Blurry Class Incremental Learning for IMU-Based Human Activity Recognition: An Empirical Study

Takumi Yamamoto[1,2], Suguru Kanoga[1], Mitsunori Tada[1], Yuta Sugiura[2]
[1]National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan
[2]Keio University, Yokohama, Japan
Email: imuka06x17@keio.jp, s.kanoga@aist.go.jp, Add Tada san's address, sugiura@keio.jp

***Abstract*—As inertial motor unit (IMU)-based human activity recognition (HAR) have attracted particular attention, the demand for long-term use of the system is increasing. In long-term usage scenarios, user needs may evolve, potentially requiring the model to recognize additional classes. Class Incremental Learning (CIL) provides a promising solution by enabling models to learn additional classes without retraining from scratch. While prior work has explored Class-Incremental Learning (CIL) in HAR, it has not considered scenarios in which same classes appear in the different task—namely, the Blurry Class Incremental Learning (B-CIL) scenario. In this study, we explore the B-CIL scenario for IMU-based HAR, and conduct extensive experiments on two public IMU datasets (UCI-HAR and USC-HAD), comparing nine continual learning methods across multiple overlap class configurations. Our results show that replay-based methods outperform regularization-based approaches under the B-CIL scenario. Furthermore, we find that increasing the number of overlapping classes can improve performance.**

***Index Terms*—human activity recognition, inertial measurement unit, continual learning**

I. Introduction

Human activity recognition (HAR) aims to classify daily human activities based on sensor data and has gained attention in recent years [1]–[3]. Among various sensing modalities, inertial measurement unit (IMU)-based HAR has been widely adopted for applications such as smart homes [4], healthcare [5], input interfaces [6], sports training [7], and fall detection [8]. As these applications are increasingly used over extended periods [9], HAR systems must adapt to new user behaviors and evolving activity classes [10]. A naive retraining approach using all available data is often infeasible due to resource constraints and privacy concerns [11]. When using plasticity-focused methods like fine-tuning, the model often suffers from catastrophic forgetting [12]–[14], where performance on prior tasks deteriorates due to overfitting to new data. Consequently, there is growing interest in continual learning (CL) methods that allow updating models using only data from newly introduced classes. CL approaches–typically categorized into regularization, replay, and parameter isolation–enable models to learn incrementally without forgetting previous tasks [15], [16].

In CL, there are various problem scenarios depending on how tasks arrive sequentially. One of the scenario is class incremental learning (CIL), in which the model must learn an increasing number of classes over time. In the typical CIL scenario, the model learn them without task labels and with no class repetition across tasks (Fig. 1(A)). However, in real-world systems, data is not always collected under conditions where classes are fully independent across tasks. Blurry class incremental learning (B-CIL) relaxes the constraint of disjoint classes across tasks, allowing class overlap and more close to real-world usage scenarios (Fig. 1(B)).

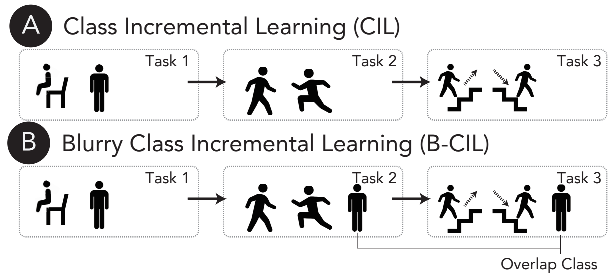
A
Class Incremental Learning (CIL)
Task 1
Task 2
Task 3
B
Blurry Class Incremental Learning (B-CIL)
Task 1
Task 2
Task 3
Overlap Class

Fig. 1. Overview of continual learning scenarios: (A) CIL, (B) B-CIL.

While CIL has been studied in HAR [17]–[24], most prior work assumes non-overlapping classes. The only existing related work is the work by Zhang et al. [25]. HarMI assumes a problem setting where one new class is added at each task, and later each class reappears once [25]. However, under this condition, old and new classes do not coexist within the same task. Thus, B-CIL has not been thoroughly investigated under more diverse conditions for IMU-based HAR.

In this study, we evaluate nine continual learning methods under B-CIL scenarios with varying degrees of class overlap. In the experiments, we used two public datasets–the University of California Human Activity Recognition (UCI-HAR) dataset and the University of Southern California Human Activity Dataset (USC-HAD). Our results show that replay-based methods outperform regularization-based ones, and that increasing class overlap improves performance.

Our research questions are as follows:

- **RQ1:** Which CL methods are most effective in B-CIL scenarios for IMU-based HAR?
- **RQ2:** How does the degree of class overlap affect the performance of CL methods in B-CIL scenarios?

