Leveraging Task-Specific VAEs for Efficient Exemplar Generation in HAR

Kann Bonpagna, Sandra Castellanos-Paez, Romain Rombourg, Philippe Lalanda

Leveraging Task-Specific VAEs for Efficient Exemplar Generation in HAR

Bonpagna Kann[1], Sandra Castellanos-Paez[1], Romain Rombourg[2], and Philippe Lalanda[1]

[1] Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
[2] Univ. Grenoble Alpes, CNRS, Grenoble INP, G2Elab, 38000 Grenoble, France
firstname.lastname@univ-grenoble-alpes.fr

Abstract. The emerging technologies of smartphones and wearable devices have transformed Human Activity Recognition (HAR), offering a rich source of sensor data for building an automated system to recognize people's daily activities. The sensor-based HAR data also enables Machine Learning (ML) algorithms to classify various activities, indicating a new era of intelligent systems for health monitoring and diagnostics. However, integrating ML into these systems faces the challenge of catastrophic forgetting, where models lose proficiency in previously learned activities when introduced to new ones by users. Continual Learning (CL) has emerged as a solution, enabling models to learn continuously from evolving data streams while reducing forgetting of past knowledge. Within CL methodologies, the use of generative models, such as Variational Autoencoders (VAEs), for example, has drawn significant interest for their capacity to generate synthetic data. This reduces storage demands by creating on-demand samples. However, the application of VAEs with a CL classifier has been limited to low-dimensional data or fine-grained features, leaving a gap in harnessing raw, high-dimensional sensor data for the HAR model. Our research aims to bridge this gap by constructing VAEs with filtering mechanism for direct training with raw sensor data from the HAR dataset, enhancing CL models' capability in class-incremental learning scenario. We demonstrate that VAE with a boundary box sampling and filtering process significantly outperforms both traditional and hybrid exemplar CL methods, offering a more balanced and diverse training set that enhances the knowledge acquisition of the model. Our findings also emphasize the importance of sampling strategies in the latent space of VAEs to maximize data diversity, crucial for recognizing the variability in human activities for better representation of each activity in each CL task.

Keywords: Continual Learning · HAR · Replay Methods · VAE

1 Introduction

With the introduction of smartphones and wearable devices, a wealth of sensor data became available, offering a more nuanced and automated approach to understanding human activities in HAR systems. These devices, carried by millions of users, continuously collect data through built-in sensors, such as accelerometers, gyroscopes, capturing the information of users' daily movements and behaviors. These data-driven HAR systems enables the development of ML algorithms to identify different activities through the patterns inherent in this sensor data. The shift towards leveraging ML models to interpret this data shows a promising insight into human activity, paving the way for intelligent systems capable of enhancing personal health tracking, and diagnosis.

However, the ever-changing nature of real world data streams pose a fundamental problem to classical ML approaches. More particularly, classically trained models tend to forget previously acquired knowledge when exposed to new data, a phenomenon called catastrophic forgetting. This issue is particularly relevant in HAR, where data streams come from sensors on smartphones or wearable devices, capturing activities of users such as walking, running, or sitting over time. As new activities are introduced, a traditional ML model might lose its ability to recognize the activities it was previously trained on. To alleviate this problem, Continual learning (CL) is introduced, focusing on developing algorithms and methodologies that allow models to learn from changing data streams while retaining prior knowledge. In the context of CL, a *task* refers to learning how to recognize a set of activities, and a CL classifier is a model whose goal is to learn from a sequence of tasks. When a user performs more activities, additional tasks are added to the model training. CL approaches are thus applied to the model training to enable the model to adapt to new tasks sequentially without losing old knowledge.

Among the strategies in CL, *replay methods* stand out for their ability to mitigate forgetting by revisiting a subset of old data when learning new tasks, thus preserving earlier knowledge. Despite their promise, storage constraints make it impractical and costly. Two approaches exist to address this issue: storing and exploiting a small subset of examples or training a generative model to generate as many examples as needed on the fly. The first approach naturally leads to heavy data imbalance while the second is more complex and some generative models are notoriously hard to train.

Employing generative models, specifically Variational Autoencoders (VAEs), in CL can be a compelling solution. VAEs present several key advantages in the context of CL on smart devices. They are lightweight, easy to train (contrary to GANs) and can generate samples at a low computational cost (contrary to Normalizing Flows or Score Matching methods).

By leveraging VAEs, smart systems can continuously learn and adapt to new activities without the limitations of data access size, ensuring that the storage costs are minimized. However, prior implementations have predominantly focused on low-dimensional data and engineered features. This leaves a gap in the application and effectiveness of these models when dealing with the complexity and high dimensionality characteristic of raw sensor-based HAR data.

In our study, we address this issue by training the CL model with VAEs on raw sensor data and generating refined data through a filtering process based on classifier predictions within the VAE. In Section 2, we present an overview of CL, elaborating different CL methodologies and their implementations across different fields, followed by a discussion on how VAEs contribute to support the CL model in mitigating forgetting while learning new knowledge. Then, the VAE-Based CL framework and the filtering process are elaborated in Section 3. In Section 4, we delve into experimental protocol details regarding CL training with both real and synthetic data samples. Experimental settings are presented in Section 5 before findings and discussion are detailed in Section 6. Finally, in Section 7, we conclude our work and suggest future research directions.

