A Benchmark and Empirical Analysis for Replay Strategies in Continual Learning

Qihan Yang\textsuperscript{1,2}, Fan Feng\textsuperscript{1}, Rosa H.M. Chan\textsuperscript{1}\textsuperscript{*}

\textsuperscript{1}City University of Hong Kong
\textsuperscript{2}Imperial College London
yangqihan1999@gmail.com, fan.feng@my.cityu.edu.hk, rosachan@cityu.edu.hk

Abstract

With the capacity of continual learning, humans can continuously acquire knowledge throughout their lifespan. However, computational systems are not, in general, capable of learning tasks sequentially. This long-standing challenge for deep neural networks (DNNs) is called catastrophic forgetting. Multiple solutions have been proposed to overcome this limitation. This paper makes an in-depth evaluation of the memory replay methods, exploring the efficiency, performance, and scalability of various sampling strategies when selecting replay data. All experiments are conducted on multiple datasets under various domains. Finally, a practical solution for selecting replay methods for various data distributions is provided.

1 Introduction

Unlike human beings who have continual learning ability, deep neural networks (DNNs), in general, suffer from a significant function decline or forgets the previously learned knowledge when encountering novel information and tasks. Extensive approaches have been proposed to overcome this limitation, including the regularization method (e.g., learning without Forgetting (LwF) [1], elastic weight consolidation (EWC) [2]), dynamic architecture method (e.g., Progressive neural networks (PNN) [3]), and the memory replay method (e.g., deep generative replay (DGR) [4]). The regularization method regularizes the gradient descent path and tries to minimize the loss for new tasks while restricting the change of specific parameters to maintain the memory for the previous tasks. The dynamic architecture approach allocates new neural resources when the model is facing novel tasks. The memory replay method, proven to have the best performance [5], rehearses the learned data from either a data buffer (i.e., experience replay) or a generative model (i.e., generative replay) and jointly trains the neural network with both old task and novel task data.

Several benchmarks are proposed to evaluate the continual learning in computer vision tasks, including general computer vision [6; 7], and robotic vision tasks [8]. However, these benchmarks only focus on specific datasets. Meanwhile, a wide range of continual learning methods is compared in these works, which are not equipped with the in-depth analysis of memory-based methods. In our work, we only focus on the analysis of replay-based methods, which are the most effective and practical methods, using different datasets. To provide this comprehensive benchmark, we evaluate various memory-based replay strategies (including random, confidence, entropy, margin, K-means, core-set, maximally interfered retrieval (MIR), and Bayesian dis-agreement), replaying complex and simple data, and the difference between experience replay and generative replay. Using the standard continual learning datasets (i.e., MNIST [9], CIFAR-10 [10], MiniImagenet [11], and OpenLORIS-Object [8]), we conduct 46 experiments in total.

2 Memory Replay

Among all continual learning approaches, memorization-based approaches using experience replay have generally proven more successful than their regularization-based alternatives [5]. There are, in general, two typical ways to get the replay data: experience replay and generative replay.

2.1 Experience Replay

Early attempts of continual learning use the experience replay approach [12] (Figure 1), which introduces a memory buffer that stores previous task data and regularly replay old samples interleaved with new task data. Specifically, when a new task comes with input data $x$ and labels $y$, the model will also load a subsample $(bx, by)$ from the buffer ($N_s$ represents the number of subsamples), and then a replay data $(mem_x, mem_y)$ is chosen from the subsamples based on certain criteria ($N_r$ represents the number of replay data and $N_c$ represents the number of classes). The model performs training on both the current-task and the replay data. After training, the current task data will be stored in the buffer. As a result, the model can acquire the novel knowledge and maintain some previous memory simultaneously.

2.2 Generative Replay

When the training data for old tasks are not available, only pseudo-inputs and pseudo-targets produced by a memory network can be fed into the model [12]. This rehearsal
technique, named generative replay (Figure 2), does not require extra memory storage but introduces a generative model parallel with the classifier, such as variational auto-encoder (VAE) [13], Generative adversarial network (GAN) [14], and Wasserstein GAN (WGAN) [15]. At any training stage, the generative model can produce pseudo data \( \tilde{x} \), which imitates the old data that have been trained; the classifier can correctly classify the tasks that have been traversed and give pseudo prediction \( \tilde{y} \) based on \( \tilde{x} \). When the new task comes with input data \( x \) and labels \( y \), both the generative model and the classifier are jointly trained with \((\tilde{x}, \tilde{y})\) and \((x, y)\). After training, the generative model will be capable of producing pseudo data that mimic both the previous tasks and the current task data; the classifier will be able to classify both the old tasks and novel task correctly.