II. Problem Definitions and Scenarios

In a CL scenario, tasks are assumed to arrive sequentially, and the model $f(\boldsymbol{x};\theta)$ is incrementally updated upon the arrival of each new task, where $\boldsymbol{x}$ denotes the input data and θ represents the model parameters. Formally, the sequence of tasks is defined as $T = \{\tau^1, \tau^2, ..., \tau^N\}$, where each task τ^t ($t \in \{1, ..., N\}$) is associated with a dataset $D^t = \{(x_i^t, y_i^t) | y_i^t \in Y^t\}_{i=1}^{N_t}$ and a corresponding label space Y^t. Each input x_i^t is sampled from a task-specific input domain $\mathcal{X}^t$, and each label y_i^t belongs to the respective label space Y^t. We denote the model before training on task τ^t as f_{θ_t}, and the model after optimization as $f_{\theta_t^*}$.

The formal definitions of the continual learning scenarios considered in this study are as follows:

- CIL: Each task introduces a set of new, mutually exclusive classes. Formally, for any $i \neq j$, the label spaces are disjoint, i.e., $Y^i \cap Y^j = \varnothing$, indicating that no class is shared between tasks.
- B-CIL: This relaxes the strict disjoint class assumption of CIL by allowing partial overlap between the label spaces of tasks. Specifically, for some $i \neq j$, it holds that $Y^i \cap Y^j \neq \varnothing$.

III. Methods

A. Backbone Model

As the backbone model, we employed one-dimensional convolutional neural network (1D CNN) architecture commonly used in prior class incremental learning studies [20]. Specifically, we adopted the same architecture proposed in [20], which comprises four convolutional blocks. Each block consists of a convolutional layer (kernel size: 5, stride: 1, padding: 2) with output channels of 64, 128, 256, and 128, respectively, followed by a ReLU activation function, a batch normalization (BN) layer, and a MaxPooling layer (kernel size: 2, stride: 2).

B. Baselines

As baselines, we considered two approaches. The first is the naïve fine-tuning strategy, where the model is incrementally updated on each new task without employing any CL methods, thereby serving as a reference for catastrophic forgetting. The second is the offline setting, where the model is trained using data from all tasks simultaneously. This serves as an upper-bound performance reference, as it assumes access to the entire dataset across tasks.

C. Continual Learning Methods

1) Regularization-based methods: We selected four representative regularization-based CL methods for comparison: Learning without Forgetting (LwF) [26], Elastic Weight Consolidation (EWC) [27], Memory Aware Synapses (MAS) [28], and Synaptic Intelligence (SI) [29]. These methods mitigate catastrophic forgetting by incorporating additional loss terms that constrain updates to important model parameters based on knowledge acquired from previous tasks.

LwF [26] mitigates catastrophic forgetting by leveraging a knowledge distillation (KD) loss as a regularization term. Specifically, LwF encourages the current model $f_{\boldsymbol{\theta}}$ to retain knowledge from previous tasks by minimizing the discrepancy between its output and that of a frozen model $f_{\boldsymbol{\theta}^*}$–trained on earlier tasks–when evaluated on the current task data. This is achieved in conjunction with minimizing the standard cross-entropy (CE) loss $\mathcal{L}_{\text{CE}}$ for the current task labels. In this study, we adopt the adaptive-weight variant of LwF, where the contribution of the CE and KD losses varies depending on the number of tasks seen so far. The total loss function for task τ^t is defined as:

$$\mathcal{L} = \frac{1}{t} \cdot \mathcal{L}_{\text{CE}} + \left(1 - \frac{1}{t}\right) \cdot \lambda \cdot \mathcal{L}_{\text{KD}^t} \quad (1)$$

$$\mathcal{L}_{\text{KD}}^t = - \sum_{c \in Y^{1:t-1}} \tilde{y}_c^{t-1} \log \tilde{y}_c^t \quad (2)$$

$$\tilde{y}_i^{t-1} = \text{Softmax}(f_{\theta_{t-1}^*}(x_i^t)/T),$$

$$\tilde{y}_i^t = \text{Softmax}(f_{\theta_t}(x_i^t)/T) \quad (3)$$

where T is the temperature parameter used for smoothing the output distributions, and is set to 2 following [30].

EWC [27] addresses catastrophic forgetting by selectively constraining updates to parameters deemed important for previously learned tasks. The importance of each parameter is quantified using the diagonal elements of the Fisher Information Matrix (FIM), which captures the sensitivity of the loss with respect to each parameter. The total loss function for task τ^t is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \frac{\lambda}{2} \sum_{c \in Y^{1:t}} F_c (\boldsymbol{\theta}_{t,c} - \boldsymbol{\theta}_{t-1,c}^*)^2, \quad (4)$$

where F_c represents the importance score of parameter θ_c computed from previous tasks, $\boldsymbol{\theta}_t$ is the current model parameter vector, and $\boldsymbol{\theta}_{t-1}^*$ is the optimized parameter vector from the previous task. The regularization term penalizes deviations from the previously important parameters, thereby preserving prior knowledge.

