On Handling Catastrophic Forgetting for Incremental Learning of Human Physical Activity on the Edge

Jingwei Zuo
Technology Innovation Institute
Abu Dhabi, UAE
jingwei.zuo@tii.ae

George Arvanitakis
Technology Innovation Institute
Abu Dhabi, UAE
george.arvanitakis@tii.ae

Hakim Hacid
Technology Innovation Institute
Abu Dhabi, UAE
hakim.hacid@tii.ae

ABSTRACT

Human activity recognition (HAR) has been a classic research problem. In particular, with recent machine learning (ML) techniques, the recognition task has been largely investigated by companies and integrated into their products for customers. However, most of them apply a predefined activity set and conduct the learning process on the cloud, hindering specific personalizations from end users (i.e., edge devices). Even though recent progress in Incremental Learning allows learning new-class data on the fly, the learning process is generally conducted on the cloud, requiring constant data exchange between cloud and edge devices, thus leading to data privacy issues. In this paper, we propose PILOTE, which pushes the incremental learning process to the extreme edge, while providing reliable data privacy and practical utility, e.g., low processing latency, personalization, etc. In particular, we consider the practical challenge of extremely limited data during the incremental learning process on edge, where catastrophic forgetting is required to be handled in a practical way. We validate PILOTE with extensive experiments on human activity data collected from mobile sensors. The results show PILOTE can work on edge devices with extremely limited resources while providing reliable performance.

1 INTRODUCTION

Human activity recognition (HAR) has gained, in recent years, a great interest from both the research community and industry players. The activity data can be collected from multiple data sources, such as images [30], videos [15], GPS trajectories [9], smart sensors [4], etc. Among which, the activity data collected from wearable sensors are widely studied and applied in real-life products by almost all tech giants, e.g., Huawei-Sussex locomotion datasets [11], Google platform [23], Samsung health activity trackers 1, and Apple CMotionActivity [7]. However, these centralized and cloud-based solutions usually lack of flexibility to user’s personalization needs. Since data exchanges between cloud and edge devices are usually required, data privacy and processing latency issues may also occur.

Edge ML [19] has brought the processing in ML to the edge of the network and intends to adapt AI technologies to the edge environments. Generally, Edge environment refers to the end-user side pervasive environment composed of devices from both the base station and end device levels. As shown in Figure 1, moving ML from a centralized cloud server to edge devices brings benefits, such as reduced communication cost or latency and preserving data privacy. However, it is challenging for the learning process because of the limited computational resources and data availability in each edge device to train a decent ML model [10].

Learning on the Edge exposes new problems that were not necessarily taken care of in the classical ML domain. The most evident issue is undoubtedly the dynamics of the data. In fact, Edge learning assumes that the observed world is (i) partial, i.e., not all the situations are observed, and (ii) dynamic, i.e., new situations may emerge. The assumptions impose new challenges on ML methods to incrementally learn new situations without forgetting previously acquired knowledge. In other words, ML methods on the Edge have to handle the catastrophic forgetting problem [17]. In the context of human activity recognition, since activity data are generated on the fly but not in a batch mode, class imbalance is also a practical challenge when learning from the dynamic streams.

The problem of catastrophic forgetting has a long history in machine learning and has recently been largely studied in deep learning [5]. Some of the most recent works have addressed this problem from multiple perspectives, which can be divided into three categories: Replay methods (i.e., store data samples) [27], regularization-based methods [25] and parameter isolation methods [28]. However, these proposals are all designed for centralized/cloud-based solutions and not applicable in Edge learning context with limited resources.

In this paper, considering the aforementioned Privacy & Utility challenges as shown in Figure 1, we propose PILOTE: Pushing Incremental Learning on human activities at the exTreme Edge, a method that contributes to reducing the effect of catastrophic forgetting with a focus on a practical case: incremental human physical activity recognition on the Edge. The Siamese Network-based model [18] with supervised Contrastive Loss [16] allows learning from extremely limited data with class imbalance. A jointly optimized Distillation Loss prevents the model from forgetting the previously learned knowledge by guiding the embedding space’s building. The performed experiments on our collected human activity data demonstrate that the proposed approach...
properly handles the addition of new classes while maintaining good performance over the previously learned classes.

