

# Practical Recommendations for Replay-based Continual Learning Methods

Gabriele Merlin<sup>1</sup>, Vincenzo Lomonaco<sup>1</sup>, Andrea Cossu<sup>1,2</sup>, Antonio Carta<sup>1</sup>, and Davide Bacciu<sup>1</sup>

<sup>1</sup> Department of Computer Science, University of Pisa, Pisa, Italy

<sup>2</sup> Scuola Normale Superiore, Pisa, Italy

**Abstract.** Continual Learning requires the model to learn from a stream of dynamic, non-stationary data without forgetting previous knowledge. Several approaches have been developed in the literature to tackle the Continual Learning challenge. Among them, Replay approaches have empirically proved to be the most effective ones [16]. Replay operates by saving some samples in memory which are then used to rehearse knowledge during training in subsequent tasks. However, an extensive comparison and deeper understanding of different replay implementation subtleties is still missing in the literature. The aim of this work is to compare and analyze existing replay-based strategies and provide practical recommendations on developing efficient, effective and generally applicable replay-based strategies. In particular, we investigate the role of the memory size value, different weighting policies and discuss about the impact of data augmentation, which allows reaching better performance with lower memory sizes.

**Keywords:** Continual learning · Replay-based approaches · Catastrophic Forgetting.

## 1 Introduction

Traditional machine learning models learn from independent and identically distributed samples. In many real-world environments, however, such properties on training data cannot be satisfied. As an example, consider a robot learning a sequence of different tasks. For artificial neural networks, learning a new task causes a deterioration of performance on the previous one. This phenomenon is known as Catastrophic Forgetting [18]. Continual learning [19] is a branch of machine learning which focuses on learning from a sequence of tasks while at the same time preventing catastrophic forgetting. Although many approaches have been developed with different degrees of success, preventing catastrophic forgetting is still a difficult task. Moreover it is difficult to compare these approaches since there is not a standard evaluation protocol [8].

The aim of this work is to deepen our understanding of replay-based strategies [2, 21, 24, 26], a specific category of continual learning strategies, and provide *practical recommendations* to achieve a better efficiency-efficacy trade-off in their

implementation. Replay strategies avoid forgetting by training the model on both current samples and some samples of the past tasks. In this paper, we extensively compare replay-based strategies on different benchmarks and settings to better characterize the role played by their main components in the mitigation of forgetting. We explore three main research directions. The first (Sec. 5) concerns the role of memory size. We extensively test the most popular replay strategies varying this parameter, finding out that the memory size value depends not only on the size of the dataset but also on the difficulty of the tasks and the number of classes involved in the learning process. The second direction (Sec. 6) is related to the balancing of the memory buffer. In the literature the replay buffer is usually balanced to have an equal amount of samples of each past task or class. We propose many weighting policies to distribute samples, unbalanced by task. We discover recent memories are more useful with respect to others, confirming the observation on the human brain [3, 23]. Finally, we test the role of data augmentation [32] in a continual learning scenario (Sec. 7). We find out that performance increases by augmenting the memory, particularly with a low memory budget.

## 2 Related Works

The problem of learning from a sequence of tasks was posed since the origin of artificial intelligence [29, 30]. However, only in 1989 Closkey [18] dealt with catastrophic forgetting directly. In 1995 a new method was proposed to prevent it named Replay [26]. This simple method consists of storing in a buffer some samples and presenting them during consecutive tasks. During the last few years we have witnesses to a significant interest in this area and many strategies have been developed. Replay-base approaches have proved to be effective [1, 2, 5, 25] and they differ mainly by the selection algorithm. Buzzega et al. in [5], proved the effectiveness of the standard Replay strategy [26] using a set of "tricks", even without changing the selection algorithm. Moreover Replay-based approaches are biologically-plausible: previous experiences rehearsal is believed to be important for stabilizing new memories [31].

Despite this prolific research paper production, none of these works compares and investigates replay-based strategies extensively.

## 3 Design Choices

Replay-based approaches rely on a simple yet effective mechanism: replay some previous samples to avoid catastrophic forgetting. However this apparently simple mechanism hides many possible modifications. In this section we describe three possible choices and variations concerning continual learning and replay-based strategies.