MAS [28] is a regularization-based approach inspired by neuroplasticity, which quantifies the importance of each parameter based on its contribution to the model's output. Unlike EWC, which relies on the FIM, MAS estimates importance by computing the magnitude of the gradients of the model outputs with respect to the parameters. The total loss function for task τ^t is given by:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \lambda \sum_{c \in Y^{1:t}} \Omega_c (\boldsymbol{\theta}_{t,c} - \boldsymbol{\theta}_{t-1,c}^*)^2, \quad (5)$$

$$\Omega_c = \frac{1}{N} \sum_{i=1}^{N} || \frac{\partial (f(x_i;\theta))}{\partial \theta_c} ||, \quad (6)$$

where Ω_c denotes the estimated importance of parameter θ_c and is computed as the average norm of the gradients over N samples. This regularization discourages large updates to parameters that significantly influence the model's predictions, thereby preserving knowledge acquired from previous tasks.

SI [29] addresses catastrophic forgetting by penalizing changes to parameters that were important for solving previous tasks. Similar to MAS, SI estimates the importance of each parameter based on its historical contribution to reducing the loss. Unlike methods that rely on second-order information, SI computes importance online during training by tracking the product of the gradient and the parameter update. The loss function for task τ^t is defined as:

$$\mathcal{L} \;=\; \mathcal{L}_{\mathrm{CE}} + \lambda \sum_{c \in Y^{1:t}} \omega_c (\boldsymbol{\theta}_{t,c} - \boldsymbol{\theta}^*_{t-1,c})^2, \tag{7}$$

where ω_c represents the estimated importance of parameter θ_c and is computed as:

$$\omega_c = \sum_t \frac{g_c^{(t)} \cdot \Delta\theta_c^{(t)}}{(\Delta\theta_c^{(t)})^2 + \epsilon}, \tag{8}$$

with $g_c^{(t)}$ denoting the gradient of the loss with respect to parameter θ_c at training step t, $\Delta\theta_c^{(t)}$ the corresponding parameter update, and ϵ a small constant added for numerical stability. This approach allows SI to adaptively preserve important knowledge while enabling learning on new tasks.

2) Replay-based methods: We selected five representative replay-based methods for comparison: Experienced Replay (ER) [31], Dark Experience Replay (DER) [32], Fast Incremental Classifier and Representation Learning (FastICARL) [33], Adversarial Shapley value Experience Replay (ASER) [34], and Generative Replay (GR) [35].

ER [31] is a fundamental replay-based method in which a fixed-size memory buffer is used to store samples from previous tasks. During training on a new task, the model is updated using a combination of current task data and replayed samples drawn from the memory buffer. In our implementation, we adopted random sampling strategies for both memory retrieval and memory update, consistent with the original ER framework. The overall loss function used in task τ^t is defined as:

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \mathcal{L}_{\mathrm{Buffer}}, \tag{9}$$

where $\mathcal{L}_{\mathrm{Buffer}}$ denotes the cross-entropy loss computed on the replayed samples from the memory buffer.

DER [32] extends the standard ER framework by incorporating a KD loss. Like ER, DER employs random sampling strategies for both memory retrieval and update. In addition to the CE loss on buffered samples, DER introduces a distillation loss that encourages the current model to mimic the output of the previous model on the same buffered inputs. The overall loss function is defined as:

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \frac{1}{2} \cdot \mathcal{L}_{\mathrm{Buffer}} + \frac{1}{2} \cdot \mathcal{L}_{\mathrm{Buffer_{KD}}}, \tag{10}$$

where $\mathcal{L}_{\mathrm{Buffer}}$ denotes the CE loss on memory samples, and $\mathcal{L}_{\mathrm{Buffer_{KD}}}$ represents the KD loss computed with respect to the soft targets (i.e., logits or softmax outputs) stored in the buffer.

FastICARL [33] is a computationally efficient variant of the original iCaRL method [36], a class-incremental learning approach that integrates exemplar memory with a nearest-mean-of-exemplars classifier. FastICARL addresses the high computational cost of iCaRL by replacing its k-nearest neighbor (k-NN)–based exemplar selection with a more efficient max-heap–based strategy. The model is trained by minimizing the standard CE loss over the combined set of current task data and stored exemplars, along with a KD loss:

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \lambda \cdot \mathcal{L}_{\mathrm{Buffer_{KD}}}. \tag{11}$$

ASER [34] is a replay-based CL method that prioritizes samples in the memory buffer based on their Shapley values. This approach aims to identify and replay samples that are not only representative of previous classes but also likely to interfere with the learning of new incoming data. ASER achieves this by computing adversarial Shapley values (ASV), which quantify the trade-off between cooperative and adversarial contributions of each sample to model performance under distributional shifts or adversarial scenarios.

$$\mathrm{ASV}(i) = \frac{1}{|S_{\text{sub}}|} \sum_{j \in S_{\text{sub}}} s_j(i) - \frac{1}{|B|} \sum_{k \in B} s_k(i), \tag{12}$$

where S_{sub} denotes a class-balanced subset of the replay buffer, B represents the current memory buffer, and $s_j(i)$ is the Shapley value of sample i with respect to sample j. The first term captures the cooperative contribution of sample i to the subset S_{sub}, while the second term measures its interference with the current batch B. Due to the computational complexity of exact Shapley value estimation, ASV is typically approximated using k-nearest neighbors. In this study, we follow [20] and set $k = 3$.

