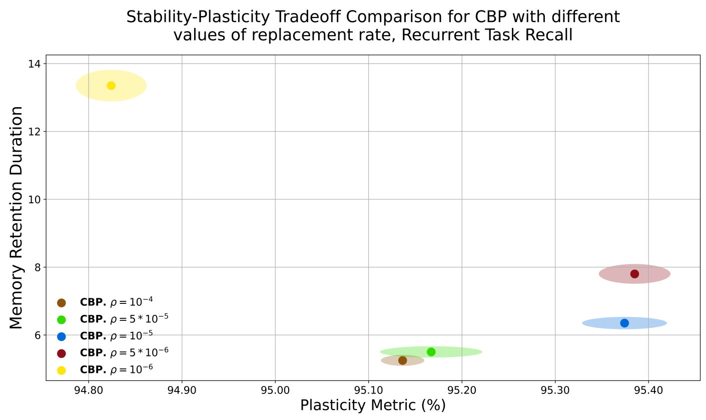
In this section, we present a more elaborate analysis of how activation and weight drifts affect forgetting. In particular, for several conducted experiments, we visualize their corresponding activation and weight drift curves, which provide insight into whether these metrics correlate. In order to simplify the visualization process, we only show the curves of the last *period* of the experiments (i. e. tasks 2401-2500). In this way, a very clear distinction between the activation, weight and forgetting curves of different algorithms is visible.

Figures 19, 20, and 21 clearly demonstrate that activation drift correlates strongly with forgetting trends. Both the Procrustes distance and CKA metrics produce consistent results across different experiment groups, and effectively capture the relative forgetting behavior of the algorithms. In particular, both feature drift metrics accurately reflect the ordering of algorithms in terms of how quickly they forget previous tasks. While we show only three groups of examples where feature drift correlates with forgetting, we actually observed this behaviour in all tested settings.

Interestingly, weight drift alone, measured using L2 distance, appears to be a weaker indicator, as it does not consistently align with the observed forgetting dynamics. For example, in Figure 20, weight drift of L2 and Shrink and Perturb algorithms is the smallest, however, these algorithms display higher forgetting than, for instance, BP. That is, however, an expected result, because the core idea beneath L2 and Shrink and Perturb is to minimize the magnitude of weights-therefore, the weights change less during training.

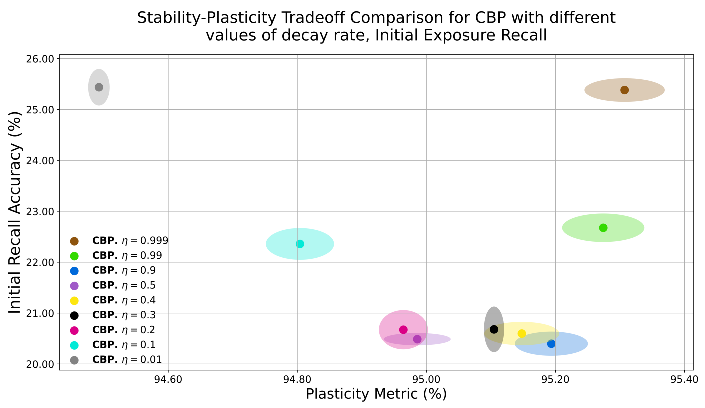


(a) Initial exposure recall experiment trade-off comparison

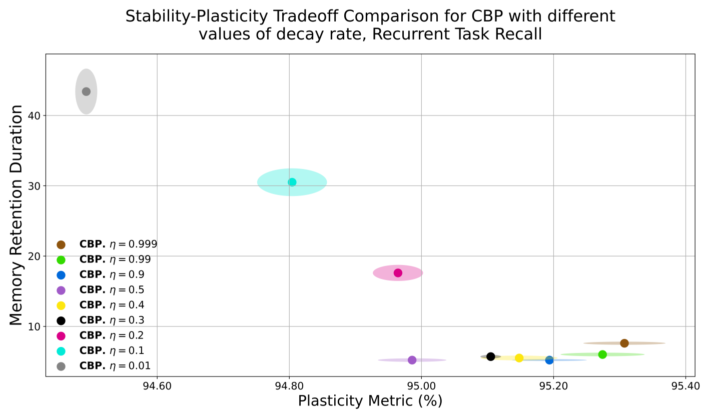


(b) Recurrent task recall experiment trade-off comparison

Figure 11: Stability-plasticity trade-off comparison for different replacement rate hyperparameter values of CBP. Other hyperparameters are set to their default values.