### 3.1 Replay Buffer

Replay buffer is the principal component of a replay-based strategy.

The *buffer structure* define how samples are distributed. Samples can be balanced in the buffer by task or class. In this case the amount of samples belonging to the same task/class is the same. This structure rely on the assumption that the task label is known.

*Selection and Discarding procedures* are the principal components of a replay-based strategy. The standard Replay strategy [26] assumes that every sample is important for learning, thus select and discard randomly samples, taking into consideration the buffer structure. More advanced approaches are possible and demonstrate to be effective with more realistic benchmarks. ASER [27] uses data shapley values [9] to score samples and keep only the most informative ones. Selecting examples in GSS [2] consists of maximizing the diversity of samples in the replay buffer as suggested in [20] but using the gradient values. ICarl [24] instead uses an herding strategy to select and discard samples.

### 3.2 Memory size

The size of the memory buffer is a common parameter among all the replay-based strategies. Despite the importance of this parameter only few papers test extensively its impact using different continual learning strategies.

In a realistic application this parameter depends on the hardware resources or time constraints for training. When applying a continual learning strategy in a new setting it is essential to know the amount of samples sufficient to have good performances. For this purpose we investigated the influence of this parameter using different strategy and benchmark (Sec. 5). The aim is to provide some practical recommendation useful to apply a continual learning strategy in new domains.

### 3.3 Weighting Policies

In literature, selection policies do not takes into account the importance of each task. However learning could be difficult for some tasks and it could require more replay of samples.

In a realistic scenario, using a random buffer, we don't have a-priori knowledge of information such as the nature of the current task, the represented classes, the number of samples or the difficulty of the current task. In this setting we have only the possibility to balance the amount of samples belonging to each previous task. For this reason we experiment with some weighting policies to verify the effects of recent and old memories in the learning process (Sec. 6).

This experiment is motivated by some recent findings on the human episodic memory [3, 23], suggesting that episodic encoding occurs preferentially at the end of events.

### 3.4 Augmentation

Data augmentation is a helpful, machine learning technique to help improving the generalization capabilities of a deep network [32]. In a continual learning scenario, using data augmentation, we can store original samples in the buffer and then augment them at training time to have more variety and hopefully increase accuracy. In this way, intuitively we can have a smaller buffer size. Data augmentation in continual learning is explored by Buzzega et al. in [5], in this case, *crops* and *horizontal-flips* are applied in the input stream and in the replay buffer. This augmentation leads to an increment in the test accuracy using the Replay strategy. In a realistic scenario, the training set augmentation is not always possible: the training time increases with an augmented dataset. We investigated the augmentation technique in section 7. The aim of this experiment is to verify whether the augmentation of the training set (and which in particular) is indeed needed to achieve better performance and which augmentation strategy is most impactful.

## 4 Experimental Setup

The goal of this section is to describe benchmarks, models and replay-based strategies used in the experiments. For the experimental part we used Avalanche [15] the reference continual learning framework based on PyTorch. The goal of this library is to provide a shared and collaborative open-source codebase for fast prototyping, training and reproducible evaluation of continual learning algorithms.

### 4.1 Benchmarks and Models

Continual learning algorithm are evaluated by *benchmarks*: they specify how the stream of data is created by defining the originating dataset(s), the amount of samples, the criteria to split the data in different *tasks* or *experiences* [6] and so on. In literature, different benchmarks are used to evaluate results.

We select benchmarks belonging to the *New Classes* scenario i.e. data samples contained in the training set at time-step  $i$  are related to a new dependent variable  $Y$  to be learned from the model. We select three three of them for our experiments: *Split-MNIST* [28], *Split-CIFAR-10* [33] and *Split-TinyImagenet* [17]. These benchmark are derived respectively from MNIST [7], CIFAR-10 [12] and TinyImagenet [13] datasets. We also include *CORe50-NC* [14] in our experiments, a benchmark specifically designed for continual learning. This benchmark is divided in 9 tasks, the first task contains 10 classes, the remaining 8 classes. In our experiments, we set the number of tasks of each benchmark to 5, except for CORe50-NC, with a random order of classes.

Concerning the neural network models, for Split-MNIST we use a Multi-Layer Perceptron with 3 layers and 300 ReLU units at each layer. For Split-Cifar10, Split-TinyImagenet and CORe50-NC we exploit the ResNet-18 model [10] instead.