2 Background and Related Work

In the domain of HAR, a number of previous works have been conducted on sensor-based HAR, emphasizing feature extraction methods and training process. Bulling et al. [3] investigated different feature extraction techniques based on statistical analysis on the features of the HAR data. Despite displaying encouraging results, the extracted features are carefully engineered and heuristic in the process, lacking a generalized or systematic approach to accurately classify human activities. To overcome this limitation, Hammerla et al. [9] examined the use of convolutional neural networks (CNNs), and recurrent neural networks (RNNs) across multiple HAR datasets [2,5,25], which comprised movement data from wearable sensors. Their findings suggest that deep learning models excel at identifying local patterns within sensor data, and the inherent translational invariance of these models contributes to their high accuracy in activity recognition. Nonetheless, new issues arise as these models were trained on predefined activities, posing difficulties in adapting to new activities that users do in their daily routine [28].

To alleviate this issue, the research in Continual Learning (CL) is gaining interest for its ability to enable models to adapt to new data while retaining previously acquired knowledge. CL displays promising applications in smart homes [7], sports training [20], and healthcare [31]. CL employs various strategies to maintain past knowledge when facing new tasks. Architecture-based methods [6, 23, 29] ensure minimal interference between tasks by modifying the network's structure for new information, but this leads to increased complexity and potential scalability issues. To address this, regularization strategies [1, 13, 18] keep the model architecture fixed, applying constraints on weight updates to protect old knowledge. However, they struggle in the class-incremental scenario where there is significant similarity between classes in each task [10].

Replay approaches [19, 22, 24, 27] address this issue by incorporating a subset of data from previous tasks alongside current task data during training, which consists of either real or synthetically generated samples. Real data ensures direct knowledge recall but raises issues like increased memory use and potential data distribution misrepresentation. To overcome these, generative models [8, 14] are used to create synthetic data, which reduces memory storage and addresses privacy concerns, all while ensuring the synthetic samples accurately reflect essential task-specific features.

The use of generative models to improve the model performance in CL have been widely implemented on computer vision with remarkable outcome [17,21,30]. However, in sensor-based HAR, the collected data is time-series data which is high dimensional. As a result, it poses a big challenge in applying CL methods to train the model in CL scenarios. Ye et al. [35] apply the work of Shin et al. [30], Deep Generative Replay (DGR), to HAR datasets, using Generative Adversarial Networks (GANs) to generate samples for CL tasks. This approach can lead to significant computational costs due to the necessity of separate GANs and classifiers for each task, contradicting the CL paradigm of using a singular classifier. Moreover, they further adopted the strategy from Van and Tolias [33] of integrating a VAE directly into the primary classifier. Despite being innovative, this approach also introduces the risk of overfitting due to an increase in parameters and model complexity, particularly when limited training data is available. Additionally, they also propose HAR-GAN, which incorporates GANs as one part of their CL framework. While showing encouraging results, their CL framework has a limitation regarding scalability issues as the network grows with each new class, making it less suitable for resource-constrained environments. In addition, the limitation also lies on the data imbalance despite their effort in including a method to deal with it.

From the benchmark results of Jha et. al [12] of different CL approaches with a variety of HAR datasets, Learning a Unified Classifier Incrementally via Rebalancing (LUCIR) [11] has demonstrated a significant contribution in tackling the issue of data imbalance using cosine normalization. Despite the effort from previous studies, most of them are conducted with low-dimensional data and fine-grained features extracted from the HAR dataset without any systematic feature extraction process. While shown effective, the limitation lies on the training of generative models, particularly when involving with the high dimensional data, including the raw sensor data. Besides, not only is sensor data high dimension but there is also a high variance in sensor data of each activity due to the different individual [16] or sensor quality [36]. Despite multiple applications of existing CL approaches across different domains, the study on conducting a CL strategy with the raw sensor-based HAR data is still limited.

3 VAE-Based CL Framework with Classifier and Filtering Process

In our work, we aim to reduce the complexity associated with training generative models by developing a task-specific VAE to assist the CL classifier in a class-incremental learning scenario.

This approach is designed to directly expose the model to raw sensor data from new activities in each task, leveraging the VAE to generate exemplars. This strategy facilitates knowledge retention, addresses storage constraints and naturally avoids training challenges related to data imbalance between the exemplars and training data in the new task. By focusing on training with raw sensor data, our study seeks to create a more direct and efficient method to accommodate new activities while accounting for the inherent variability and high dimensionality of sensor data in HAR.