3 Evaluation

This paper evaluates 8 replay strategies, which can be classified as random selection, feature embedding, model prediction, and maximal interference.

- **Model Prediction**: entropy, confidence, margin, Bayesian dis-agreement, and MIR;
- **Feature Embedding**: K-means and core-set.
- **Random sampling**.

With the random replay (randomly choosing \( N_r \) replay data from \( N_s \) subsamples) as a benchmark, we explored the open problems with multiple datasets under various domains.

1. the efficiency, performance, and scalability of different sampling strategies when selecting replay data;
2. the relationship between the efficiency and the difficulty of replay data;
3. the performance gap between experience replay and generative replay methods;
4. the relationship between the performances and continual learning orders.

Based on the empirical findings and analysis, we also provide a practical solution for selecting replay methods for datasets under different domains.

3.1 Feature Embedding

We use ResNet-18 [16] in the experience replay methods. For simplicity, the output features of layer1, layer2, and layer3 are modified to 20, 40, and 80. The current classifier takes the subsamples \( bx = [bx^1, \ldots, bx^{N_s}] \) as input, and the output of layer3 will be extracted as feature maps. Notice the dimension of these maps is \((N_s, 80, 1, 1)\). Then, a 2D adaptive average pooling layer will extract the feature vector with dimension \((N_s, 80, 1, 1)\) from the feature maps, followed by a flatten layer stretches the feature vectors to the embeddings \( e = [e^1, \ldots, e^{N_s}] \) with each \( e^i \) for \( i \in \{1, \ldots, N_s\} \) has a dimension of \((N_s, 80)\). After getting the embeddings for subsamples, the replay data can be selected from it with different algorithms (embeddings’ **K-means** and **core-set**).

- **K-means**: K-means is one of the most well-known clustering method [17]. With the \( N_s \) embeddings, it performs a \( N_r \) center clustering. Then, from each cluster, the embedding closest to the centre is the most representative point, which will be chosen to be the replay data from the feature maps, followed by a flatten layer stretches the feature vectors to the embeddings \( e = [e^1, \ldots, e^{N_s}] \) with each \( e^i \) for \( i \in \{1, \ldots, N_s\} \) has a dimension of \((N_s, 80)\). After getting the embeddings for subsamples, the replay data can be selected from it with different algorithms (embeddings’ **K-means** and **core-set**).

- **core-set**: Core-set is a small set of points that approximate a large group of the point’s shape and distribution [18; 19; 20]. Here, we use **Max-Min Distance** to select the core-set. Detailed procedures are shown in algorithm 1.

![Algorithm 1: Max-Min Distance Algorithm](image)

**Algorithm 1: Max-Min Distance Algorithm**

**Result:** Get \( N_r \) representative replay data from \( N_s \) subsamples

**Input:** Randomly select an embedding \( e^i \) as the initial point, \( i \in \{1, \ldots, N_s\} \); mark the initial point as ”selected” point; mark all the other embeddings as ”unselected” points.

**repeat**
- For all the unselected points, cluster them to the selected points based on the minimal distance;
- Select the embedding with the maximum minimal distance as the subsequent replay data;
- Mark the new embedding as the selected point.

**until** \( N_r \) embeddings have been marked as ”selected”;

3.2 Model Prediction

Besides strategies based on feature embedding, there are four popular strategies based on the model prediction. The current model takes the subsamples as input, performs an evaluation, and gets the predictions with size \((N_s, N_r)\). For any subsample \( bx^i \) for \( i \in \{1, \ldots, N_s\} \), the prediction is given...
\( b y^i = [b y^i_1, \ldots, b y^i_{N_s}] \). With this prediction, the replay data (\( \text{mem}_x, \text{mem}_y \)) can be selected based on different calculation methods (confidence, entropy, margin, and Bayesian dis-agreement).