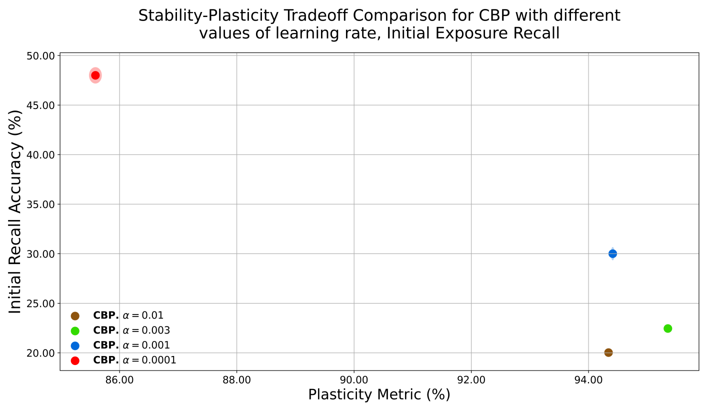


(a) Initial exposure recall experiment trade-off comparison

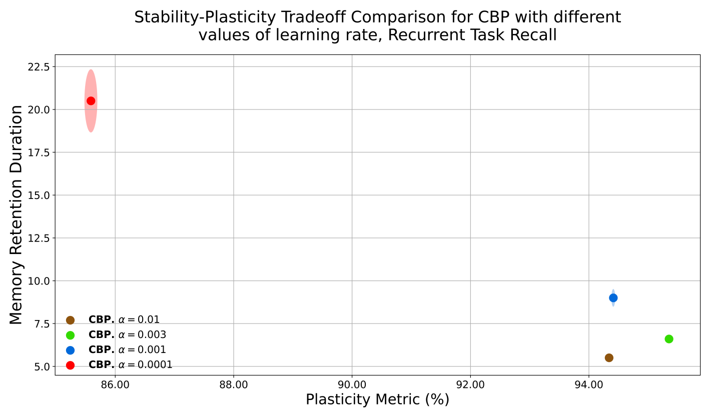


(b) Recurrent task recall experiment trade-off comparison

Figure 12: Stability-plasticity trade-off comparison for different decay rate hyperparameter values of CBP. Other hyperparameters are set to their default values.