3.1 VAE and CL classifier architecture

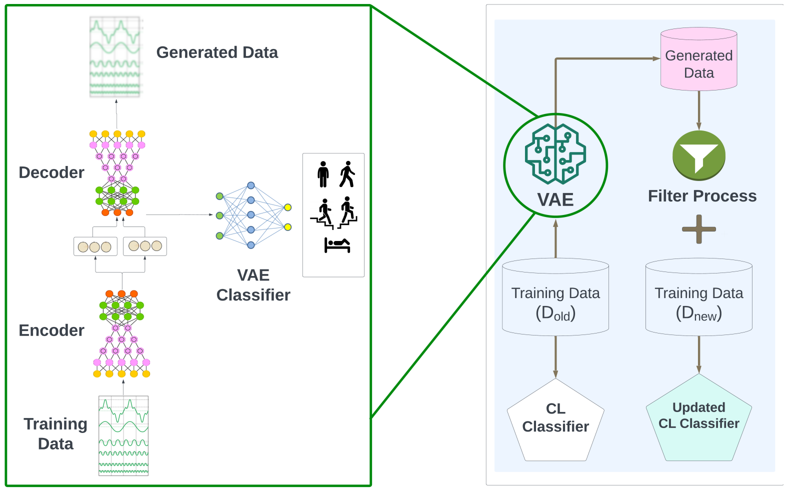
We construct a VAE with three main components: an encoder, a decoder, and a classifier as displayed in Fig 1(a). The encoder transforms high-dimensional input data into a lower-dimensional latent space, creating a compressed representation that captures essential features of the data. From this latent space, sampled latent vectors are passed to the decoder, which reconstructs the input data. The VAE classifier, built upon the latent representation, is used for labeling the reconstructed data. This classifier also imposes additional structure on the latent space, enhancing feature distinction between different classes. This dual capability makes VAEs valuable for generating meaningful and diverse samples with accurate labels. The reconstructed data can then be used as generated data to combine with the data from the new task to train the CL classifier during the CL training process.

The encoder consists of five 1D convolutional layers, each followed by batch normalization and LeakyReLU activation. Max-pooling operations with the kernel size and stride of 2 are also conducted between each convolutional layer to reduce the spatial dimensions of the feature maps. In the first 3 convolutional layers, a kernel size of 3 is used and the number of kernels are arranged as 16, 32, and 64 respectively. Finally, the final 2 convolutional layers have 64 filters with a kernel size of 5. The tensor is then fed into two separate fully connected layers with 64 neurons to produce the mean and log variance of the latent distribution, denoting the parameters for the probabilistic encoding of the inputs.

The decoder begins with a fully connected layer with the size of 384. After that, another series of three transposed convolutional layers, each equipped with LeakyReLU activation. The first two layers used 16 filters with the kernel sizes of 5 and 3. The final transposed convolutional layer has 6 filters and a kernel size of 3.

Lastly, the VAE classifier is a MLP (multilayer perceptron) with a single hidden layer of 32 neurons with LeakyReLU activation. The softmax activation function is applied in the final layer to obtain a probability distribution over the class labels.

On the other hand, the CL classifier, which is responsible for learning tasks incrementally, is a Convolutional Neural Network (CNN) with four convolutional layers with 16, 32, 32, 64 filters respectively. The first two layers use filter size of 3, and the following two layers use a filter size of 5. Each convolutional layer is equipped with a batch normalization, ReLU activation and max-pooling operation using a filter size of 2. Finally, the tensor is fed to the fully connected layer with 32 units, followed by the final output layer.



(a) (b)

Fig. 1. (a). VAE architecture (b). Overview of VAE with filtering process framework

3.2 Filtering mechanism in generated sample selection

For each task in CL training, a VAE is trained individually as described in Section 4.3. This designed process aims to enhance the fidelity of the generated data, improving the representativeness of the generated data for each class. Through the filtering criterion that excludes samples below the prediction score threshold, this strategy can minimize the inclusion of ambiguous samples. This can improve the overall quality and reliability of the pseudo-samples, thereby optimizing the learning efficiency and predictive performance of models. This proposed framework is illustrated in Fig 1(b).

4 Experimental protocol

4.1 CL Training with random sampling from the real data

In this experiment, two classes are randomly selected and input into the CL classifier for training. At each training task, a predetermined quantity of samples are chosen from the training data via random selection. Following the completion of the first task, the CL model is trained with two new classes for subsequent tasks. After each task is completed, the model is evaluated using a test set that includes all the classes previously encountered during the training so far. This process repeats, continually updating the classifier with new classes and assessing its performance after each task.

4.2 Training with Hybrid CL Methods using real data

This experiment assesses model performance when training with different CL strategies. Also, it allows us to evaluate the effectiveness of using generated samples with VAE compared to the different CL methods. The training process with hybrid CL methods is illustrated in Fig 2.

Similarly to the previous experiment, it starts by randomly selecting two classes for the initial task. However, different methods are applied in sample selection for each CL approach such as random and herding sampling [34]. In addition, the model from the previous task is also used for different processes in each CL approach. The CL approaches selected for these experiments are Elastic Weight Consolidation with Replay (EWC Replay), Incremental Classifier and Representation Learning (iCaRL), and LUCIR.

Elastic Weight Consolidation (EWC) Proposed by Kirkpatrick et al. [15], EWC addresses catastrophic forgetting in CL by identifying and preserving the weights crucial for previously learned tasks. In this approach, the Fisher Information Matrix (FIM) is constructed to calculate the importance of each parameter in each task. When learning new tasks, a penalty is applied to the loss function for significant changes from the important parameters. In our implementation, EWC serves as a weight regularizer of the CL classifier to maintain knowledge of previous tasks, complemented by exemplars randomly selected from the training data for each task.