We summarize the paper’s main contributions as follows:

- **Incremental Edge Learning:** We introduce a new primitive, PILOTE, for incremental human activity learning on the Edge, which is rarely investigated by the community.
- **Catastrophic Forgetting on Edge Learning:** We study and tackle the catastrophic forgetting problem in PILOTE when learning on the Edge.
- **Edge resource constraints:** We consider the limited computation resource and storage capacity in Edge devices. PILOTE is designed to be with low computation and storage costs.
- **Extensive experiments on real-life data:** We apply PILOTE on real-life human activity data, which is collected by volunteers in our data collection campaigns.

The rest of this paper is organized as follows: Section 2 presents the most related work to our proposal. Section 3 shows the MAGNETO platform, where we deploy our Edge learning proposal for real-life human activity learning. Section 4 introduces the problem definition. Section 5 formalizes the proposed method. An extensive set of experiments is performed and discussed in Section 6. We conclude and show future work in Section 7.

## 2 RELATED WORK

In this section, we show the most related work of our proposal in the Incremental Learning and Edge Machine Learning, with an application on Human Activity Recognition (HAR) task.

### 2.1 Incremental Learning

Incremental learning has been investigated for decades in the machine learning community. It refers to integrating new knowledge on the fly to the learned model, with unseen data coming in the future. In different contexts, the incremental learning is illustrated differently with various terminologies. It is also referred to as continual learning [5], lifelong learning [21] or sequential learning [2]. According to the learning targets, or the input and output distributions $P(X(t))$ and $P(Y(t))$ of an incoming batch data at time $t$, the incremental learning can be categorized into three types [5]:

- **Class incremental learning:** $\{Y(t) \subset Y(t+1)\}$ with $P(Y(t)) \neq P(Y(t+1))$
- **Task incremental learning:** $\{Y(t) \neq Y(t+1)\}$
- **Domain incremental learning:** $\{Y(t) = Y(t+1)\}$ with $P(Y(t)) = P(Y(t+1))$

Considering the newly emerged activities (i.e., classes) collected by Edge devices, we focus our attention in this paper on the class incremental learning task. A handful of works [17, 20, 24, 26, 29] are devoted to handling the catastrophic forgetting problem in the class incremental learning context. For instance, as a regularization-based method, Learning without Forgetting (LwF) [20] learns new classes, meanwhile conserves the learned information via knowledge distillation from the old classes; iCaRL [26] keeps a set of class prototypes (i.e., memory replay) for representing the old classes, which are combined with new-class samples to update the model. Similarly, GDumb [24] optimized the class prototype selection via Greedy Search; Instead of building up a memory replay for old-class data, Generative Replay [29] learns a generative model and a solver (i.e., classifier) for learned data, that allows modeling the past class distribution instead of conserving raw data instances.

However, to push the incremental learning paradigm to the Edge, the above-mentioned methods are hardly applicable due to their context-specific model structure for Cloud-based processing. For instance, though LwF [20] does not require caching any past data for building a new model, the model itself is huge enough to be transferred or executed in Edge devices.

### 2.2 Edge Machine Learning (Edge ML)

Edge Machine Learning (Edge ML) emerges as a new paradigm, owning to its unbeatable advantages on low latency and strong privacy guarantee. In general, Edge ML can be characterized into two categories: i) Inference on the Edge, ii) Training on the Edge.