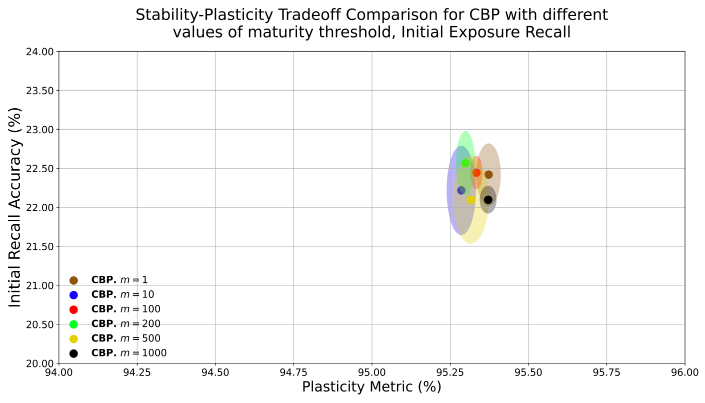


(a) Initial exposure recall experiment trade-off comparison

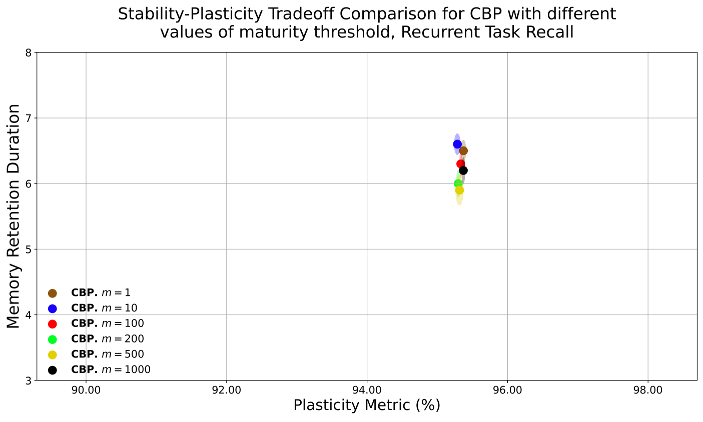


(b) Recurrent task recall experiment trade-off comparison

Figure 13: Stability-plasticity trade-off comparison for different learning rate hyperparameter values of CBP. Other hyperparameters are set to their default values.



(a) Initial exposure recall experiment trade-off comparison



(b) Recurrent task recall experiment trade-off comparison

Figure 14: Stability-plasticity trade-off comparison for different maturity threshold hyperparameter values of CBP. Other hyperparameters are set to their default values.