Incremental Classifier and Representation Learning (iCaRL) iCaRL, introduced by Rebuffi et al. [24], is a hybrid approach in CL that leverages knowledge distillation and memory replay. In each task, exemplars are selected using the herding sampling technique from each class of training data and stored in fixed memory. In herding sampling, samples are selected based on their closeness to the class mean in the feature space to ensure a comprehensive representation of each class. During model training, the loss function of iCaRL combines Cross-Entropy (CE) loss for new class learning with Knowledge Distillation (KD) loss for the preservation of previously acquired knowledge, facilitating seamless knowledge transfer. With this combination of components, iCaRL allows for updating the model with new data without forgetting old knowledge, enabling the network to perform well in different CL scenarios.

Learning a Unified Classifier Incrementally via Rebalancing (LUCIR) LUCIR is a hybrid CL approach which is implemented in our experiment with the use of exemplars. Proposed by Hou et al [11], this method combines the three components to deal with the data imbalance that naturally occurs when using exemplars. The first component, Cosine Normalization, is applied to the final layer to level the differences caused by the varying magnitudes of embeddings and biases as those of the new classes tend to be significantly higher than those from previous tasks. The second component, the Less-Forget Constraint, is applied to preserve the integrity of knowledge from earlier tasks. It maintains the established geometric configuration of the older class embeddings of older classes throughout subsequent tasks. The last component, Inter-Class Separation, applies margin ranking loss to create a distinction between old and new class samples within the training data.

With the combination of these three elements, LUCIR has shown to be an effective CL algorithm for preserving knowledge from earlier tasks [11,12]. Hence, in this study, we are going implement this method with real-data exemplars to compare the results with the use of VAE in the class-incremental learning scenario.

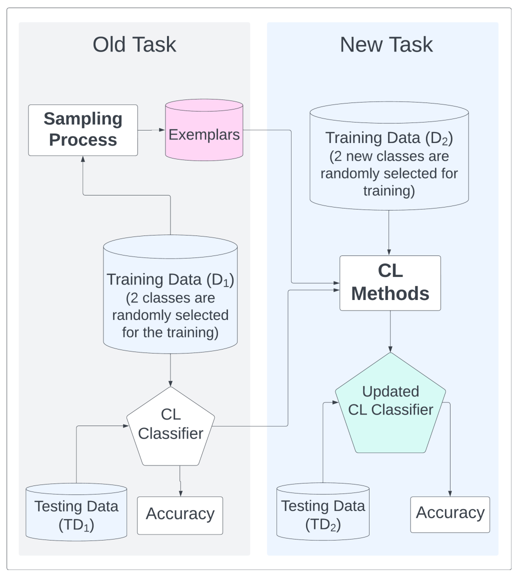


Fig. 2. Training process with CL methods using real data as exemplars

4.3 CL training with generated data from VAE and the filtering process

For our approach, the CL classifier is trained with the first two randomly selected classes. In addition, the training data from the current task is also used to train a VAE specialized in generating sample from the current task. For subsequent tasks, the saved VAEs from each of the previous tasks are loaded to generate data of the previous classes according to the assigned task of the VAE.

In the data generation process, we used two strategies for sampling latent vectors from the latent space:

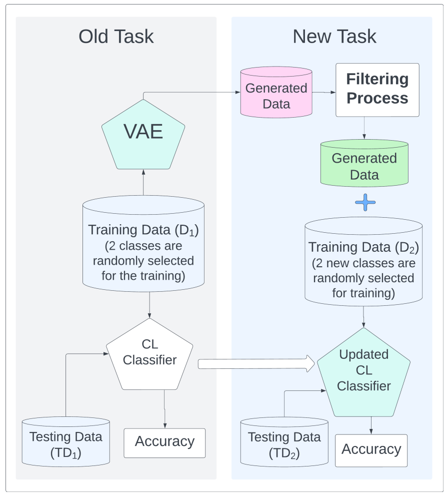


Fig. 3. Training process with VAE as a generative model.

- Adaptive Boundary Sampling: Adjusting boundaries dynamically in each latent space dimension based on data distribution percentiles, ensuring coverage of key variations learned by the VAE. However, this can lead to a lack of diversity of the data representation. Latent vectors are generated by sampling random values within these flexible boundaries.
- Boundary Box Sampling: Setting fixed boundaries using the minimum and maximum values for each latent space dimension, enhancing data diversity but potentially including lower-quality samples. Latent vectors are generated by sampling random values within these predefined ranges.

The sampled latent vectors are input into the VAE classifier for the labeling process, and further into the decoder for generating pseudo-samples. Then, a filtering process was implemented. For larger coverage, it was only used with the boundary box strategy. This filtering process only keeps samples with a classification confidence above a given threshold, p, from the VAE classifier. The performance of the model is evaluated with and without filtering process.

When the data generation process is completed, the generated data is merged with the training data of the two new classes in the current task. After training, the classifier is then evaluated in the same way as presented in Section 4.1. The training process with generated data from VAE is displayed in Fig 3.

4.4 Metrics

In CL context, the model is trained with a continually updating data stream of multiple tasks. Hence, it is necessary to conduct a comprehensive evaluation which not only assesses the model's capability to adapt to new information but also to retain past knowledge. In this experiment, we evaluate the performance of the model by focusing on the following metrics:

Accuracy by Tasks (ACT) is the accuracy of recognizing all classes trained so far in each task. This metric is used to indicate the overall performance of the model with both newly and previously learnt classes in all tasks.

New-Class Accuracy by Tasks (NCT) is the accuracy of recognizing new activities in the current task. This metric helps evaluating plasticity: the model's ability to learn new information.