## 4.2 Strategies

We selected four strategies among the most popular and promising rehearsal approaches.

**Replay.** We select Replay [22, 26] because it is powerful, simple and easily adjustable. It is also a simple way to prevent catastrophic forgetting, and it performs better with respect to more complicated strategies [5]. In our experiment we use random sampling and we randomly choose the samples to discard, to maintain simplicity.

**GDumb.** *Greedy Sampler and Dumb Learner* (GDumb) [21] is a simple approach that is surprisingly effective. The model is able to classify all the labels since a given moment  $t$  using only samples stored in the memory. Whenever it encounters a new task, the sampler just creates a new bucket for that task and starts removing samples from the one with the maximum number of samples. Samples are removed randomly. Compared to others, with the same memory size, this strategy is more efficient, in terms of execution time and resources. In particular setting this simple strategy can outperforms other approaches. However, it is not a valid continual learning strategy, since for each new task the model does not adapt, it must be re-trained from scratch.

**ICarl.** *Incremental Classifier and Representation Learning* (ICarl) [24] is a hybrid approach between rehearsal and regularization. The model parameters are updated by minimizing both a classification loss and a distillation loss. The replay memory is managed by a herding strategy: a sample is added if it causes the average feature vector over all exemplars to best approximate the average feature vector over all training examples. The order of its elements matters, with exemplars earlier in the list being more important. Reducing the exemplar set means discarding the less important samples. We selected ICarl because it is an effective hybrid strategy, in particular with low memory budget.

**GSS.** *Gradient based Sample Selection* [2] is a replay-based strategy. The selection of the memory buffer population is seen as a constraint selection problem. The goal is to optimize the loss on the current examples without increasing the losses on the previously learned ones. Selecting examples consists of maximizing the diversity of samples in the replay buffer using the gradient. The first way to select samples is based on integer quadratic programming, the second solution consists of a faster greedy-alternative and it is sufficient to achieve good performances. Scores for each sample is based on the maximal cosine similarity with a fixed number of others random samples in the buffer. The probability of choosing a specific sample to be replaced is its normalized score. The score of the candidate is then compared to the score of the new sample to determine whether the replacement should happen or not.

Avalanche [15] includes many Continual Learning strategies. It has been necessary to validate the strategies used in the experiments. We made sure to reproduce results of the original paper with the new Avalanche implementation.

## 5 Memory Size Experiment

This experiment is designed to understand the impact of memory size for every selected strategy and have an insight on the amount of samples sufficient to have good performances in a classification task as we propose in section 3.2. In our work, we analyze a vast set of results and try to generalize those across different benchmarks and strategies.

### 5.1 Grid search and Final Models

We select the models through a grid search and we choose a fixed order of classes. The selected parameters for the grid search are chosen following the parameters used in other works [2, 4, 5, 11, 17, 21]. The memory buffer is balanced by task i.e. the memory contains an equal number of samples belonging to each task. For each benchmark we use 10% of the training set as validation set and a batch size of 32 examples. We use 4 epochs for Split-MNIST, 50 epochs for Split-CIFAR10, 100 epochs for Split-Tiny-Imagenet and CORE50-NC. GSS takes up to 10x higher execution time with respect to other strategies. As a result it was necessary to simplify the grid search for Split-MNIST benchmark and we did not test it using other benchmarks.

We have averaged the results of final models over 3 runs changing in each of them the classes order in a random manner. We plot the accuracy values in Figures 1, 2, 3, 4. For each curve we calculate the elbow point, depicted with a black square. In this case, it indicates the optimal trade-off between accuracy and memory size. These values give us an idea of the memory sizes useful to have good performance.

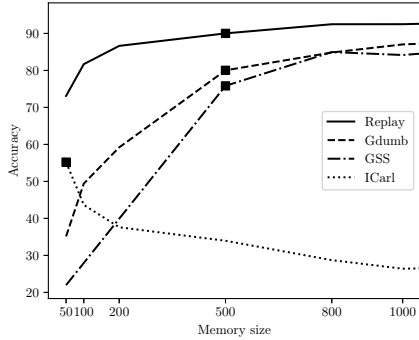
### 5.2 Discussion

Our results show that the Replay strategy is a powerful and simple mechanism that most of the time is able to achieve good performance. Instead, ICarl has a particular behaviour: it performs well with lower memory size. This is due to the herding strategy as confirmed in other works [4, 24]. In the following sections we analyze more in detail these results.