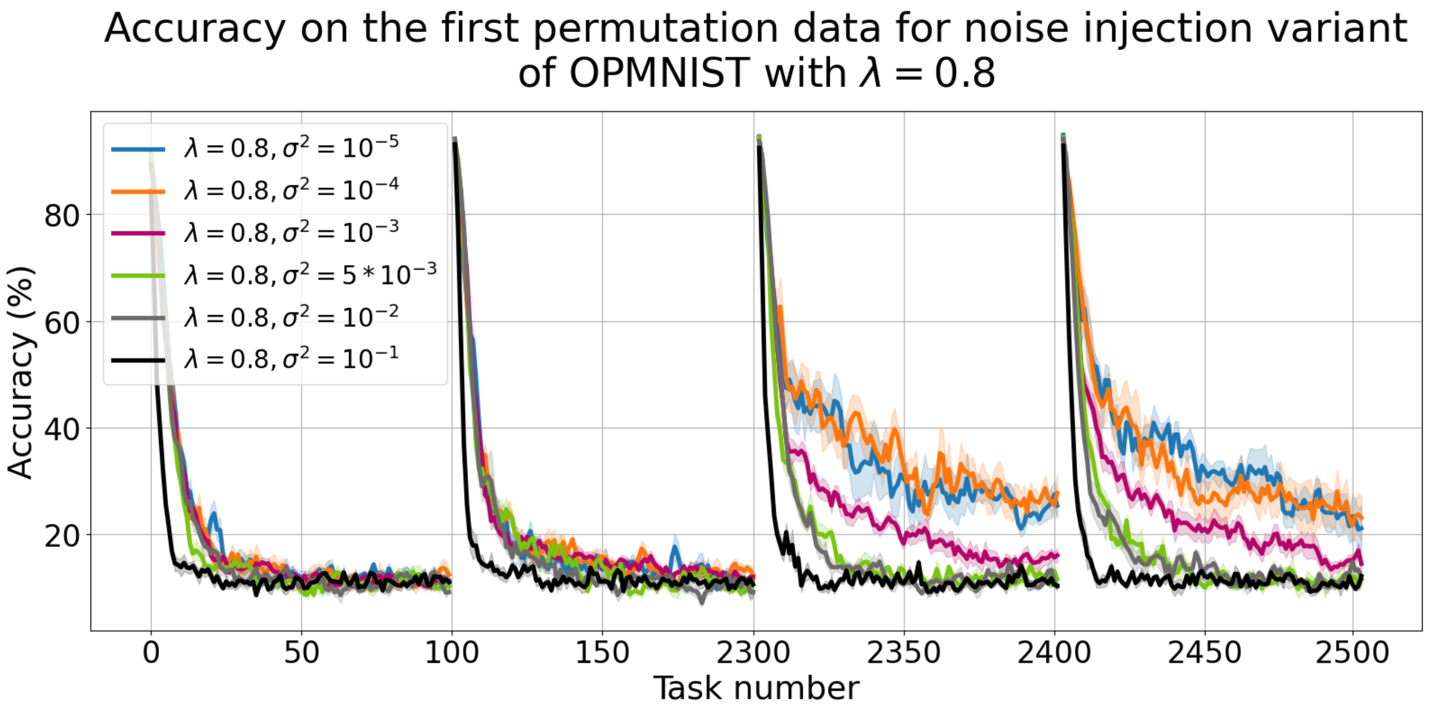


Figure 15: Comparison of *recurrent accuracy curves* of models trained with noise injection variant of CBP, with $\lambda^2 = 0.8$ and varying noise variance (σ^2).

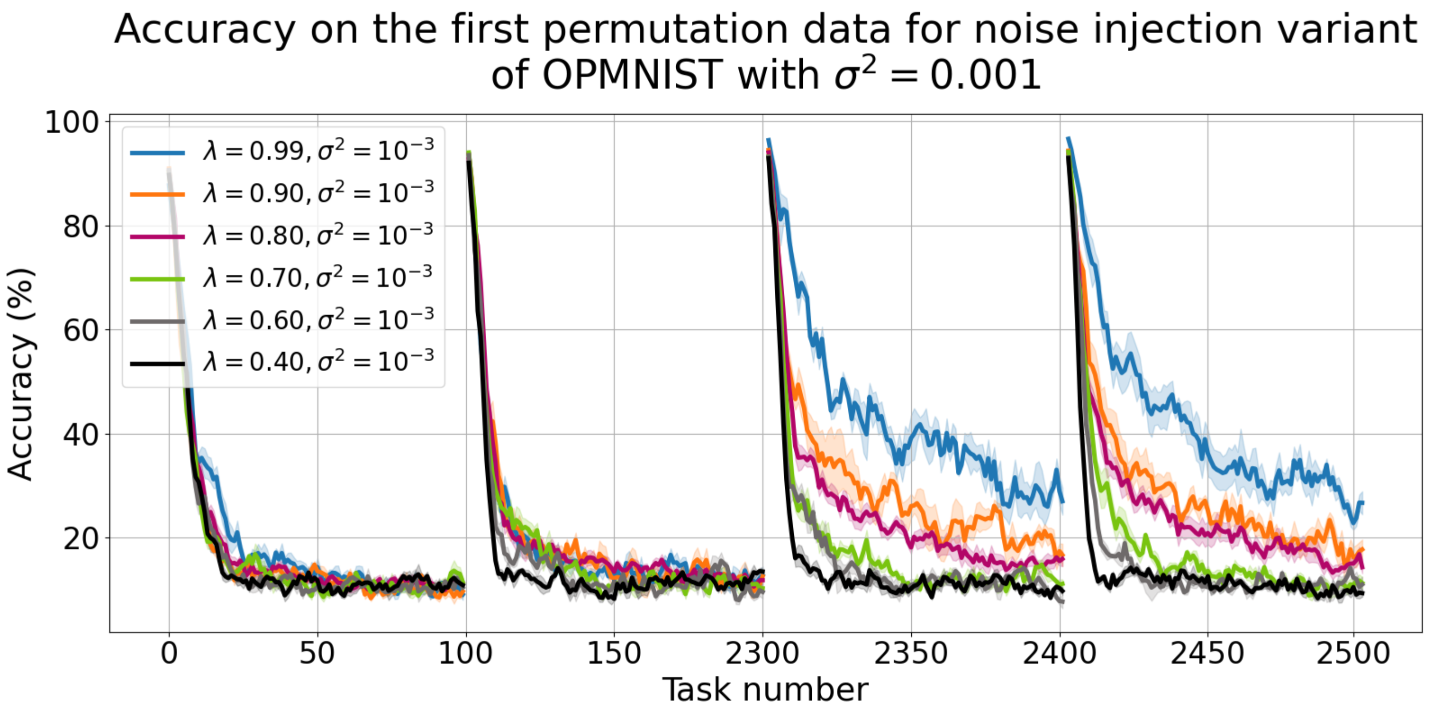
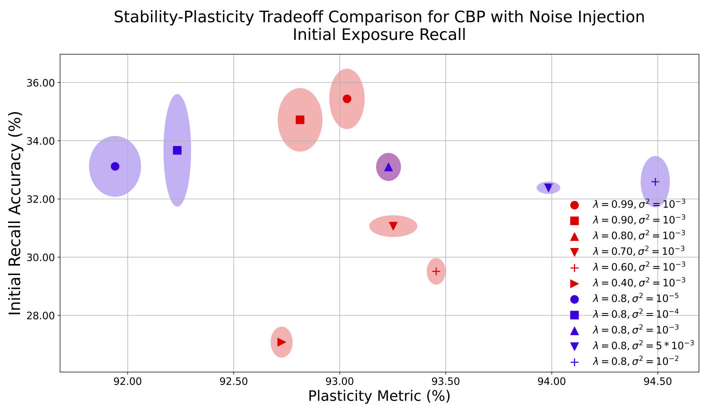
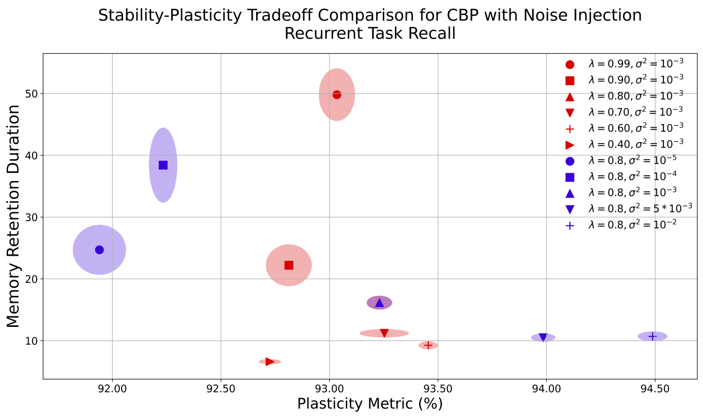