For model inference on the Edge, past studies tend to optimize the model’s scale and quantize its weights to reduce resource costs. For instance, commonly existing in Deep Neural Networks (DNNs), the model’s parameter redundancies can be investigated to compress/re-design the model, while ensuring a low loss in performance. Typical model optimization methods include parameter pruning [12], low-rank factorization [6], knowledge distillation [13], etc. These methods can be applied to different DNNs or be composed to optimize a complex model’s inference on the Edge. Another branch of work [32] proposes to divide the DNN models and perform distributed inference computation.

For model training on the Edge, the resource cost is much higher than that in inference. Basically, the Edge training can be either based on tiny models requiring minimal resources, or splitting the training task via distributed/federated learning [31]. In this paper, we investigate the lightweight models which can be applied on Edge devices with limited resources.

### 2.3 Human Activity Recognition (HAR)

As a classic research problem in Data Mining and Machine Learning, Human Activity Recognition (HAR) gathers enormous studies regarding the data format, pre-processing techniques, learning models, post-processing methods, and model deployments. Among various data sources of HAR tasks, we consider human physical activities collected from wearable sensors. These activity data are generally represented by Time Series, more precisely, Multivariate Time Series (MTS) [34] on multiple sensors.

As a classification task, the HAR model can be designed differently regarding data resources and targeted applications. For instance, one can use handcrafted features to feed any general ML models for downstream tasks, which is easy-to-deploy and requires linear processing time. More advanced work has been proposed in the Time Series Classification domain, where researchers aim to build general ML models covering various application domains [34], including HAR tasks. For instance, authors in [33] extract interpretable Shapelet features from time series, and combine with a kNN classifier; The recent end-to-end models [34] have shown promising results on HAR tasks, that generally rely on automatic feature extraction and selection [1, 8] with regard to specific learning tasks, e.g., classification or forecasting.

However, these approaches aim to reach optimal model performance with less attention on the model’s scale or complexity, which are essential in the Edge context regarding the extremely limited resources. In this paper, we consider the HAR task as an application and focus on the Edge model’s incremental learning behavior. Therefore, we adopt a primary feature extractor that
relies on handcrafted statistic features, requiring linear processing time. Nevertheless, more advanced feature extractors can be explored and integrated into our framework, by considering the Edge constraints. This is orthogonal to our work.

3 MAGNETO PLATFORM

MAGNETO, sMArt sensinG for humaN activity rEcogniTIOn, is a platform that provides inference on an Edge device without transferring user’s data to the Cloud. Figure 2 shows its main architecture. The Cloud-based approach for Human Activity Recognition (HAR) tasks generally relies on continuous data exchange between Cloud and Edge, leading to high processing latency and privacy issues. MAGNETO is built primarily with an Edge-based structure, in which an initial HAR model is pre-trained on the Cloud as a warm starting point [3] for the Edge learning process. The pre-trained model benefits from the rich computation resources on the Cloud computing center, meanwhile allowing further adaptation for new-comming real-time data on the Edge.

With limited computation resources but high flexibility, the Edge devices in MAGNETO platform provide the possibility for real-time data collection, model adaptation/re-training/calibration, and model inference. Importantly, all the operations are taking place on the Edge, and should not have any data exchange with the Cloud, considering data privacy issues. By adopting the MAGNETO platform, we are able to deploy PILOTE in a real industrial pipeline for Edge learning on real-time human activities.

Figure 2: MAGNETO processing steps: (left side: Cloud-based architecture) (i) human activity data is captured on the Edge device, (ii) edge data is transferred to the Cloud, (iii) model training or inference on the Cloud, and (iv) send predictions back to the Edge device. (right side: Edge-based architecture) (i) an initial model is pretrained on the Cloud, (ii) data is collected on the Edge device, (iii) the Edge device performs model’s updating (with new data samples) or inference, (iv) the Edge device displays the predictions.

4 PROBLEM FORMULATION

We consider a warm-start problem [3] for learning new human activities on the edge: a learning model is pre-trained on the Cloud with its scalable resources, serving as a warm starting point for incremental learning on the edge. Table 1 summarizes the notations used in the paper.