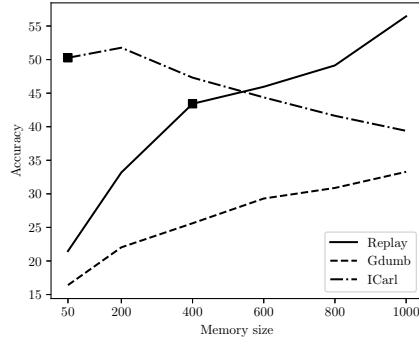
**Split-MNIST.** Replay strategy achieved the best performance with respect to the others. However, GDumb is able to reach good performance with high memory size and a considerably lower training time. ICarl is valid and effective using a smaller memory size. Concerning GSS, the performance are worse than others strategies, but the parameters used for grid search are fewer.

**Split-CIFAR-10.** Interestingly, in Split-CIFAR-10 the Replay strategy is effective only for high memory sizes. Instead, ICarl is much more effective with low memory sizes, it reaches with only 200 samples in memory the same accuracy of Replay strategy with 800 samples in memory. GDumb is not effective in this more challenging benchmark.

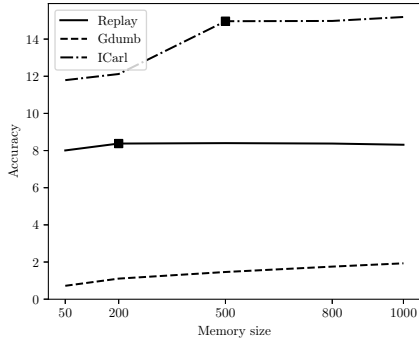
**Split-Tiny-Imagenet.** The performance of various strategies are poor. Instead, ICarl gains accuracy as memory size increases. This behaviour is different



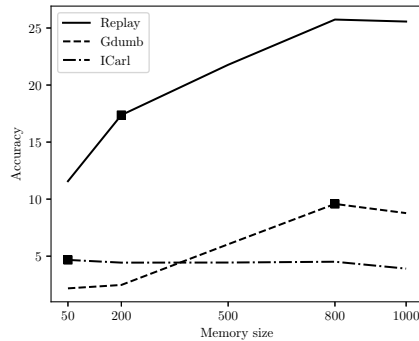
**Fig. 1.** Split-MNIST memory-accuracy curve



**Fig. 2.** Split-CIFAR-10 memory-accuracy curve



**Fig. 3.** Split-TinyImagenet memory-accuracy curve



**Fig. 4.** CORE50-NC memory-accuracy curve

with respect to Split-MNIST and Split-CIFAR-10. This is due to the difference in their tasks, since, contrarily to Cifar and MNIST, Tiny-Imagenet has 200 classes. In fact, if a benchmark includes more classes than another, a greater memory size should be granted.

**CORE50-NC.** In this benchmark, Replay strategy is the most effective with both low and high memory sizes. GDumb and ICarl are ineffective with this benchmark. ICarl slowly increases its accuracy as the memory size increases up to 800. In this case, we can observe a trend inversion in the accuracy values.

This experiment give an insight on the memory size value needed to have good performance. Results show that in most of the case 1% of the training set is sufficient to achieve reasonable results.

## 6 Examples Weighting

The goal of this experiment is to verify the effects of recent and old memories in the learning process. Recent or old memories can have a different impact on the learning process as we declared in section 3.3 We propose and investigate 7 alternatives to the balanced policy over tasks.

### 6.1 Weighting Policies

We propose different weighting policies i.e. methods to distribute samples, for the replay strategy, unbalanced by task. We report them in table 1 with their abbreviations.

Except for *Balanced* policy, all the policies are parametrized by a *factor* parameter. This variable regulates the relevance of a task with respect to the others. For example, in *Increasing* policy the number of memory samples for each task is *factor - time* greater than the number of memory samples for the previous task. If the amount of samples of first task is  $x$ , there will be  $factor * x$  samples for the second,  $factor^2 * x$  for the third and so on.