- **Confidence (C):** For any subsample \( b x^i \) for \( i \in \{1, \ldots, N_s\} \), the confidence \( C^i \) is given by its largest class prediction \( \hat{b} y^i_{\max} \) among all the \( \hat{b} y^i_c \) for \( c \in \{1, \ldots, N_c\} \) (equation 1). A large confidence stands for difficult replay data.

\[
C^i \left( \hat{b} y^i | b x^i \right) = \hat{b} y^i_{\max} = \max \left( \hat{b} y^i_1, \ldots, \hat{b} y^i_{N_c} \right) \tag{1}
\]

- Simple replay: top \( N_r \) largest \( C^i \), \( i \in \{1, \ldots, N_s\} \).
- Difficult replay: top \( N_r \) smallest \( C^i \), \( i \in \{1, \ldots, N_s\} \).

- **Entropy (H):** Entropy denotes the randomness or the uncertainty of the data. From the model prediction \( b y^i = [b y^i_1, \ldots, b y^i_{N_c}] \), the entropy \( H^i \) can be calculated with equation 2 for any subsample \( b x^i \) for \( i \in \{1, \ldots, N_s\} \). Similarly, a small entropy stands for high certainty (simple) replay data and a large entropy stands for low certainty (difficult) replay data [21].

\[
H^i \left( \hat{b} y^i | b x^i \right) = -\sum_{c=1}^{N_c} \left( \hat{b} y^i_c \times \log \hat{b} y^i_c \right) \tag{2}
\]

- Simple replay: top \( N_r \) smallest \( H^i \), \( i \in \{1, \ldots, N_s\} \).
- Difficult replay: top \( N_r \) largest \( H^i \), \( i \in \{1, \ldots, N_s\} \).

- **Margin (M):** For any subsample \( b x^i \) for \( i \in \{1, \ldots, N_s\} \), with its prediction \( b y^i = [b y^i_1, \ldots, b y^i_{N_c}] \), the margin \( M^i \) is given by difference between the largest class prediction \( b y^i_{\max} \) and the smallest class prediction \( b y^i_{\min} \) (equation 3). A large margin stands for simple replay data, and a small margin stands for difficult replay data [22].

\[
M^i \left( \hat{b} y^i | b x^i \right) = \hat{b} y^i_{\max} - \hat{b} y^i_{\min} \tag{3}
\]

- Simple replay: top \( N_r \) largest \( M^i \), \( i \in \{1, \ldots, N_s\} \).
- Difficult replay: top \( N_r \) smallest \( M^i \), \( i \in \{1, \ldots, N_s\} \).

- **Bayesian Dis-agreement (B):** Bayesian dis-agreement strategy is inherited from the (Bayesian Active Learning by Dis-agreement (BALD) [23]). This method takes the assumption that there are some model parameters control the dependence between inputs and outputs. As a result, an acquisition function is defined to estimate the mutual information between the model parameters and the model predictions. In other words, it captures how strongly the model parameters and the model predictions for a given data point are coupled [24].

Given a model \( M \) with parameter \( \omega \), a training set \( D_{\text{train}} \), the Bayesian dis-agreement \( B^i \) for a subsample \( b x^i \) for \( i \in \{1, \ldots, N_s\} \) is calculated with equation 4.

\[
B^i \left( \hat{b} y^i, \omega | b x^i, D_{\text{train}} \right) = H \left( \hat{b} y^i | b x^i, D_{\text{train}} \right) - E_{\omega | D_{\text{train}}} \left[ H \left( \hat{b} y^i | b x^i, \omega \right) \right] \tag{4}
\]

The first term \( H \left( \hat{b} y^i | b x^i, D_{\text{train}} \right) \) is the Entropy of the model prediction. For given dataset, the first term is small if the model prediction is certain. The second term \( E_{\omega | D_{\text{train}}} \left[ H \left( \hat{b} y^i | b x^i, \omega \right) \right] \) is an expectation of the entropy of the model prediction over the posterior of the model parameters. For each draw of model parameters from the posterior, the second term is small if the model prediction is certain [25].

In general, the input will have a large \( B \) if its model prediction is prone to vary with changing model parameters and vice versa. A small Bayesian dis-agreement represents stable (simple) replay data, and a large bayesian dis-agreement represents unstable (difficult) replay data.

- Simple replay: top \( N_r \) smallest \( B^i \), \( i \in \{1, \ldots, N_s\} \).
- Difficult replay: top \( N_r \) largest \( B^i \), \( i \in \{1, \ldots, N_s\} \).