Definition 1. (Incremental learning). Given a model \( \Theta_o \) trained on \( D_o \), new sample set \( D_n = \{X_n, \ldots, X_t\} \) with unknown activities \((s, \ldots, t)\) will enrich \( \Theta_o \) incrementally, leading to a new model \( \Theta_n \).

\[ D_o = \{X^*_1, \ldots, X^*_t\} \]
\[ D_n = \{X^*_1, \ldots, X^*_t\} \]
\[ \mathcal{P}^o = \{P^1, \ldots, P^o\} \]
\[ \varphi : \mathcal{X} \rightarrow \mathcal{R}^d \]
\[ \Theta \]

Table 1: Notation

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( D_o )</td>
<td>Dataset of old classes (1, \ldots, s-1)</td>
</tr>
<tr>
<td>( D_n )</td>
<td>Dataset of new classes (s, \ldots, t)</td>
</tr>
<tr>
<td>( \mathcal{P}^o )</td>
<td>Class exemplar sets</td>
</tr>
<tr>
<td>( \varphi )</td>
<td>Feature map function</td>
</tr>
<tr>
<td>( \Theta )</td>
<td>Model parameters</td>
</tr>
</tbody>
</table>

Definition 2. (Catastrophic forgetting). Given a newly updated model \( \Theta_n \) trained sequentially on \( D_o \) and \( D_n \), the model tends to learn more from \( D_n \) but forget what has been already learned on \( D_o \), i.e., \( \sum_{(x_i, y_i) \in D_o} L(\varphi_{\Theta_n}(x_i), y_i) > \sum_{(x_i, y_i) \in D_n} L(\varphi_{\Theta_n}(x_i), y_i) \).

Incremental learning of human physical activities on the edge requires considering not only the effect of catastrophic forgetting when updating the learning model, but also the extremely limited activity data collected by the edge devices, leading to imbalanced data distributions among classes. Therefore, given a model \( \Theta_o \) trained on \( D_o \), a new sample set \( D_n \) comes with a small size \( |D_n| \ll |D_o| \), the incremental learning at the extreme edge aims to learn a model \( \Theta_n \), that guarantees the performance on \( D_n \): \( \min\sum_{(x_i, y_i) \in D_n} L(\varphi_{\Theta_n}(x_i), y_i) \) and maintaining the performance on \( D_o \): \( \sum_{(x_i, y_i) \in D_o} L(\varphi_{\Theta_n}(x_i), y_i) = \sum_{(x_i, y_i) \in D_o} L(\varphi_{\Theta_o}(x_i), y_i) \).

5 PROPOSAL: PILOTE

In order to answer the aforementioned challenges for incremental human activity learning on the edge, we propose PILOTE, a method that updates the learning model with new knowledge extracted from incoming new-class instances, while preserving the learned knowledge from existing classes.

Figure 3 shows the global system structure of PILOTE. The activity data collected from smartphone sensors are represented as Multivariate Time Series (MTS) [34]. The preprocessing steps (e.g., denoising, segmentation, normalization, etc.), with linear time operations, are conducted equally on the Cloud and Edge devices. In particular, following the architecture design of the MAGNETO platform, PILOTE learns a feature representation from a data stream on the Edge, i.e., \( X^1, \ldots, X^t \) in a class-incremental manner, where all examples of a set \( X^y = \{x^y_1, \ldots, x^y_{n_y}\} \) are of class \( y \in N \). An incremental contrastive loss is designed to integrate new knowledge from \( D_n \), which forms the representation space gradually with a minimal computation cost. Meanwhile, inspired by knowledge distillation [25] for handling catastrophic forgetting problems, we propose a distillation loss which is designed explicitly for conserving the learned knowledge from existing datasets via guiding the representation space’s building. Joint optimization of the distillation loss and contrastive loss allows for building the representation space in an extremely efficient manner.