A particular case is the *Middle* policy that works assigning greater weights to middle distance tasks. For this reason, once a new task arrives, the splitting is recalculated according to the new distribution. Some tasks may need more weight than before, as a result the medium policy does not exploit the full buffer due to those re-calibration. Contrarily, the *Middle+replications* replicates some random samples to fill the buffer. *MiddleHigh* policy gives more weight to middle and low distance samples. In this case the amount of samples of low distance task is the same as the one of middle distance task. The weight of other task is e regulated by the *factor* parameter. Using the same principles we propose *MiddleLow* and *MiddleLow+replications*.

### 6.2 Grid Search and Final Models

We exploit the same grid search parameter and architecture adopted in the first experiment described in section 5.1 for the *Split-MNIST* [28] benchmark with 5 experiences. We average the final results over 6 runs using 3 as *factor* parameter. In table 1 we report the accuracy and standard deviation of these policies varying the memory size.

### 6.3 Discussion

The results highlight that the balanced policy is the best among all. Besides this, the results are interesting. Let us analyze results starting with the most simple weighting policies: Decreasing, Increasing and Middle. For most memory size values, the Increasing policy achieves better results than the Decreasing, but lower with respect to the Middle policy. From this observation we can infer that the most valuable samples are those from low and middle distances from the



**Table 1.** Accuracy and std of the final models, averaged over 6 runs. Bal.=Balanced; Dec.=Decreasing; Inc.=Increasing; Mid.=Middle; Mid.+ =Middle+replications; Mid.Hig.=MiddleHigh; Mid.Low=MiddleLow; Mid.Low+=MiddleLow+replications

	Bal.	Dec.	Inc.	Mid.	Mid. +	Mid.Hig.	Mid.Low	Mid.Low+
50	74.54±3.64	60.36±4.03	62.72±2.43	<b>71.63±4.14</b>	66.02±3.07	71.05±5.55	70.59±5.51	66.51±5.22
100	82.35±1.24	72.80±3.71	73.56±3.22	77.48±3.32	77.55±1.77	<b>81.01±1.55</b>	77.41±2.31	77.92±2.37
200	85.59±0.69	79.37±4.83	77.24±2.82	83.18±1.30	82.93±1.52	84.07±2.30	<b>85.14±1.25</b>	83.72±3.01
500	90.69±0.42	84.08±5.04	83.89±3.62	89.09±0.72	<b>89.57±0.68</b>	89.02±1.07	88.94±0.68	89.48±1.18
800	91.89±0.43	87.19±3.27	86.89±3.07	91.48±0.55	90.78±0.55	91.33±0.79	90.64±0.75	<b>91.70±1.05</b>
1k	92.94±0.36	87.18±2.04	88.60±1.73	91.36±0.6	92.52±0.78	<b>92.70±0.43</b>	92.48±0.45	91.92±0.94
2k	94.69±0.31	90.60±1.71	91.28±1.32	94.15±0.17	93.58±0.67	<b>94.58±0.23</b>	94.50±0.71	93.66±1.08
4k	95.56±0.21	92.43±1.83	93.01±0.79	95.03±0.57	94.97±0.32	95.35±0.19	95.35±0.67	<b>95.38±0.24</b>
5k	95.76±0.22	93.54±1.49	94.14±0.82	95.25±0.31	94.71±0.65	<b>95.71±0.26</b>	95.39±0.34	95.34±0.30

current task. We continue our analysis with the other policies. We observe that the best policy among all is MiddleHigh, confirming our previous statement. The performance of this policy is similar to those of the Balanced strategy. Concerning the policies with replications, results are not better with respect to the same policy without replications. This might depend on the fact that we replicate data without further transformations decreasing the diversity of the data.

## 7 Augmentation

The aim of this experiment is to investigate on the augmentation technique in a continual learning scenario as we propose in section 3.4. Inspired by Buzzega et al. [5] we test different augmentation strategy applied only in the memory samples. The goal is to verify if the augmentation of the training set is indeed needed or if augmenting the buffer memory is sufficient to achieve better performance.