GR [35] is a replay-based CL method that synthesizes pseudo-samples resembling previous task data using a generative model. These generated samples are then replayed alongside new task data during training to mitigate catastrophic forgetting. In GR, a generator g_ϕ and a learner f_θ are trained jointly. At each task step, the generator from the previous task generates synthetic samples, while the learner assigns pseudo-labels to these samples. The model is then trained on a combination of the generated data and the current task data. After updating the learner, the generator is also retrained to reflect the updated knowledge. In this study, we adopt TimeVAE [37] as the generative model, which is well-suited for sequential and time-series data generation.

IV. Materials

We used two public dataset; (i) UCI-HAR, and (ii) USC-HAD.

UCI-HAR [38] contains data from 30 subjects, collected using a waist-mounted smartphone (Samsung Galaxy S II) with embedded inertial sensors. The sensor's sampling rate was set to 50 Hz. The dataset includes six activities: (1) walking, (2) walking upstairs, (3) walking downstairs, (4) sitting, (5) standing, and (6) lying. The data has nine feature dimensions, which consist of 3-axis total acceleration, 3-axis estimated body acceleration, and 3-axis angular velocity. Sensor signals were preprocessed using sliding windows of 2.56 s and 50 % overlap, resulting in window shapes of 128×9.

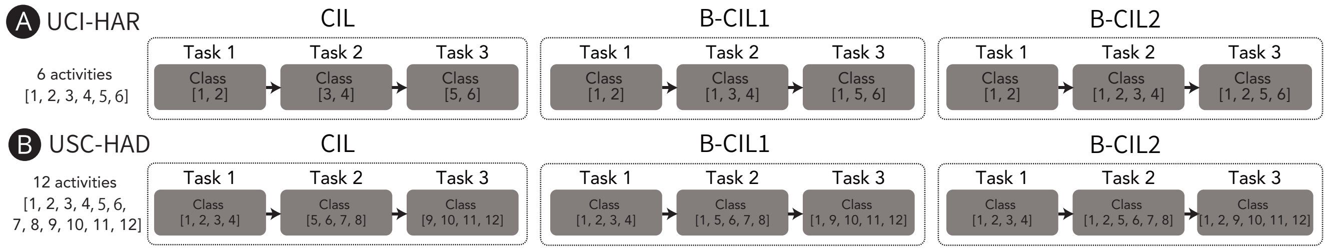

Fig. 2. The scenarios of continual learning in the (A) UCI-HAR and (B) USC-HAD datasets.

USC-HAD [39] contains data from 14 subjects, collected using a single IMU sensor (MotionNode) positioned on the front right side of the body. The sensor's sampling rate was set to 100 Hz. The dataset includes 12 activities: (1) walking forwards, (2) walking left, (3) walking right, (4) walking upstairs, (5) walking downstairs, (6) running forwards, (7) jumping, (8) sitting, (9) standing, (10) sleeping, (11) elevator up, and (12) elevator down. The data has six feature dimensions, which consist of 3-axis acceleration and 3-axis angular velocity. Sensor signals were preprocessed using sliding windows of 1.28 s and 50% overlap, resulting in window shapes of 128×6.

V. Experiments

In this study, we evaluated two B-CIL scenarios (B-CIL1 and B-CIL2), and compared them with the CIL scenarios. Fig 2 shows the detail of scenarios of each two dataset, (A) UCI-HAR and (B) USC-HAD. In this study, the number of tasks was set to three.

In the CIL scenario, the classes from each dataset are evenly assigned to each task to ensure that there is no overlap of classes across tasks. Consequently, in the UCI-HAR dataset, which has six classes, each task is assigned two classes, while in the USC-HAD dataset, which has twelve classes, each task is assigned four classes. In the B-CIL1 scenario, compared to the CIL scenario, one of the classes from the first task (Task 1) is also assigned to other tasks (Task 2 and Task 3). As a result, the number of classes per task is [2, 3, 3] for the UCI-HAR dataset and [4, 5, 5] for the USC-HAD dataset. In the B-CIL2 scenario, two classes from the task 1 are assigned to both Tasks 2 and 3. Therefore, the number of classes per task is [2, 4, 4] for the UCI-HAR dataset and [4, 6, 6] for the USC-HAD dataset. The order of classes in each task was determined randomly. The split into training, validation, and test datasets was also performed randomly. Sixty percent of the data was used for training, twenty percent for validation, and twenty percent for testing.