On the representation space, we apply the nearest class mean (NCM) classifier for a given sample \( x_i \) which can be denoted as:
\[
y_i^* = \arg \min_{y \in \{1, \ldots, t\}} \text{dist} (\varphi_{\Theta_n}(x_i), \mu_y) ,
\]
\[
\mu_y = \frac{1}{n_y} \sum_{p_i^y} \varphi_{\Theta_n}(p_i^y),
\]
where \( \mu_y \) denotes the prototype of class \( y \), \( P^o \) is the exemplar set of class \( y \) and \( n_y \) is the number of exemplars in class \( y \).
5.1 Incremental representation learning
Compared to large volumes of data adopted for training the learning model on the cloud, new human activities recorded at edge devices usually show limited data samples. The imbalanced data distribution will lead to training constraints for learning the latest activities. To this end, instead of building an end-to-end classification model, we adopt the concept of Few-shot Learning: a Siamese Network [18] is applied to learn a robust representation space, for which much less data is required. Instead of learning feature-label mappings, the Siamese Network only learns separable boundaries in the representation space between the activities. In this space, the intra-class embeddings are clustered, whereas the inter-class embeddings are clearly bounded. A set of class prototypes are generated to reduce the computation cost of the NCM classifier. The supervised contrastive loss [16] is applied to the representation space:

\[
L(x_i, x_j, Y) = Y \cdot \|\phi_{\Theta}(x_i) - \phi_{\Theta}(x_j)\|^2 \\
+ (1 - Y) \cdot \max \{0, m^2 - \|\phi_{\Theta}(x_i) - \phi_{\Theta}(x_j)\|^2\} \quad (2)
\]

where \(m > 0\) is the margin parameter, \(Y = 1\) if \(x_i, x_j\) are similar (i.e., \(y_i = y_j\)), otherwise \(Y = 0\).

As shown in Algo. 1, we first select the most representative exemplars from old-class samples as a support set, referring to the \(m\) closest samples to each class prototype \(\mu_k\). The support set will be transferred along with a pre-trained model \(\Theta_0\) from the Cloud to the Edge devices. Then, the representation space is built over the support set and new-class samples. The exemplar sets and class prototypes are updated continuously with newly integrated samples. Following this process, the approach incrementally learns new classes. However, it suffers from the limitation of forgetting past knowledge. Therefore, to complement this method, we investigate the problem and propose a solution, discussed in the next section.

5.2 Catastrophic forgetting at the extreme edge
When learning from new activities, the model tends to forget what has been learned from previous activities. Precisely, the boundary between existing activities in the embedding space becomes unclear after learning new activities. Therefore, it is required to maintain the learned boundary between the old-class and new-class samples, while learning new activity embeddings by staying far away from existing ones.

As a concept from Transfer Learning, Distillation Loss [13] has been widely adopted for extracting key knowledge from a trained neural network. We apply the Distillation Loss under the