### 7.1 Settings and Results

The experiments have been performed with the Split-CIFAR-10 benchmark. Model, epochs, and batch size are the same described in 4. In this experiment, we fix the learning rate to 0.01 and the momentum to 0. We average the results over 4 runs using different memory size and varying the type of augmentation: Vertical-Flip, Horizontal-Flip, Resize-Crop and Rotation. Results are reported in Table 2.

**Table 2.** Experiment 3. Accuracy and std of final models averaged over 4 runs

	Memory size					
	20	50	250	500	750	1000
<b>Original</b>	19.80±0.54	22.04±0.77	38.74±3.92	44.97±4.08	53.14±1.36	57.42±3.39
<b>Vertical</b>	20.48±1.13	23.12±1.29	37.86±2.22	<b>45.85±2.17</b>	<b>53.33±1.57</b>	54.28±1.06
<b>Horizontal</b>	20.02±0.33	<b>23.73±1.31</b>	35.09±4.73	44.56±3.30	50.11±2.82	55.46±0.57
<b>Crop</b>	19.59±0.91	23.07±0.83	36.63±5.82	45.43±4.29	51.71±2.03	56.92±1.49
<b>Rotation</b>	<b>20.50±0.57</b>	23.07±0.83	36.63±5.82	45.43± 4.29	51.71±2.03	56.92±1.49

## 7.2 Discussion

In this case, the data augmentation shows just a slight increment in accuracy. We suppose that this is due to the training set’s lack of augmentation and the transformation used. However, our experimental results reflect the findings reported in [5]: data augmentation is effective with low memory size. With 20 and 50 of memory size, accuracy is significantly higher.

## 8 Conclusion and Future Works

This work aims to deepen replay-based strategies, providing some insights and practical recommendations on specific implementation issues. We have validated many replay-based strategies already implemented in Avalanche. We investigated multiple aspect of continual learning strategies by means of three experiments.

Concerning the memory size experiment we extensively investigated the behavior of each strategy and benchmark varying the memory size. For each benchmark and for each strategy we found the amount of samples sufficient to have reasonably good results and we provided a general guideline to set this parameter in unseen benchmarks. We understand the role of memory samples of different tasks, testing different weighting policies. The variation of the standard balanced policy has proved to be useful to understand the impact of samples belonging to different tasks. We found out that Middle and Low distance tasks are more important than others. This paves the way to other experiments regarding this aspect, as well as to the development of new strategies exploiting this discovery.

We explored the usage of data augmentation in continual learning. We confirmed the results presented in [5], remarking the importance of augmenting not only the memory buffer, but also the training set. However, augmenting only the memory buffer helps to improve the accuracy, in particular with lower memory size. More experiments concerning these strategies could be performed. Concerning the weighting experiment in chapter 6, it could be interesting to test the weighting policies with more challenging benchmarks. In our experiment we fixed the *factor* parameter but it could be interesting to test other values. Another possible modification is changing the type of augmentation, since we simply replicate some samples. Regarding the augmentation experiment, it could be interesting to observe the same phenomenon on more challenging benchmarks. Concerning the type of augmentation, we test only simple augmentation techniques. It could be interesting to test other neural-based technique.

**Acknowledgements** This work has been partially supported by the H2020 TEACHING project (GA 871385).

## References

1. Aljundi, R., Belilovsky, E., Tuytelaars, T., Charlin, L., Caccia, M., Lin, M., Page-Caccia, L.: Online continual learning with maximal interfered retrieval. *Advances in neural information processing systems* **32** (2019)