For any subsample, it will get \( k \) predictions based on distinct parameters with the help of a drop out layer; then the first term of the acquisition function can be calculated as the entropy of the mean of the \( k \) predictions, and the second term can be calculated as the mean of the \( k \) entropy.

The strategies that rely on feature embeddings and model predictions manage to study the properties of sampled data points and select some representative points to form the replay data. However, they do not take the influence of the new coming data into account. MIR (Figure 3), on the other hand, is a replay strategy based on the novel task data’s interference.

### 3.3 Maximal Interfered Retrieval (MIR)

The crucial idea of MIR is to filter out the samples which will be maximally interfered by the new task data [26]. The interference is measured by the difference between the loss value before and after training the novel data. To get the loss after training the new task data, MIR performs a *virtual update* according to the gradients generated during training the novel data.

The loss function of the classifier is represented by equation 5, and the virtual update can be seen as equation 6.

\[
L \left( M_\omega \left( x \right), y \right) \tag{5}
\]
be selected as replay data. More affected by the new data, and those high-score data will be selected as replay data. A lower score means that the subsample is less affected by the new data; a higher score represents that the subsample is more affected by the new data, and those high-score data will be selected as replay data.

The detailed algorithm is shown in algorithm 2.

**Algorithm 2**: Maximal Interfered Retrieval algorithm

**Result**: Get \( N_r \) most interfered replay data from \( N_s \) subsamples.

**Input**: new task data \((x, y)\), subsamples \((bx, by)\);

**Given**: Model \( M \) parameters \( \omega \):
- Train the model \( M_\omega \) with new data \((x, y)\) and record the gradients \( \nabla L \);
- Copy the current model to a new model, perform virtual update based on the gradients, and get the virtual model \( M_\omega^v \);
- Evaluate the subsamples \((bx, by)\) with both the current model and the virtual model, get two losses \( L, L^v \);
- Calculate the MIR score \( S = L^v - L \);
- Select replay data: \((mem_x, mem_y)\) is the \((bx, by)\) with the top \( N_r \) largest \( S^i \), for \( i \in \{1, \ldots, N_s\} \).

\[
\omega^v = \omega - \eta \nabla L(M_\omega(x), y) \quad (6)
\]

For any subsample \( bx^i \) for \( i \in \{1, \ldots, N_s\} \), with its prediction \( by^i \), current model \( M_\omega \) and virtual updated model \( M_\omega^v \), the MIR score \( S^i \) is calculated with equation 7.

\[
S^i = L^v - L = L(M_\omega(bx^i), by^i) - L(M_\omega(bx^i), by^i) \quad (7)
\]

A lower score means that the subsample is less affected by the new data; a higher score represents that the subsample is more affected by the new data, and those high-score data will be selected as replay data.

3.4 Datasets

The datasets used in this project are **MNIST**, **CIFAR-10**, **MiniImagenet**, and **OpenLORIS-Object**. Each of them is separated into several tasks.

- **MNIST**: separated into 5 disjointed tasks. Each task, 1,000 samples are used for training.
- **CIFAR-10**: separated into 5 disjointed tasks. Each task consists of 2 non-overlapped classes with 9,750 training data, 250 validation data, and 2,000 test data.
- **MiniImagenet**: divided into 20 disjointed tasks with 5 categories each. Each task consists of 2400 training images.
- **OpenLORIS-Object**: separated into four tasks based on the four factors, and each task has all 121 classes.

4 Results and Discussion

4.1 Evaluation Criteria

Three evaluation criteria are used to denote the model performance. Assume a given dataset \( D \) has \( T \) tasks \( T_1, \ldots, T_T \). After training the last task \( T_T \), the **Accuracy** and **Forgetting rate** is given by equation 8 and equation 9. The \( Acc_{T_i} \) represents the accuracy for task \( T_i \) after training the last task, and the \( Best_{T_i} \) represents the best accuracy for task \( T_i \) during the whole training procedure. Usually the \( Best_{T_i} \) is achieved right after the \( T_i \) just be trained.

\[
\text{Accuracy} = \frac{1}{T} \sum_{i=1}^{T} Acc_{T_i} \quad (8)
\]

\[
\text{Forgetting Rate} = \frac{1}{T} \sum_{i=1}^{T} \left( Best_{T_i} - Acc_{T_i} \right) \quad (9)
\]