Figure 16: Comparison of *recurrent accuracy curves* of models trained with noise injection variant of CBP, with $\sigma^2 = 0.001$ and varying values of shrink rate (λ) parameter.

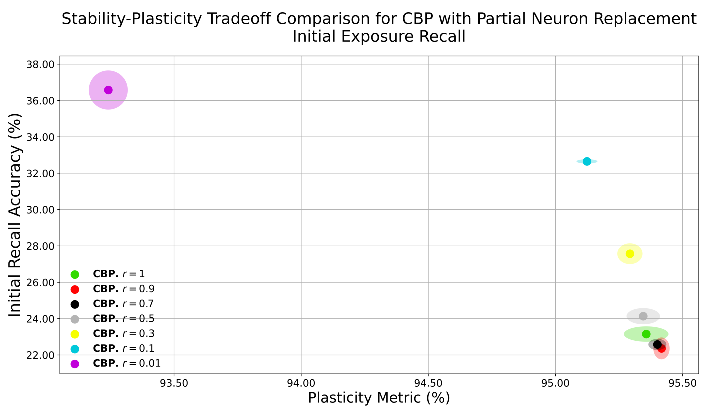


(a) Initial exposure recall experiment trade-off comparison

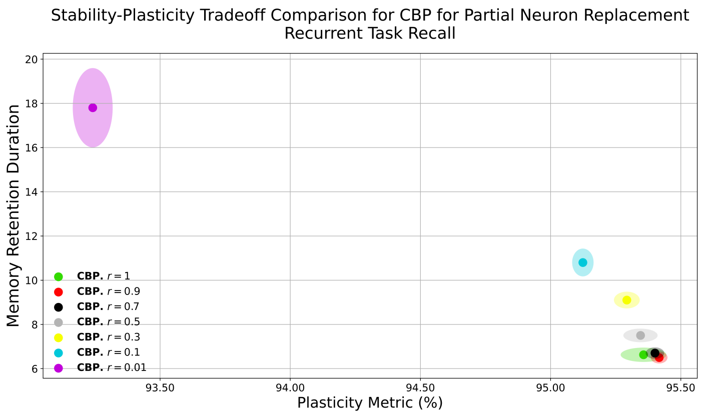


(b) Recurrent task recall experiment trade-off comparison

Figure 17: Stability-plasticity trade-off comparison for noise injection variant of CBP, with different values of shrink rate and noise variance parameters.

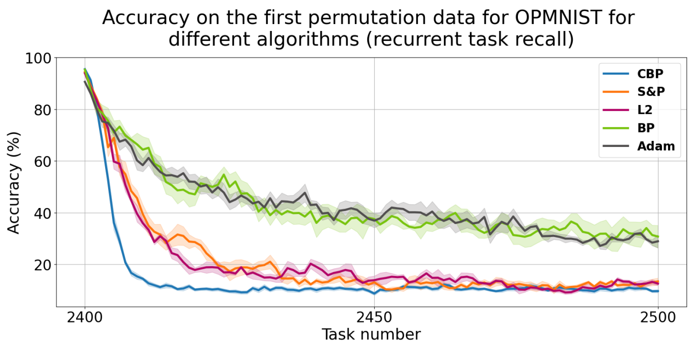


(a) Initial exposure recall experiment trade-off comparison

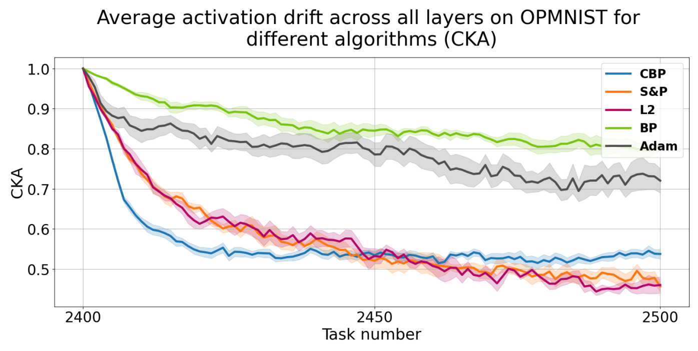


(b) Recurrent task recall experiment trade-off comparison

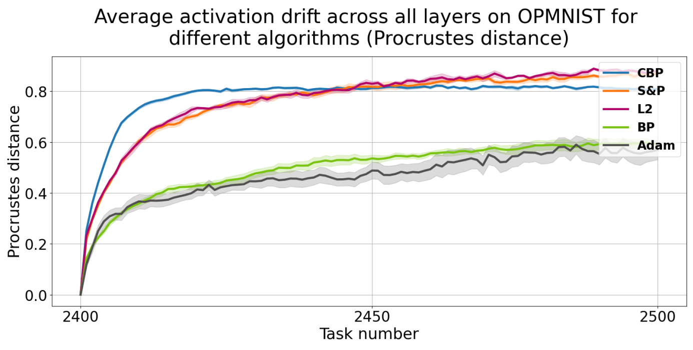
Figure 18: Stability-plasticity trade-off comparison for partial neuron replacement variant of CBP, with different replacement ratio value.