---

**Algorithm 1:** Incremental Representation Learning considering Catastrophic Forgetting on the Edge

<table>
<thead>
<tr>
<th>Step</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(m = K/(s - 1))</td>
</tr>
<tr>
<td>2</td>
<td>for (y \in {1, \ldots, s - 1}) do</td>
</tr>
<tr>
<td>3</td>
<td>(X^y = (x_{1}^{y}, \ldots, x_{m}^{y}))</td>
</tr>
<tr>
<td>4</td>
<td>(\mu^y = \frac{1}{m} \sum_{i=1}^{m} \phi_{\Theta_0}(x_i))</td>
</tr>
<tr>
<td>5</td>
<td>/* Select (m) closest samples in the embedding space */</td>
</tr>
<tr>
<td>6</td>
<td>for (k \in {1, \ldots, m}) do</td>
</tr>
<tr>
<td>7</td>
<td>(p_k^y \leftarrow \arg\min_{x \in X^y} |\mu^y - \frac{1}{k} \left(\sum_{j=1}^{k-1} \phi_{\Theta_0}(p_j^y)\right)|)</td>
</tr>
<tr>
<td>8</td>
<td>(p^y \leftarrow (p_1^y, \ldots, p_m^y))</td>
</tr>
<tr>
<td>9</td>
<td>/* (Edge) Incremental Representation Learning with new-class samples */</td>
</tr>
<tr>
<td>10</td>
<td>(D_0 \leftarrow \bigcup_{y=1, \ldots, s-1} { (x, y) : x \in X^y } )</td>
</tr>
<tr>
<td>11</td>
<td>(D_n \leftarrow \bigcup_{y=s, \ldots, s} { (x, y) : x \in X^y } )</td>
</tr>
<tr>
<td>12</td>
<td>Run network training with an updated parameter set (\Theta_n) with loss function (L = \alpha \times L_{dist} + (1 - \alpha) \times L_{contra}), where</td>
</tr>
<tr>
<td>13</td>
<td>(L_{dist} = \sum_{(x_i, y_i) \in D_n} |\phi_{\Theta_n}(x_i) - \phi_{\Theta_n}(x_j)|^2)</td>
</tr>
<tr>
<td>14</td>
<td>(L_{contra} = \sum_{(x_i, y_i) \in D_n} \sum_{(x_j, y_j) \in D_n} L(x_i, x_j, y_i = y_j))</td>
</tr>
<tr>
<td>15</td>
<td>return (\Theta_n)</td>
</tr>
</tbody>
</table>
few-shot learning context, where the previously learned model can be constrained while integrating newly learned knowledge from extremely limited instances. Importantly, the introduced constraints allow simplifying the computations of the Contrastive Loss, thus reducing the time complexity of the incremental learning process. For instance, given \( n_t \) new samples of class \( t \), the number of pairs in Contrastive Loss can be reduced to \( \frac{n_t}{2(n_t - 2)!} \), instead of \( \sum_{y \in \{1, \ldots, t-1\}} f * \frac{n_t}{2(n_t - 2)!} \), where \( n_t \) is the number of samples in class \( y \). In the embedding space, the learned representations from old-class instances are already constrained by Distillation Loss.

Finally, by combining both the Distillation Loss from old-class instances and the Contrastive Loss of new-class instances with a balancing weight \( \alpha \), we are capable of learning a robust incremental learning model with constant memory.

6 EXPERIMENTS

In this section, we demonstrate the effectiveness of PILOTE with real-life human activity datasets. The experiments were designed to answer the following questions:

Q1 Catastrophic forgetting: How successful is our model at learning new activities while considering the catastrophic forgetting problem?

Q2 Applicability on the edge: How well does our model perform on the edge with limited storage and computing resources?

Q3 Extreme edge with minimal new-class samples: How does our model perform when new-class samples are extremely limited?

6.1 Experimental settings

6.1.1 Dataset description. We base our experiments on real human physical activity data collected on edge devices. We have launched data collection campaigns, capturing an initial dataset of more than 100GB of sensory data. Even though the preprocessing was conducted on the cloud, the real-time coming data can be processed instantly, as the preprocessing operation requires linear time. Precisely, we split the sensory data into a one-second recording window with roughly 120 sequential measurements from 22 mobile sensors, e.g., accelerometer, gyroscope, and magnetometer. We extract 80 statistical features such as the average, the variance for each feature, the average jerk, and the variance of the jerk for each three-dimensional feature sensor. Currently, five activities with ~ 200k records are collected: ‘Drive’, ‘E-scooter’, ‘Run’, ‘Still’, ‘Walk’. To simplify the test scenarios and validate the learning model, we design the incremental learning process by setting one of the activities as the new-class data to be learned.