Old-Class Accuracy by Tasks (OCT) is the accuracy of recognizing all old classes which have been learnt from the previous tasks. It is a measure of the model's stability: the model's ability to retain knowledge of previously learned classes when trained on new tasks.

Forgetting Score by Tasks (FS) [4] measures the degree to which a model's performance on previous tasks degrades as it learns new tasks. The forgetting score at task k, FS_k, is computed as:

$$FS_k = \frac{1}{k-1}\sum_{j=1}^{k-1} f_j^k$$

where $f_j^k = 1 - \frac{A_j^k}{\max_{i\in\{1,\ldots,k-1\}} A_j^i}$ and A_j^k is the accuracy on the classes learnt at task j with the model trained up to task k. Intuitively this metric gives the average performance loss across all tasks seen. A forgetting score of 1 thus indicates that all performance has been lost (the model entirely forgot) and a forgetting score of 0 indicates that the model is the best performing so far across all tasks seen (no forgetting).

5 Experimental Settings

5.1 Datasets

Experiments in this study were conducted using the UCI HAR Dataset [26]. This dataset is an open dataset in which the data is gathered from inertial sensors, including accelerometers and gyroscopes, across the x, y, and z axes. The specifications of the dataset are depicted in Table 1, and the frequency of each activity in the dataset is presented in Table 2.

5.2 Sampling Process from the Real Data

In CL tasks, due to the limited storage capacity, it is very important to select the samples which are the good representatives for each class. With random sampling, the samples from each class are randomly selected based on the defined number of samples. This sample selection process is also applied to the case of EWC Replay. However, for iCaRL and LUCIR, following the methodology originally conducted in [24] and [11], herding sampling [34] is applied to select the samples to train the model.

Table 1. Specifications of UCI HAR Dataset

Number of participants	30
Device	Smartphone (Samsung Galaxy SII)
Collected data	3-axial linear data (x, y, z) from accelerometers and gyroscopes
Sampling rate	50Hz
Year	2012
Number of classes	6

Table 2. Frequency of each activity in UCI HAR dataset

Description	Activity	# samples
Walking	0	1722
Walking Up	1	1544
Walking Down	2	1406
Sitting	3	1777
Standing	4	1906
Laying	5	1944

5.3 Implementation Details

Defining exemplar sizes for the experiment We use exemplar set sizes k of (10, 14, 17, 21, and 25) for random sampling process. For iCaRL, which necessitates a fixed memory size [24], we determine the exemplar size by dividing the total memory by the current number of classes in each task. As indicated by Table 3, the total size of VAE is more efficient than larger real samples (17, 21, and 25) across all tasks, yet we also explore performance with smaller real data sizes (10 and 14) for baseline comparison. This analysis aims to determine if VAE-generated data can match or outperform the use of more substantial real data in CL training, given the similar amount of space in each task. This provides insights of using generated data over real data, considering the trade-offs between sample quantity and quality in the training process.

Table 3. Size of real data as exemplars and VAE across all tasks in Kilobytes (KB).

	Task 1	Task 2	Task 3	Total (KB)
Real sample (k=17)	206	410	614	1230
Real sample (k=21)	254	506	758	1518
Real sample (k=25)	301	602	901	1804
VAE Model	392	392	392	1176

CL Training with real data and generated data as exemplars With real data, random sampling is used in EWC Replay while herding sampling is applied in iCaRL and LUCIR to follow the principle of each CL approach. With generated data from VAE, we align the generated data size per class to the average training data size for each new task, ensuring balanced training sets. In the filtering process of the generated sample, a number of predicted probabilities p from VAE classifier within $[0.75, 0.97]$ are also used as the minimum probability for sample selection. The CL classifier demonstrates the best performance in $p = 0.80$. Hence, these results are included in our comparative analysis of CL strategies. More details on the training parameters of the implementation of CL training are described in Table 4.

Table 4. Parameter setting for the experiment

Parameters	**Value**
Learning Rate	0.0005
Batch Size	64
Number of Epochs	20
Latent Space Dimension (VAE)	64
Coefficient of Reconstruction Loss (VAE)	1
Coefficient of KL Divergence Loss (VAE)	0.001
Coefficient of Classification Loss (VAE)	1

6 Results

Fig 4 illustrates the all-class accuracy across tasks. The result reveals that random sampling, EWC Replay, and iCaRL have comparable accuracy, ranging from 53% to 60% in Task 2 and from 38% to 45% in Task 3. LUCIR demonstrates a lower accuracy in Task 2 with accuracies between 51% and 56%, but sees a notably higher accuracy in Task 3, increasing to the range of 42% and 45%. The VAE with boundary box technique outperforms others, maintaining around 62% accuracy in Task 2 and approximately 48% in Task 3. The filtering process of the VAE further enhancing accuracy by about 4% and 2% in Task 2 and Task 3 respectively. In contrast, VAE with the adaptive boundary approach exhibits a decrease in accuracy, falling from 57% to just under 40% by Task 3.

6.1 Plasticity of the CL classifier

The new-class accuracy for each method is presented in Fig. 5. The data reveals that across all tasks, most sampling methods maintain a high accuracy rate, hovering from 92% to 97%. iCaRL shows a significant decline in the accuracy in Task 3, falling from roughly 92% in Task 2 to just around 84% in Task 3. Meanwhile, LUCIR, which has a lower accuracy of approximately 85% in Task 2, displays a significant improvement by reaching around 93% in Task 3.