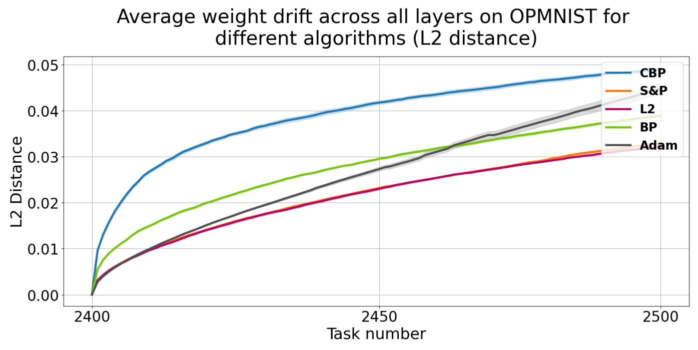
(a) Forgetting



(b) Activation drift (CKA)

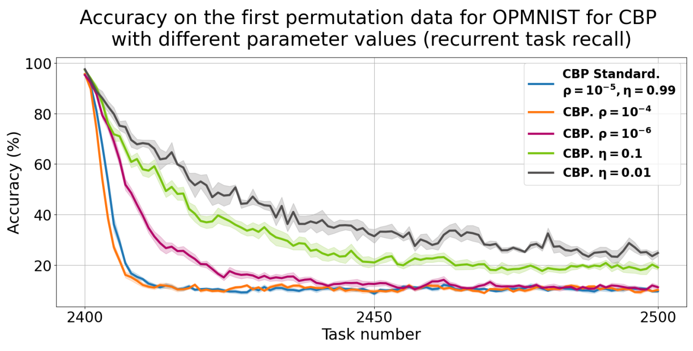


(c) Activation drift (Procrustes)

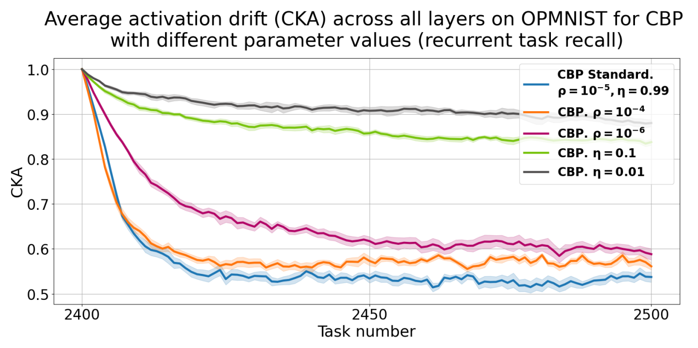


(d) Weight drift (L2)

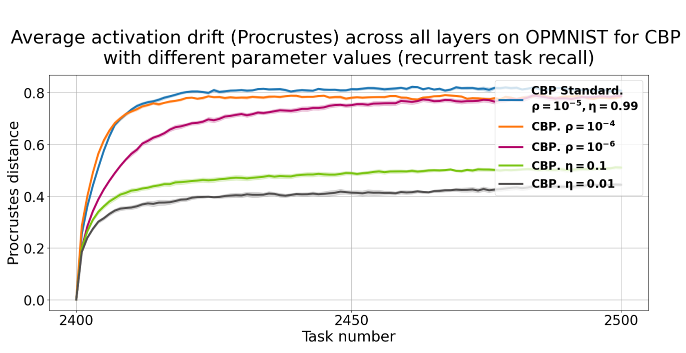
Figure 19: Comparison of forgetting, activation drift (CKA and Procrustes), and weight drift (L2 distance) across different algorithms on OPMNIST.



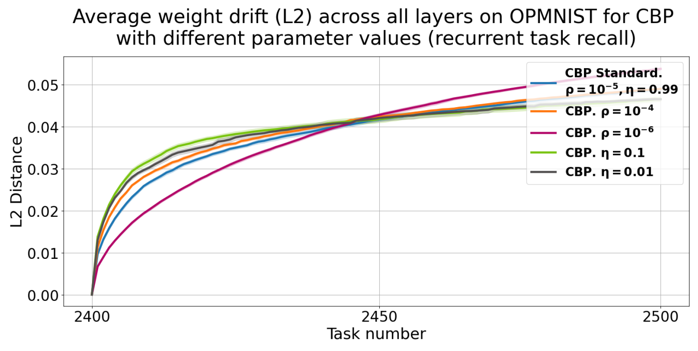
(a) Forgetting



(b) Activation drift (CKA)