A. Evaluation Metrics

We used the (1) Final Average Accuracy, (2) Final Average Forgetting, and (3) Average Learning Accuracy as the evaluation metrics. Let $a_{i,j}$ denote the average classification accuracy when the model, after being trained on Task i, is tested on Task $j \leqq i$. The total number of task is T.

(1) Final Average Accuracy is the average accuracy of the model for all tasks when the model has finished learning all tasks.

$$\mathcal{A}_T = \frac{1}{T}\sum_{i=1}^{T} a_{T,i} \tag{13}$$

(2) Final Average Forgetting represents how much accuracy decreases on task j due to learning task T:

$$\mathcal{F}_T = \frac{1}{T-1}\sum_{j=1}^{T} \max_{k\in\{1,\ldots,T-1\}} (a_{k,j} - a_{i,j})(j < i) \tag{14}$$

(3) Average Learning Accuracy is the average accuracy of the model for the new task when the model has updated in that tasks:

$$\mathcal{A}_{cur} = \frac{1}{T}\sum_{i=1}^{T} a_{i,i} \tag{15}$$

B. Learning Protocol

For the training parameters, Cross Entropy Loss was used as the loss function, and the OneCycle learning rate scheduling was employed. The model was trained for 100 epochs. Early stopping was applied to regularization-based method and baseline method with the patience parameter; 5.

Common to all training methods, the batch size, maximum learning rate, and learning rate adjustment strategy were optimized using grid search. The search range for the batch size was [32, 64, 128], and for the initial learning rate, it was [0.01, 0.001, 0.0001]. For the learning rate adjustment strategy, three approaches were explored: reducing the learning rate by a factor of 0.1 every 10 epochs, reducing it by a factor of 0.1 every 25 epochs, and using only the scheduler without manual adjustments. These hyperparameters were optimized using only the validation data from Task 1. The hyperparameters for each classification algorithm are shown in Table I.

C. Implementation

We implemented the methods in Python 3.10.10 and PyTorch 1.13.1. These codes were executed on a machine running Ubuntu 22.04, equipped with four L40S GPU.

TABLE I
HYPERPARAMETER GRID SEARCH FOR THE CONTINUAL LEARNING METHODS

Method	parameter	Search
LwF	λ	1, 0.1, 0.01, 0.001, 0.0001
EWC	λ	0.01, 0.001, 0.0001
MAS	λ	0.001, 0.0001, 0.00001
SI	λ	0.01, 0.001, 0.0001
ASER	Number of sample saved in buffer per class	2, 4
GR	Learning rate for generator Weight for reconstruction loss	0.001, 0.0001 0.01, 0.1, 1, 10

VI. RESULTS

Table II shows the results of the continual learning methods on the UCI-HAR and USC-HAD datasets. When comparing the thrre scenarios in the two datasets, CIL had the lowest final average accuracy, followed by B-CIL1, with B-CIL2 having the highest, except for FastICARL in USC-HAD. In terms of Final Average Forgetting, CIL exhibited the highest level of forgetting. As the number of Overlap classes increased, the amount of forgetting decreased, with B-CIL2 showing the lowest forgetting (except for ASER in USC-HAD). Average Current Accuracy varied across algorithms, showing no consistent trend. These results indicate that increasing the number of Overlap classes helps reduce forgetting and ultimately improves final accuracy.

We compared the final average accuracy of continual learning algorithms with the baseline method, Naive. In the CIL scenario, most continual learning algorithms achieved higher average final accuracy except GR on USC-HAD. In the B-CIL1 scenario, continual learning algorithms outperformed Naive [Add]. In the B-CIL2 scenario, continual learning algorithms outperformed Naive except for SI on UCI-HAR and USC-HAD. Overall, in scenarios such as CIL and those with task overlap, many CL algorithms tend to outperform the Naive baseline.

Among the evaluated continual learning algorithms, most replay-based methods outperformed regularization-based methods in both datasets and across all scenario, except GR. Four replay-based methods (ER, DER, ASER, FastICARL) had an average Final Average Accuracy of 75.20 in CIL, 91.62 in B-CIL1, and 95.97 in B-CIL2 for UCI-HAR, and 77.15 in CIL, 82.88 in B-CIL1, and 84.76 in B-CIL2 for USC-HAD. In contrast, For regularization-based method (LwF, EWC, MAS, SI), the average Final Average Accuracy was 36.13 in CIL, 70.04 in B-CIL1, and 82.94 in B-CIL2 for UCI-HAR, and 37.05 in CIL, 55.40 in B-CIL1, and 61.37 in B-CIL2 for USC-HAD. Among the replay-based methods, ASER achieved the highest final average accuracy in the CIL, B-CIL1, and B-CIL2 scenarios for UCI-HAR and USC-HAD.