6.2 Stability of the CL classifier

The old-class accuracy is shown in Fig 6. Despite a significant decline in the accuracy in all sampling methods from Task 2, VAE Boundary Box with filtering maintains the highest accuracy when transitioning to new tasks. Specifically, it achieves an accuracy of about 45% accuracy in Task 2 and nearly 30% in Task 3. This represents gains of approximately 12% and 3% in Task 2 and Task 3, respectively, compared to its non-filtered counterpart. Additionally, LUCIR has consistently outperformed iCaRL by approximately 2-4% in accuracy across each task. Moreover, it is also noticeable that the accuracy of the VAE with adaptive boundary sampling method is remarkably low, on par with the performance when using 10 and 14 real-data exemplars.

This is also explained by Fig 7 where using adaptive boundary as a sampling method in VAE shows a higher forgetting score compared to other methods. In this measure, both the filtered and non-filtered versions of the VAE Boundary Box achieved significantly lower forgetting scores compared to other methods, at just 48% and 62% respectively. In Task 3, the filtered VAE Boundary Box outperforms all other methods with a significantly lower forgetting score of about 65%.

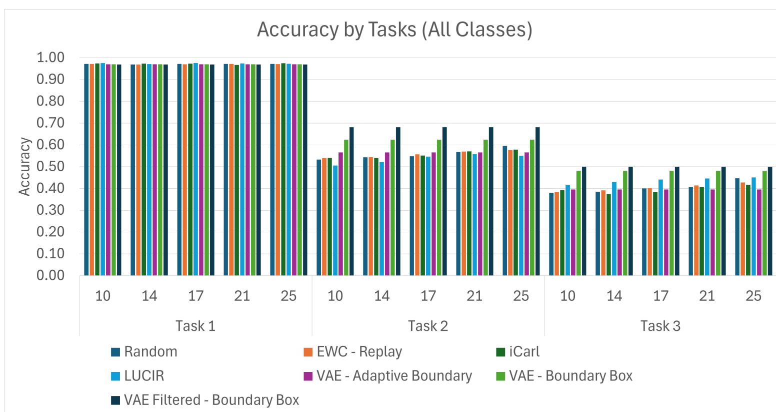


Fig. 4. All-class accuracy by tasks comparison between methods involving real data sampling (Random, EWC-Replay, iCaRL, and LUCIR) in different sample size k (10, 14, 17, 21, 25) and the methods using generated data from VAE (Adaptive boundary, boundary box with and without filtering process)

6.3 Discussion

Our findings show that the choice of replay strategy for the CL classifier plays a crucial role in the plasticity and stability performance of the model. Our analysis

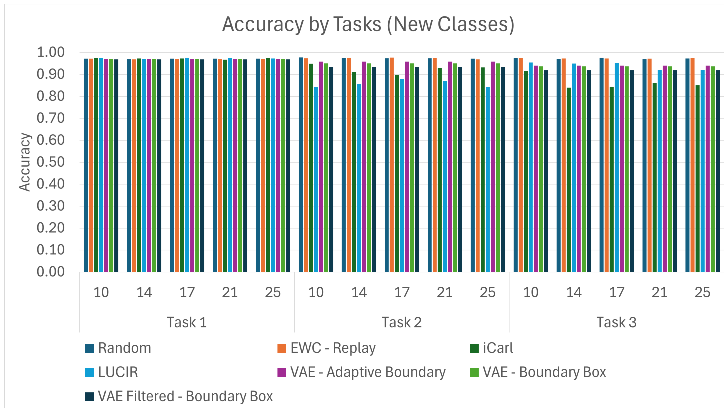


Fig. 5. New-class accuracy by tasks comparison between methods involving real data sampling (Random, EWC-Replay, iCaRL, and LUCIR) in different sample size k (10, 14, 17, 21, 25) and the methods using generated data from VAE (Adaptive boundary, boundary box with and without filtering process)

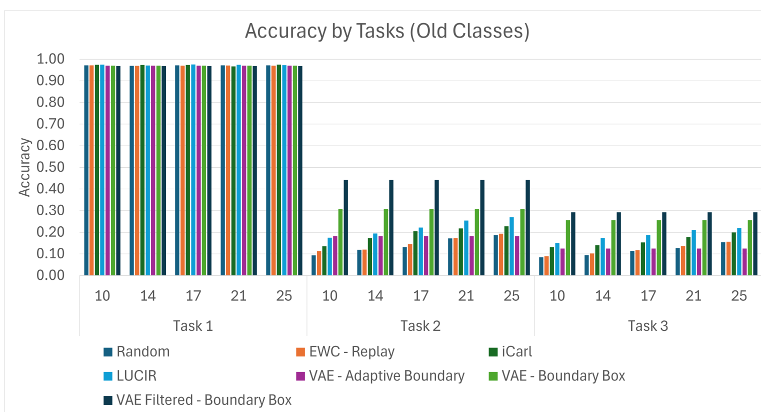


Fig. 6. Old-class accuracy by tasks comparison between methods involving real data sampling (Random, EWC-Replay, iCaRL, and LUCIR) in different sample size k (10, 14, 17, 21, 25) and the methods using generated data from VAE (Adaptive boundary, boundary box with and without filtering process)