2. Aljundi, R., Lin, M., Goujaud, B., Bengio, Y.: Gradient based sample selection for online continual learning. *Advances in neural information processing systems* **32** (2019)
3. Ben-Yakov, A., Henson, R.N.: The hippocampal film editor: sensitivity and specificity to event boundaries in continuous experience. *Journal of Neuroscience* **38**(47), 10057–10068 (2018)
4. Buzzega, P., Boschini, M., Porrello, A., Abati, D., Calderara, S.: Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems* **33**, 15920–15930 (2020)
5. Buzzega, P., Boschini, M., Porrello, A., Calderara, S.: Rethinking experience replay: a bag of tricks for continual learning. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 2180–2187. IEEE (2021)
6. Carta, A., Cossu, A., Lomonaco, V., Bacciu, D.: Ex-model: Continual learning from a stream of trained models. *arXiv preprint arXiv:2112.06511* (2021)
7. Deng, L.: The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine* **29**(6), 141–142 (2012)
8. Díaz-Rodríguez, N., Lomonaco, V., Filliat, D., Maltoni, D.: Don't forget, there is more than forgetting: new metrics for continual learning. *arXiv preprint arXiv:1810.13166* (2018)
9. Ghorbani, A., Zou, J.: Data shapley: Equitable valuation of data for machine learning. In: International Conference on Machine Learning. pp. 2242–2251. PMLR (2019)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
11. Hsu, Y.C., Liu, Y.C., Ramasamy, A., Kira, Z.: Re-evaluating continual learning scenarios: A categorization and case for strong baselines. *arXiv preprint arXiv:1810.12488* (2018)
12. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
13. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. *CS 231N* **7**(7), 3 (2015)
14. Lomonaco, V., Maltoni, D.: Core50: a new dataset and benchmark for continuous object recognition. In: Conference on Robot Learning. pp. 17–26. PMLR (2017)
15. Lomonaco, V., Pellegrini, L., Cossu, A., Carta, A., Graffieti, G., Hayes, T.L., De Lange, M., Masana, M., Pomponi, J., Van de Ven, G.M., et al.: Avalanche: an end-to-end library for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3600–3610 (2021)
16. Lomonaco, V., Pellegrini, L., Rodriguez, P., Caccia, M., She, Q., Chen, Y., Jodelet, Q., Wang, R., Mai, Z., Vazquez, D., et al.: Cvpr 2020 continual learning in computer vision competition: Approaches, results, current challenges and future directions. *Artificial Intelligence* **303**, 103635 (2022)
17. Mai, Z., Li, R., Jeong, J., Quispe, D., Kim, H., Sanner, S.: Online continual learning in image classification: An empirical survey. *Neurocomputing* **469**, 28–51 (2022)
18. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: The sequential learning problem. In: *Psychology of learning and motivation*, vol. 24, pp. 109–165. Elsevier (1989)
19. Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual lifelong learning with neural networks: A review. *Neural Networks* **113**, 54–71 (2019)

20. Pomponi, J., Scardapane, S., Lomonaco, V., Uncini, A.: Efficient continual learning in neural networks with embedding regularization. *Neurocomputing* **397**, 139–148 (2020)
21. Prabhu, A., Torr, P.H., Dokania, P.K.: Gdumb: A simple approach that questions our progress in continual learning. In: *European conference on computer vision*. pp. 524–540. Springer (2020)
22. Ratcliff, R.: Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review* **97**(2), 285 (1990)
23. Reagh, Z.M., Delarazan, A.I., Garber, A., Ranganath, C.: Aging alters neural activity at event boundaries in the hippocampus and posterior medial network. *Nature communications* **11**(1), 1–12 (2020)
24. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. pp. 2001–2010 (2017)
25. Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., Tesauro, G.: Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910* (2018)
26. Robins, A.: Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science* **7**(2), 123–146 (1995)
27. Shim, D., Mai, Z., Jeong, J., Sanner, S., Kim, H., Jang, J.: Online class-incremental continual learning with adversarial shapley value. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. vol. 35, pp. 9630–9638 (2021)
28. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. *Advances in neural information processing systems* **30** (2017)
29. Turing, A.M.: Computing machinery and intelligence. In: *Parsing the turing test*, pp. 23–65. Springer (2009)
30. Weng, J., McClelland, J., Pentland, A., Sporns, O., Stockman, I., Sur, M., Thelen, E.: Autonomous mental development by robots and animals. *Science* **291**(5504), 599–600 (2001)
31. Wilson, M.A., McNaughton, B.L.: Reactivation of hippocampal ensemble memories during sleep. *Science* **265**(5172), 676–679 (1994)
32. Wong, S.C., Gatt, A., Stamatescu, V., McDonnell, M.D.: Understanding data augmentation for classification: when to warp? In: *2016 international conference on digital image computing: techniques and applications (DICTA)*. pp. 1–6. IEEE (2016)
33. Zenke, F., Poole, B., Ganguli, S.: Continual learning through synaptic intelligence. In: *International Conference on Machine Learning*. pp. 3987–3995. PMLR (2017)