The third criteria **Average Time** is the average running time for a single run. Given total running time \( T_r \) and number of runs \( n_r \), the average time \( (s) \) is calculated by equation 10.

\[
\text{Average Time} = \frac{T_r}{n_r} \quad (10)
\]

4.2 Replay Strategies Comparison

When comparing different replay strategies, all other factors are set to default: experience replay with simple data. Eight replay strategies are trained with the same settings under same conditions, and compared parallelly. The detailed results are shown in appendix table 5. The **highest accuracy**, the **lowest forgetting rate**, and the **minimal running time** (besides random replay) are embolden. The recordings of the **maximal running time** are italic. Table 1 and 2 summarize the results.

Table 1: Highest accuracy, lowest forgetting rate, least average time, and largest average time achieved on each dataset.

| Dataset         | Accuracy | Forgetting Rate | Least Average Time | Largest Average Time |
|-----------------|----------|-----------------|--------------------|---------------------|
| MNIST           | 0.876    | 0.096           | 3.700              | 22.400              |
| CIFAR-10        | 0.409    | 0.221           | 89.200             | 282.933             |
| MiniImagenet    | 0.158    | 0.280           | 1284.400           | 1745.800            |
| OpenLORIS-Object| 0.966    | 0.014           | 5777.000           | 12270.400           |

For MNIST, almost all the replay strategies achieve a similar result, and the performances are, by and large, reasonably good (accuracy \( 0.870 \), forgetting rate \( 0.100 \)). The reason behind this might be that MNIST is too simple, and no matter what kind of replay data, even using random selected data, makes no significant difference. However, for a slightly diverse dataset like CIFAR-10, the general accuracy is lower, and the forgetting rate is higher (accuracy between \( 0.339 \) to...
0.409 and a forgetting rate between 0.221 to 0.400). For more complex dataset like MiniImagenet, the accuracy drop exponentially, and the forgetting problem is more severe (accuracy between 0.107 to 0.158 and a forgetting rate between 0.280 to 0.402). OpenLORIS-Object also has complex data distribution, but it achieves an accuracy between 0.960 to 0.966 and a forgetting rate between 0.014 to 0.022. Nevertheless, this outstanding performance is not comparable to the other datasets due to the task separation difference.

In the continual learning field, there are different Incremental Learning Scenarios as follow:

- **Instance incremental:** The class number is fixed, while in each learning stage, more novel instances are involved.
- **Class-incremental:** New classes are introduced in each learning stage.
- **Domain-incremental:** The class number is also fixed; however, the model will be used in the context of a different but related input distribution.

The tasks for MNIST, CIFAR-10, and MiniImagenet are in a class-incremental manner. Each time the classifier will encounter new categories that it has not seen before. However, for OpenLORIS-Object, with all classes being presented in the first task, the following tasks are trying to perform similar classification to the same categories under different environmental factors. This follows the domain-incremental manner. The difficulty for training a class-incremental model is much greater than training a domain-incremental model. Thus, the forgetting rate for OpenLORIS-Object is generally lower than that in the other three datasets.

For CIFAR-10 and MiniImagenet and OpenLORIS-Object, different replay strategies show their uniqueness.

- **Bayesian Dis-agreement:** Bayesian dis-agreement has an outstanding performance on the CIFAR-10 and MiniImagenet; however, its forgetting rate on the OpenLORIS-Object increases dramatically compared with other strategies. The class number could be a problem. CIFAR-10 only has 2 classes per task (5 tasks and 10 categories in total), and MiniImagenet has 5 classes per task (20 tasks and 100 categories in total). However, OpenLORIS-Object includes all the 121 classes for each task. This outcome implies that the Bayesian dis-agreement strategy has good performance with a small class number, but it has low scalability.

Intuitively, when the class number gets larger, the mutual information between the model parameters and output prediction will be more complex and ambiguous; thus, the Bayesian dis-agreement strategy will fail to select the representative replay data based on that mutual information.

- **Entropy, confidence, and margin:** These three methods have neutral performances for any dataset. Since they select the replay data only based on the model prediction, the results are more stable and plastic: They neither have a satisfying result as Bayesian dis-agreement on CIFAR-10 and MiniImagenet nor suffer performance decay when the class number gets more extensive as in OpenLORIS-Object. Moreover, the training time for these three methods, in general, are less than Bayesian dis-agreement, K-means, and MIR. As a result, for complex datasets with large data scale and class number, these three methods might be the most efficient choices.