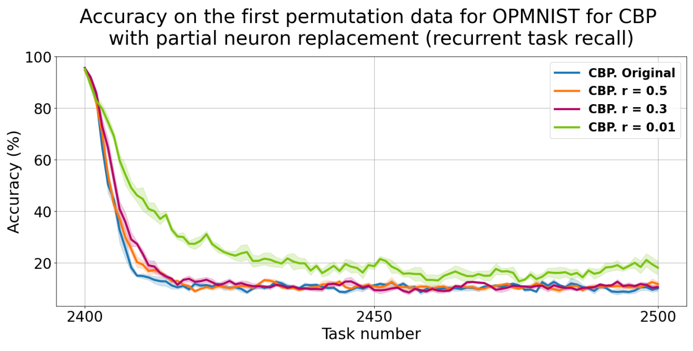


(c) Activation drift (Procrustes)

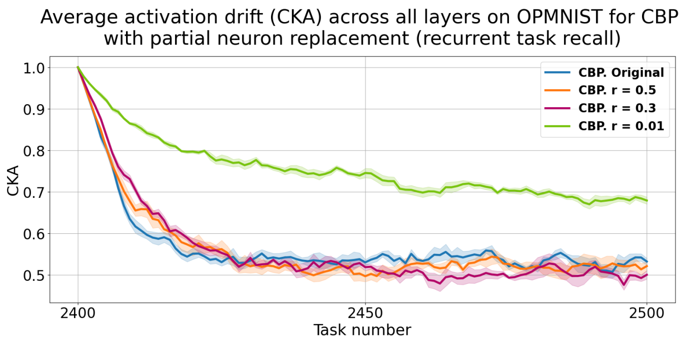


(d) Weight drift (L2)

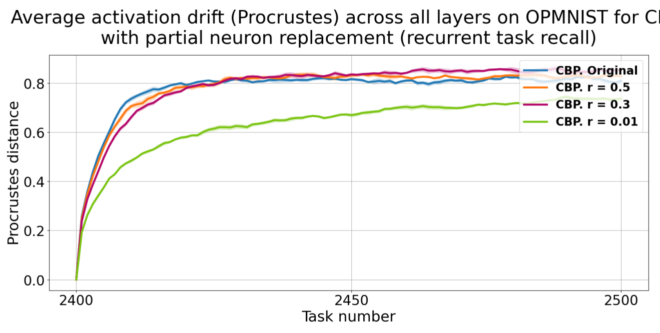
Figure 20: Comparison of forgetting, activation drift (CKA and Procrustes), and weight drift (L2 distance) for CBP across different hyperparameter values.



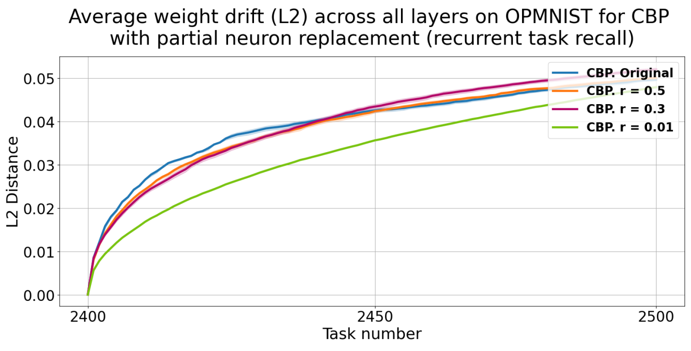
(a) Forgetting



(b) Activation drift (CKA)



(c) Activation drift (Procrustes)



(d) Weight drift (L2)

Figure 21: Comparison of forgetting, activation drift (CKA and Procrustes), and weight drift (L2 distance) for CBP with partial neuron replacement, across different r values.