When comparing the UCI-HAR and USC-HAD datasets, in the CIL scenario, LwF, EWC, ER, and FastICARL showed higher Final Average Accuracy on the USC-HAD dataset. However, in the B-CIL1 and B-CIL2 scenarios, the UCI-HAR dataset exhibited higher accuracy. This may be influenced by the fact that the UCI-HAR dataset has fewer classes per task, resulting in a higher proportion of overlapping classes in B-CIL scenarios.

VII. DISCUSSION

A. Answers to Research Questions

1) RQ1: Which CL methods are most effective in B-CIL scenarios for IMU-based HAR?: In both UCI-HAR and USC-HAD datasets, replay-based methods generally outperformed regularization-based methods in B-CIL scenarios. This finding aligns with previous research in CIL scenarios [20], which also reported the superiority of replay-based methods. Among the replay-based method, ASER consistently achieved the highest final average accuracy across both datasets and all B-CIL scenarios.

2) RQ2: How does the degree of class overlap affect the performance of CL methods in B-CIL scenarios?: Increasing the degree of class overlap in B-CIL scenarios positively impacted the performance of CL methods. As the number of overlapping classes increased from B-CIL1 to B-CIL2, there was a notable reduction in forgetting and an improvement in final average accuracy across most CL methods. In addition, the higher the proportion of overlapping classes relative to existing classes, the more pronounced the improvement in accuracy when classes overlap.

B. Limitations and Future Work

1) Expansion to other datasets and algorithms: In this study, we used the UCI-HAR and USC-HAD datasets to evaluate the performance of continual learning algorithms in Blurry Class Incremental Learning (Blurry CIL) scenarios. A limitation of this study is that it considered only two datasets. In the future, we will extend this research to other human activity datasets to examine how the number and order of overlapping classes affect model accuracy.

To the best of our knowledge, no existing studies have applied a Blurry CIL scenario to time-series data. CIL has been explored not only for IMU data but also for other types of time-series data, such as electromyography (EMG) signals [40]. In the future, we plan to extend the Blurry CIL setting to other types of time-series data.

As a continual learning methods, we selected four regularization-based methods (LwF, EWC, MAS, SI) and five replay-based methods (ER, ASER, DER, FastICARL, GR). In our study, parameter isolated methods such as Progressive Neural Networks (PNN) [41] were not considered. We will include such parameter isolation methods in our future studies.

2) More various blurry scenarios: In this study, we trained and updated the model using batch learning. In real-world applications, data often arrives in a streaming format. As a future work, it is necessary to evaluate the performance of online continual learning [22], where the model is updated as data arrives in a streaming manner, under Blurry CIL scenarios.

In this study, we applied the Blurry problem setting to Class Incremental Learning (CIL). In contrast, Blurry problem

TABLE II
RESULTS OF THE CONTINUAL LEARNING METHODS IN THE UCI-HAR AND USC-HAD DATASETS (ORDER: FF)

Dataset	Metrics	Scenario	Naive	Offline	LwF	EWC	MAS	SI	ER	DER	ASER	FastICARL	GR
$\mathcal{A}_T$	UCI-HAR	CIL	33.04	95.98	34.35	35.36	40.47	34.35	70.97	66.62	**96.76**	66.44	36.34
		B-CIL1	61.16	N.A.	65.49	63.52	70.58	63.91	89.17	88.74	**98.86**	89.69	62.70
		B-CIL2	82.52	N.A.	83.15	83.44	84.60	80.56	96.00	92.83	**99.40**	95.64	83.49
	USC-HAD	CIL	32.82	57.71	38.72	37.15	38.69	33.64	76.94	64.42	**94.11**	73.13	29.23
		B-CIL1	49.07	N.A.	57.11	54.75	54.59	51.55	82.58	72.32	**95.67**	80.93	
		B-CIL2	58.04	N.A.	62.34	65.18	60.79	57.17	86.78	77.30	**96.28**	78.67	
$\mathcal{F}_T$	UCI-HAR	CIL	99.15	N.A.	91.35	95.44	85.20	88.95	19.80	17.06	**4.24**	29.51	
		B-CIL1	56.78	N.A.	46.46	53.12	39.78	46.72	7.38	5.43	**1.27**	8.06	
		B-CIL2	23.98	N.A.	19.42	22.66	18.24	20.89	2.81	3.53	**0.57**	2.97	
	USC-HAD	CIL	98.62	N.A.	77.66	88.55	78.80	70.18	12.36	9.16	**3.94**	14.32	
		B-CIL1	72.65	N.A.	55.73	62.03	55.33	50.61	8.38	6.32	**2.23**	11.47	
		B-CIL2	58.60	N.A.	45.21	44.56	46.31	41.62	5.35	4.90	**1.48**	16.43	
$\mathcal{A}_{cur}$	UCI-HAR	CIL	99.14	N.A.	96.75	98.98	97.27	93.65	81.37	76.74	**99.58**	85.54	
		B-CIL1	99.01	N.A.	96.46	98.84	97.10	94.06	93.31	92.03	**99.69**	95.03	
		B-CIL2	98.40	N.A.	96.08	98.46	96.74	94.48	97.78	95.06	**99.67**	97.52	
	USC-HAD	CIL	**97.90**	N.A.	90.49	96.18	91.22	80.43	85.01	68.95	96.73	82.64	
		B-CIL1	**97.50**	N.A.	94.27	96.10	91.48	85.21	88.07	75.81	97.15	88.53	
		B-CIL2	97.11	N.A.	92.48	94.89	91.67	84.80	90.26	79.71	**97.15**	89.61	