- **MIR:** Theoretically, MIR should be the best method since it tries to replay the data that interfered the most with the new coming data. Practically, it does not show its overwhelming advantage.

On the one hand, the subsample size $N_s$ may not be sufficient for MIR to traverse enough learned data and determine the most interfered samples. On the other hand, the prior assumption for MIR - replay the data with the largest loss difference before and after training the new data will reinforce the old memory - is reasonably correct, but may not be as efficient as studying the model output directly as entropy, confidence, and margin strategies. It also takes time to evaluate the interference by making the prediction twice with both the current model and virtual updated model. Consequently, MIR strategy producing limited improvement, consumes more time than the model prediction-based strategies.

- **K-means and core-Set:** K-means method is very time consuming, and it has a slight improvement when the data is simple and the class number is low. Similarly, a small class number and the limited number of subsamples may not be sufficient for K-means to provide meaningful clusters and discover valuable replay data. Core-Set strategy is also optimal considering its running time and the optimization effect it has.

### 4.3 Replay Data Difficulty Comparison

Entropy, confidence, margin, and Bayesian dis-agreement can be used to compare the difference between replay simple data and replay difficult data. Detailed results are shown in appendix table 6. The better performance in accuracy (higher), forgetting rate (lower) and average time (shorter) are embolden. For MNIST, replay simple data and replay difficult data does not have clear distinctions. However, for CIFAR-10 and MiniImagenet, replaying difficult data generally lead to higher accuracy and lower forgetting rate. The running time for the two cases are comparable.

### 4.4 Experience & Generative Replay Comparison

MNIST is used to compare the efficiency of experience replay and generative replay. The generative replay uses VAE model [13]. The results for experience replay and generative replay comparison are shown in the table 3. The replay data samples

---

**Table 2: Best performances of different replay strategies on the CIFAR-10, MiniImagenet and OpenLORIS-Object.**

| Method       | Accuracy | Forgetting Rate | Least Average Time (s) | Largest Average Time (s) |
|--------------|----------|-----------------|------------------------|--------------------------|
| CIFAR-10     | MIR      | Bayesian Entropy | K-means                |
| MiniImagenet | Bayesian  | Bayesian C. & M. | K-means                |
| OpenLORIS    | H. & C.  | Entropy         | Bayesian               |
during training each task and the reconstruction images for 5 tasks are shown in figure 4 and 5.

| Evaluation Criteria | MNIST ER | Generative Replay |
|---------------------|---------|--------------------|
| Accuracy            | 0.870 +/- 0.012 | 0.563 +/- 0.005 |
| Forgetting rate     | 0.105 +/- 0.015 | 0.486 +/- 0.007 |
| Average time (s)    | 3.050    | 6.754              |

According to the result, the experience replay has higher accuracy, lower forgetting rate, and less time than the generative replay. On the one hand, the generated data do not faithfully imitate the original image due to the replay model’s imperfections. On the other hand, training the generative model introduces extra time consumption. Following the conclusion in [27], the generative replay has worse performance on complex dataset such as CIFAR-10, and continual learning with generative replay remains a challenge.

4.5 Training Sequence Comparison

Since the OpenLORIS-Object dataset splits the data difficulty into three levels, the difference between training the hard data (weak illumination, high occlusion rate, small object size or longer distance from the camera, complex environment) first, and training the easy data (strong illumination, low occlusion rate, big object size or short distance from the camera, simple environment) first could be explored and evaluated. The result is shown in table 4. According to the result, learn the data with small environmental interfered (low difficulty level) and simple data distribution first, then moving to the challenging training samples (high difficulty level) helps the model get higher final accuracy and lower forgetting rate.

| Evaluation Criteria | OpenLORIS Easy First | Hard First |
|---------------------|-----------------------|------------|
| Accuracy            | 0.096 +/- 0.000       | 0.095 +/- 0.016 |
| Forgetting rate     | 0.022 +/- 0.007       | 0.034 +/- 0.020 |
| Average time (s)    | 5521.800              | 5294.400   |

4.6 Analysis

Based on the experimental results\(^1\), we draw the concluding remarks:

- With the same setting, experience replay can achieve better accuracy and a lower forgetting rate compared with generative replay, but it needs an extra working memory to store all previous data. This may lead to memory problem when the task number gets larger or the scale of the dataset is large.
- Under experience replay, for a simple dataset like MNIST, different replay strategies make no significant difference, and they all achieve reasonable good accuracy and a low forgetting rate. For slightly complex datasets like CIFAR-10 or even more challenging dataset like MiniImagenet, Bayesian dis-agreement leads to a good forgetting rate when the categories per task are small. However, Bayesian dis-agreement has low scalability, and it is relatively time-consuming.
- MIR improves the performance compared to random replay, but it also takes more time to train than model prediction-based methods. Moreover, it is not guaranteed to get the lowest forgetting rate or the best accuracy. Embedding based methods like K-means and core-set are not ideal under the same training condition compared with other methods. Thus, for a complex dataset with a large class number for any single take, simple replay strategies like entropy, confidence, and margin are the most effective approaches.
- Replay difficult data is, in general, more optimal for the continual learning model to achieve a low forgetting rate. For OpenLORIS-Object, train the model with objects under simple environmental factors first slightly improves the model performance.

5 Conclusion

This paper provides a benchmark and empirical analysis on the replay-based methods, which are theoretically proven to be the best-performing methods in continual learning [5]. The extensive experiments with various sampling strategies are conducted under different datasets. Based on our analysis, the continual learning problem certainly needs further research, since for complex datasets, even with advanced replay strategies, the accuracy remains low, and the forgetting

\(^1\) All the methods are implemented using PyTorch [28] toolbox with an Intel Core i9 CPU and 4 Nvidia RTX 2080 Ti GPUs.
rate remains high. This work would shed some light on selecting replay methods for various data distributions and lay a solid foundation for further theoretical studies.

References

[1] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018.

[2] L. Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, Ryan Johnson,莴Hinton, Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[3] D. Krizhevsky, Alex Sutskever, and Geoffrey Hinton. Imagenet: A large-scale hierarchical image database. In *2012 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2012.

[4] J. Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
[24] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian Active Learning for Classification and Preference Learning. *arXiv e-prints*, page arXiv:1112.5745, December 2011.

[25] Andreas Kirsch, Joost Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. submitted, 06 2019.

[26] Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[27] T. Lesort, H. Caselles-Dupré, M. Garcia-Ortiz, A. Stoian, and D. Filliat. Generative models from the perspective of continual learning. In 2019 *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019.

[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Neural Information Processing Systems (NeurIPS)*, pages 8024–8035, 2019.
## 6 Appendix