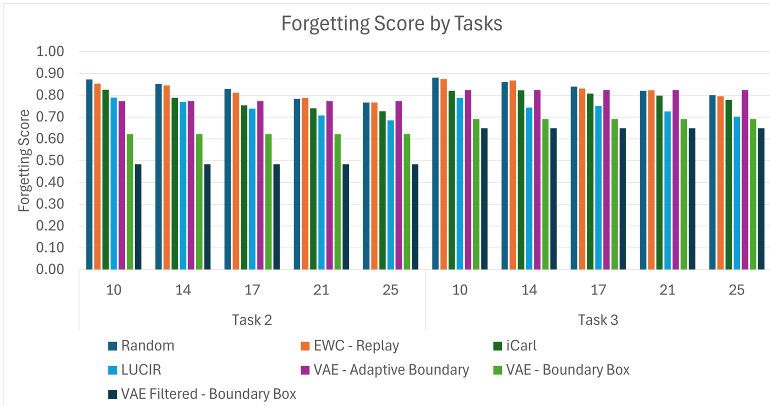


Fig. 7. Forgetting score by tasks (lower is better) comparison between methods involving real data sampling (Random, EWC-Replay, iCaRL, and LUCIR) in different sample size k (10, 14, 17, 21, 25) and the methods using generated data from VAE (Adaptive boundary, boundary box with and without filtering process)

across various tasks shows that generative replay with VAE outperforms experience replay methods via exemplars with equivalent memory footprint. The all-class accuracy depicted in Fig 4 shows the VAE with boundary box sampling as a superior technique, consistently outperforming others across all tasks. This implies that VAE-generated samples can effectively complement real data, ensuring a diversified and representative training set that supports the classifier on learning new classes incrementally. Additionally, applying a filtering process based on the confidence of VAE classifiers refine model performance by supplying more reliable generated data.

Moreover, the new-class and old-class accuracy results from Fig 5 and 6 demonstrate plasticity-stability performance of the model which address the critical balance between preserving old knowledge and acquiring new information in continual learning models. In the case of hybrid CL approaches such as iCaRL and LUCIR, despite being able to maintain a better old-class accuracy and forgetting score across tasks compared to most of the other techniques, it has a lower accuracy in new-class accuracy in each task. This trend exemplifies the plasticity-stability trade-off inherent in both approaches, where the safeguarding of old information may be prioritized at the expense of optimally learning new data. From our experiments, VAE shows its capability to deal with this issue by improving the knowledge retention of old tasks without having to sacrifice the learning of activities in new tasks.

Furthermore, our study highlights the role of sampling methods in the latent space of VAEs, revealing how boundary box sampling captures a broader diversity in the latent space than the adaptive boundary method, which has a higher forgetting score indicating a focus on dominant features over diversity. This diversity is crucial for smartphone sensor-based HAR, where individual variability is significant, influenced by health condition and age factors [32]. The boundary box sampling shows its potential by covering a wider range in the latent space, ensuring the generation of diverse samples. However, despite a notably low forgetting score in the second task, the VAE Boundary Box performance aligns with other methods by Task 3, indicating a potential area for improvement.

One research avenue could be the development of smart strategies for enhancing both the data representation quality of the generated data and its labeling. Our study shows that selectively filtering samples based on a classifier's minimum prediction probability enhances model performance. More advanced techniques, such as clustering in the latent space to identify and select core data points close to the centroids, could latent vector generation. The use of a GMM to sample latent vectors may also lead to a more accurate data generation.

7 Conclusion and Future Work

In conclusion, our findings highlight the efficiency of using VAE as a generative model to support replay based CL methods. The results reveal that VAEs, particularly with boundary box sampling, remarkably enhance the model performance across tasks compared to examplar based methods. This superiority suggests that VAE-generated samples can effectively augment real data, fostering a training set that can enhance the classifier's learning capabilities in CL scenario. Moreover, our analysis of plasticity-stability performance across different tasks illustrates the effectiveness of the VAE in maintaining a balance between preserving old knowledge and assimilating new information, a crucial aspect of CL models. As demonstrated in our study, the improvement in model performance from the refining process applied to the generated data could be a good initiative for future research work focused enhancing the quality of data produced by generative models.

For future work, we aim to delve into both advanced sampling strategies from the latent space and latent space optimization to enhance the diversity and quality of data generated for CL models. By exploring innovative methods for sampling from the latent space, we aim to improve the fidelity and variety of synthetic data.

References

1. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: Learning what (not) to forget. In: Proceedings of the European conference on computer vision (ECCV). pp. 139–154 (2018)
2. Bachlin, M., Roggen, D., Troster, G., Plotnik, M., Inbar, N., Meidan, I., Herman, T., Brozgol, M., Shaviv, E., Giladi, N., et al.: Potentials of enhanced context awareness in wearable assistants for parkinson's disease patients with the freezing of gait syndrome. In: 2009 International Symposium on Wearable Computers. pp. 123–130. IEEE (2009)
3. Bulling, A., Blanke, U., Schiele, B.: A tutorial on human activity recognition using body-worn inertial sensors. ACM Computing Surveys (CSUR) **46**(3), 1–33 (2014)