settings can also be applied to Domain Incremental Learning (Domain IL) [42], [43], where the domain changes rather than the classes. As a future work, it is necessary to apply the Blurry problem setting to Domain IL and evaluate its effectiveness.

VIII. CONCLUSION

In this study, we investigated the effectiveness of various continual learning algorithms in Blurry Class Incremental Learning (B-CIL) scenarios for IMU-based Human Activity Recognition (HAR). We evaluated nine continual learning algorithms, including four regularization-based methods (LwF, EWC, MAS, SI) and five replay-based methods (ER, ASER, DER, FastICARL, GR), on two public datasets: UCI-HAR and USC-HAD. Our experiments revealed that replay-based methods generally outperformed regularization-based methods in B-CIL scenarios, and that increasing the degree of class overlap positively impacted model performance by reducing forgetting and improving final average accuracy. In the future, we plan to extend our research to other datasets and explore more realistic Blurry scenarios, such as online continual learning.

REFERENCES

[1] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Commun. Surv. Tutor.*, vol. 15, no. 3, pp. 1192–1209, 2012.

[2] F. Attal, S. Mohammed, M. Dedabrishvili, F. Chamroukhi, L. Oukhellou, and Y. Amirat, "Physical human activity recognition using wearable sensors," *Sensors*, vol. 15, no. 12, pp. 31 314–31 338, 2015.

[3] F. Kulsoom, S. Narejo, Z. Mehmood, H. N. Chaudhry, A. Butt, and A. K. Bashir, "A review of machine learning-based human activity recognition for diverse applications," *Neural Comput. Appl.*, vol. 34, no. 21, pp. 18 289–18 324, 2022.

[4] Y. Du, Y. Lim, and Y. Tan, "A novel human activity recognition and prediction in smart home based on interaction," *Sensors*, vol. 19, no. 20, 2019.

[5] L. Schrader, A. Vargas Toro, S. Konietzny, S. Rüping, B. Schäpers, M. Steinböck, C. Krewer, F. Müller, J. Güttler, and T. Bock, "Advanced sensing and human activity recognition in early intervention and rehabilitation of elderly people," *J. Popul. Ageing*, vol. 13, pp. 139–165, 2020.

[6] J.-H. Hsiao, Y.-H. Deng, T.-Y. Pao, H.-R. Chou, and J.-Y. Chang, "Design of a wireless 3D hand motion tracking and gesture recognition glove for virtual reality applications," in *Inf. Stor. Proc. Syst.*, vol. 58103, 2017.

[7] J. Haladjian, D. Schlabbers, S. Taheri, M. Tharr, and B. Bruegge, "Sensor-based detection and classification of soccer goalkeeper training exercises," *ACM Trans. Internet Things*, vol. 1, no. 2, pp. 1–20, 2020.

[8] M. M. Hassan, A. Gumaei, G. Aloi, G. Fortino, and M. Zhou, "A smartphone-enabled fall detection framework for elderly people in connected home healthcare," *IEEE Netw.*, vol. 33, no. 6, pp. 58–63, 2019.

[9] S. K. Hiremath and T. Plötz, "The lifespan of human activity recognition systems for smart homes," *Sensors*, vol. 23, no. 18, 2023.

[10] S. K. Hiremath, Y. Nishimura, S. Chernova, and T. Plötz, "Bootstrapping human activity recognition systems for smart homes from scratch," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 6, no. 3, pp. 1–27, 2022.

[11] J. Han, Z. Zhang, C. Mascolo, E. André, J. Tao, Z. Zhao, and B. W. Schuller, "Deep learning for mobile mental health: Challenges and recent advances," *IEEE Signal Process. Mag.*, vol. 38, no. 6, pp. 96–105, 2021.

[12] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends Cogn. Sci.*, vol. 3, no. 4, pp. 128–135, 1999.

[13] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychol. Learn. Motiv.* Elsevier, 1989, vol. 24, pp. 109–165.

[14] S. Tian, L. Li, W. Li, H. Ran, X. Ning, and P. Tiwari, "A survey on few-shot class-incremental learning," *Neural Netw.*, vol. 169, pp. 307–324, 2024.