### Table 5: Replay Strategies Comparison Result Summary

|          | Random  | Least Entropy | Largest Confidence | Largest Margin | Least Bayesian Disagreement | K-means | Core-Set | MIR |
|----------|---------|---------------|-------------------|---------------|----------------------------|---------|----------|-----|
| **MNIST** | Accuracy | 0.870 +/- 0.012 | 0.871 +/- 0.014 | 0.869 +/- 0.009 | 0.876 +/- 0.014 | 0.876 +/- 0.008 | 0.870 +/- 0.010 | 0.871 +/- 0.014 |
|          | Forgetting rate | 0.105 +/- 0.015 | 0.101 +/- 0.019 | 0.099 +/- 0.014 | 0.099 +/- 0.017 | 0.099 +/- 0.014 | 0.096 +/- 0.010 | 0.100 +/- 0.013 | 0.101 +/- 0.018 |
|          | Average time (s) | 3.050 | 3.990 | 3.700 | 3.850 | 5.150 | 22.400 | 4.150 | 4.350 |
| **CIFAR-10** | Accuracy | 0.339 +/- 0.024 | 0.393 +/- 0.020 | 0.402 +/- 0.018 | 0.400 +/- 0.018 | 0.395 +/- 0.009 | 0.349 +/- 0.018 | 0.345 +/- 0.018 | 0.409 +/- 0.013 |
|          | Forgetting rate | 0.400 +/- 0.040 | 0.267 +/- 0.036 | 0.259 +/- 0.036 | 0.274 +/- 0.035 | 0.211 +/- 0.018 | 0.373 +/- 0.031 | 0.351 +/- 0.026 | 0.248 +/- 0.019 |
|          | Average time (s) | 62.400 | 89.200 | 90.067 | 89.800 | 234.333 | 282.933 | 89.267 | 152.133 |
| **MiniImagenet** | Accuracy | 0.107 +/- 0.010 | 0.116 +/- 0.016 | 0.134 +/- 0.018 | 0.134 +/- 0.022 | 0.158 +/- 0.014 | 0.111 +/- 0.012 | 0.109 +/- 0.018 | 0.126 +/- 0.015 |
|          | Forgetting rate | 0.402 +/- 0.012 | 0.364 +/- 0.013 | 0.348 +/- 0.026 | 0.334 +/- 0.027 | 0.280 +/- 0.020 | 0.382 +/- 0.013 | 0.358 +/- 0.022 | 0.338 +/- 0.037 |
|          | Average time (s) | 1183.000 | 1290.800 | 1284.400 | 1284.400 | 1622.800 | 1745.800 | 1465.500 | 1379.600 |
| **OpenLORIS** | Accuracy | 0.960 +/- 0.000 | 0.966 +/- 0.004 | 0.966 +/- 0.007 | 0.964 +/- 0.011 | 0.962 +/- 0.004 | 0.962 +/- 0.004 | 0.962 +/- 0.004 | 0.964 +/- 0.007 |
|          | Forgetting rate | 0.022 +/- 0.007 | 0.014 +/- 0.004 | 0.020 +/- 0.010 | 0.018 +/- 0.010 | 0.024 +/- 0.004 | 0.016 +/- 0.007 | 0.026 +/- 0.009 | 0.020 +/- 0.010 |
|          | Average time (s) | 5521.800 | 5777.000 | 9582.000 | 12017.400 | 12270.400 | 7400.600 | 8828.200 | 6753.600 |

### Table 6: Replay Data Difficulty Comparison Result Summary

|          | Least Entropy | Largest Confidence | Largest Margin | Least Bayesian Disagreement | Accuracy | Forgetting rate | Average time (s) |
|----------|---------------|-------------------|---------------|----------------------------|---------|----------------|------------------|
| **Least Simple Data** | 0.871 +/- 0.014 | 0.869 +/- 0.009 | 0.876 +/- 0.014 | 0.876 +/- 0.010 | 0.877 +/- 0.008 | 0.874 +/- 0.009 | 0.872 +/- 0.014 | 0.876 +/- 0.009 |
| MNIST    | 0.101 +/- 0.019 | 0.099 +/- 0.014 | 0.099 +/- 0.017 | 0.099 +/- 0.014 | 0.095 +/- 0.010 | 0.098 +/- 0.016 | 0.100 +/- 0.019 | 0.101 +/- 0.014 |
|          | 3.900 | 3.700 | 3.850 | 5.150 | 3.700 | 4.000 | 4.100 | 4.750 |
| **Least Difficult Data** | 0.393 +/- 0.020 | 0.402 +/- 0.018 | 0.400 +/- 0.018 | 0.395 +/- 0.009 | 0.402 +/- 0.017 | 0.403 +/- 0.011 | 0.407 +/- 0.013 | 0.403 +/- 0.008 |
| CIFAR-10 | 0.267 +/- 0.036 | 0.259 +/- 0.036 | 0.274 +/- 0.035 | 0.221 +/- 0.018 | 0.244 +/- 0.028 | 0.233 +/- 0.023 | 0.247 +/- 0.017 | 0.211 +/- 0.018 |
|          | 89.200 | 90.067 | 89.800 | 234.333 | 89.933 | 91.667 | 88.733 | 235.800 |
| **Least Simple Data** | 0.116 +/- 0.016 | 0.134 +/- 0.018 | 0.134 +/- 0.022 | 0.158 +/- 0.014 | 0.122 +/- 0.007 | 0.146 +/- 0.018 | 0.136 +/- 0.029 | 0.148 +/- 0.017 |
| MinImagenet | 0.364 +/- 0.013 | 0.348 +/- 0.026 | 0.334 +/- 0.027 | 0.280 +/- 0.020 | 0.358 +/- 0.004 | 0.322 +/- 0.031 | 0.336 +/- 0.025 | 0.260 +/- 0.032 |
|          | 1290.800 | 1284.400 | 1284.400 | 1622.800 | 1266.800 | 1267.000 | 1280.400 | 1607.200 |