4. Chaudhry, A., Dokania, P.K., Ajanthan, T., Torr, P.H.: Riemannian walk for incremental learning: Understanding forgetting and intransigence. In: Proceedings of the European conference on computer vision (ECCV). pp. 532–547 (2018)
5. Chavarriaga, R., Sagha, H., Calatroni, A., Digumarti, S.T., Tröster, G., Millán, J.d.R., Roggen, D.: The opportunity challenge: A benchmark database for on-body sensor-based activity recognition. Pattern Recognition Letters **34**(15), 2033–2042 (2013)
6. Chen, T., Goodfellow, I., Shlens, J.: Net2net: Accelerating learning via knowledge transfer. arXiv preprint arXiv:1511.05641 (2015)
7. Chua, S.L., Foo, L.K., Guesgen, H.W., Marsland, S.: Incremental learning of human activities in smart homes. Sensors **22**(21), 8458 (2022)
8. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. Communications of the ACM **63**(11), 139–144 (2020)
9. Hammerla, N.Y., Halloran, S., Plötz, T.: Deep, convolutional, and recurrent models for human activity recognition using wearables. arXiv preprint arXiv:1604.08880 (2016)
10. He, J., Mao, R., Shao, Z., Zhu, F.: Incremental learning in online scenario. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 13926–13935 (2020)
11. Hou, S., Pan, X., Loy, C.C., Wang, Z., Lin, D.: Learning a unified classifier incrementally via rebalancing. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 831–839 (2019)
12. Jha, S., Schiemer, M., Zambonelli, F., Ye, J.: Continual learning in sensor-based human activity recognition: An empirical benchmark analysis. Information Sciences **575**, 1–21 (2021)
13. Kemker, R., McClure, M., Abitino, A., Hayes, T., Kanan, C.: Measuring catastrophic forgetting in neural networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 32 (2018)
14. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
15. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences **114**(13), 3521–3526 (2017)
16. Lara, O.D., Labrador, M.A.: A survey on human activity recognition using wearable sensors. IEEE communications surveys & tutorials **15**(3), 1192–1209 (2012)
17. Lavda, F., Ramapuram, J., Gregorova, M., Kalousis, A.: Continual classification learning using generative models. arXiv preprint arXiv:1810.10612 (2018)
18. Li, Z., Hoiem, D.: Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence **40**(12), 2935–2947 (2017)
19. Lopez-Paz, D., Ranzato, M.: Gradient episodic memory for continual learning. Advances in neural information processing systems **30** (2017)
20. Minhas, R., Mohammed, A.A., Wu, Q.J.: Incremental learning in human action recognition based on snippets. IEEE Transactions on Circuits and Systems for Video Technology **22**(11), 1529–1541 (2011)
21. Nguyen, C.V., Li, Y., Bui, T.D., Turner, R.E.: Variational continual learning. arXiv preprint arXiv:1710.10628 (2017)
22. Ostapenko, O., Puscas, M., Klein, T., Jahnichen, P., Nabi, M.: Learning to remember: A synaptic plasticity driven framework for continual learning. In: Proceedings

of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11321–11329 (2019)
23. Rakaraddi, A., Siew Kei, L., Pratama, M., De Carvalho, M.: Reinforced continual learning for graphs. In: Proceedings of the 31st ACM International Conference on Information & Knowledge Management. pp. 1666–1674 (2022)
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. Reiss, A., Stricker, D.: Introducing a new benchmarked dataset for activity monitoring. In: 2012 16th international symposium on wearable computers. pp. 108–109. IEEE (2012)
26. Reyes-Ortiz, J., Anguita, D., Ghio, A., Oneto, L., Parra, X.: Human Activity Recognition Using Smartphones. UCI Machine Learning Repository (2012), DOI: https://doi.org/10.24432/C54S4K
27. Rios, A., Itti, L.: Closed-loop gan for continual learning. arXiv preprint arXiv:1811.01146 (2018)
28. Ros, M., Cuéllar, M.P., Delgado, M., Vila, A.: Online recognition of human activities and adaptation to habit changes by means of learning automata and fuzzy temporal windows. Information Sciences **220**, 86–101 (2013)
29. Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R.: Progressive neural networks. arXiv preprint arXiv:1606.04671 (2016)
30. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. Advances in neural information processing systems **30** (2017)
31. Sun, L., Zhang, M., Wang, B., Tiwari, P.: Few-shot class-incremental learning for medical time series classification. IEEE Journal of Biomedical and Health Informatics (2023)
32. Thu, N.T.H., Han, D.S.: Hihar: A hierarchical hybrid deep learning architecture for wearable sensor-based human activity recognition. IEEE Access **9**, 145271–145281 (2021)
33. Van de Ven, G.M., Tolias, A.S.: Generative replay with feedback connections as a general strategy for continual learning. arXiv preprint arXiv:1809.10635 (2018)
34. Welling, M.: Herding dynamical weights to learn. In: Proceedings of the 26th annual international conference on machine learning. pp. 1121–1128 (2009)
35. Ye, J., Nakwijit, P., Schiemer, M., Jha, S., Zambonelli, F.: Continual activity recognition with generative adversarial networks. ACM Transactions on Internet of Things **2**(2), 1–25 (2021)
36. Ye, J., Stevenson, G., Dobson, S.: Detecting abnormal events on binary sensors in smart home environments. Pervasive and Mobile Computing **33**, 32–49 (2016)