[15] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3366–3385, 2021.

[16] Z. Chen and B. Liu, *Lifelong machine learning*. Morgan & Claypool Publishers, 2018.

[17] C. F. S. Leite and Y. Xiao, "Resource-efficient continual learning for sensor-based human activity recognition," *ACM Trans. Embed. Comput. Syst.*, vol. 21, no. 6, pp. 1–25, 2022.

[18] B. Kann, S. Castellanos-Paez, and P. Lalanda, "Evaluation of regularization-based continual learning approaches: Application to HAR," in *2023 IEEE International Conference on Pervasive Computing*

and Communications Workshops and other Affiliated Events (PerCom Workshops). IEEE, 2023, pp. 460–465.

[19] Y. D. Kwon, J. Chauhan, A. Kumar, P. H. HKUST, and C. Mascolo, "Exploring system performance of continual learning for mobile and embedded sensing applications," in *2021 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 2021, pp. 319–332.

[20] Z. Qiao, Q. Pham, Z. Cao, H. H. Le, P. N. Suganthan, X. Jiang, and S. Ramasamy, "Class-incremental learning for time series: Benchmark and evaluation," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 5613–5624.

[21] R. Adaimi and E. Thomaz, "Lifelong adaptive machine learning for sensor-based human activity recognition using prototypical networks," *Sensors*, vol. 22, no. 18, 2022.

[22] M. Schiemer, L. Fang, S. Dobson, and J. Ye, "Online continual learning for human activity recognition," *Pervasive Mob. Comput.*, vol. 93, 2023.

[23] K. Fan, J. Li, S. Lai, L. Lv, A. Liu, J. Tang, H. H. Song, Y. Yue, and H. Zhuang, "TS-ACL: A time series analytic continual learning framework for privacy-preserving and class-incremental pattern recognition," *arXiv preprint arXiv:2410.15954*, 2024.

[24] Y. Wu, M. Nie, T. Zhu, L. Chen, H. Ning, and Y. Wan, "PTMs-TSCIL pre-trained models based class-incremental learning," *arXiv preprint arXiv:2503.07153*, 2025.

[25] X. Zhang, H. Yu, Y. Yang, J. Gu, Y. Li, F. Zhuang, D. Yu, and Z. Ren, "HarMI: Human activity recognition via multi-modality incremental learning," *IEEE J. Biomed. Health Inform.*, vol. 26, no. 3, pp. 939–951, 2021.

[26] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, 2017.

[27] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 114, no. 13, pp. 3521–3526, 2017.

[28] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 139–154.

[29] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Int. Conf. Mach. Learn.* PMLR, 2017, pp. 3987–3995.

[30] Z. Mai, R. Li, J. Jeong, D. Quispe, H. Kim, and S. Sanner, "Online continual learning in image classification: An empirical survey," *Neurocomputing*, vol. 469, pp. 28–51, 2022.

[31] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne, "Experience replay for continual learning," *Adv. Neural Inf. Process. Syst.*, vol. 32, 2019.

[32] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[33] Y. D. Kwon, J. Chauhan, and C. Mascolo, "FastICARL: Fast incremental classifier and representation learning with efficient budget allocation in audio sensing applications," *arXiv preprint arXiv:2106.07268*, 2021.

[34] D. Shim, Z. Mai, J. Jeong, S. Sanner, H. Kim, and J. Jang, "Online class-incremental continual learning with adversarial shapley value," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, 2021, pp. 9630–9638.

[35] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[36] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "iCaRL: Incremental classifier and representation learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[37] A. Desai, C. Freeman, Z. Wang, and I. Beaver, "TimeVAE: A variational auto-encoder for multivariate time series generation," *arXiv preprint arXiv:2111.08095*, 2021.

[38] D. Anguita, A. Ghio, L. Oneto, X. Parra Perez, and J. L. Reyes Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proceedings of the 21th International European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, 2013, pp. 437–442.

[39] M. Zhang and A. A. Sawchuk, "USC-HAD: A daily activity dataset for ubiquitous activity recognition using wearable sensors," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, 2012, pp. 1036–1043.

[40] S. Kanoga, R. Karakida, T. Hoshino, Y. Okawa, and M. Tada, "Deep generative replay-based class-incremental continual learning in sEMG-based pattern recognition," in *2024 46th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2024.

[41] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[42] F. Matteoni, A. Cossu, C. Gallicchio, V. Lomonaco, and D. Bacciu, "Continual learning for human state monitoring," *arXiv preprint arXiv:2207.00010*, 2022.

[43] B. Kann, S. Castellanos-Paez, and P. Lalanda, "Cross-dataset continual learning: assessing pre-trained models to enhance generalization in HAR," in *2024 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. IEEE, 2024, pp. 1–6.