6.1.2 Execution and Parameter Settings. The proposed model is implemented in PyTorch 1.6.0 and is trained using the Adam optimizer. We set an adaptive learning rate regarding training epochs, i.e., the learning rate starts from 0.01 and decreases by half every training epoch. The backbone model is a simple Fully Connected (FC) neural network with dimensions \([1024 \times 512 \times 128 \times 64 \times 128]\). We apply Batch Normalization [14] and Rectified Linear Units (ReLU) [22] activation functions on the first four layers. The last layer projects the input samples into a representation space with an embedding size of 128. The backbone model can be any other advanced network structure. Here we focus on the model’s incremental learning behaviors. We set the balancing weight \( \alpha = 0.5 \) in all experiments. We split 30% of the collected records as the test set. The validation split is set to 0.2 for both pre-training and incremental training. We set the stopping condition to hold when the difference of validation loss between epochs is less than a small threshold, 0.0001 for five consecutive steps. We execute each model in five rounds and report the average accuracy and the standard deviations.

6.1.3 Baselines. To the best of our knowledge, previous work rarely considers the extremely limited learning data and catastrophic forgetting problem simultaneously in the Edge learning context. Therefore, we compare PILOTE with two popular strategies for human activity recognition on the edge:

1 Pre-trained model: The model is pre-trained on the cloud on four activities. It is transferred to the edge with a support set. The model generates class prototypes for new-class samples and enriches the support set with random new-class data.

2 Re-trained model: The pre-trained model is re-trained on the edge using the enriched support set with new-class samples.

6.2 Q1: Catastrophic forgetting

To validate the model’s robustness in handling the catastrophic forgetting problem, we pick one of the five activities each time as the new class to be learned. Table 2 shows the models’ performance comparisons on the five new-class scenarios. The re-trained model and PILOTE in each scenario are based on the same pre-trained model. We report the average accuracy and the standard deviations of five executions for both the re-trained model and PILOTE.

<table>
<thead>
<tr>
<th>New class</th>
<th>Pre-trained</th>
<th>Re-trained</th>
<th>PILOTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drive</td>
<td>0.8443</td>
<td>0.8825±0.0391</td>
<td>0.8837±0.0278</td>
</tr>
<tr>
<td>E-scooter</td>
<td>0.8744</td>
<td>0.9491±0.0089</td>
<td>0.9516±0.0074</td>
</tr>
<tr>
<td>Run</td>
<td>0.7856</td>
<td>0.9126±0.0321</td>
<td>0.9372±0.0319</td>
</tr>
<tr>
<td>Still</td>
<td>0.8213</td>
<td>0.9143±0.0288</td>
<td>0.9349±0.0340</td>
</tr>
<tr>
<td>Walk</td>
<td>0.8376</td>
<td>0.8909±0.0383</td>
<td>0.9193±0.0386</td>
</tr>
</tbody>
</table>

Table 2 suggests that PILOTE does achieve better performances by conserving the learned knowledge from old-class samples. In particular, for activities ‘Run’, ‘Still’, and ‘Walk’, PILOTE obtained more than 2% performance improvement over the re-trained model. These activities are more challenging to learn and distinguish than ‘Drive’ and ‘E-scooter’. This can also be illustrated by Figure 4, which shows confusion matrices when learning the new activity ‘Run’. The confusion matrices suggest that the re-trained model tends to learn from ‘Run’ but forgets what has been learned from ‘Walk’, leading to a massive amount of false positives when predicting ‘Run’, and false negatives for ‘Walk’. Whereas PILOTE allows separating the two similar activities by conserving the knowledge learned previously from ‘Walk’, Figure 5 shows the visualization of the embedding spaces of the three models, which further validates our claims: a re-trained model can better separate ‘Run’ and ‘Walk’ than a pre-trained model, but shows a more blurred boundary than PILOTE.

6.3 Q2: Applicability on the edge

With limited computing and storage resources on the edge, the learning process is required to be efficient, and effective when learning from a small amount of data. In Figure 6, we validate the models’ performance regarding the support set’s size or the...
catastrophic forgetting and learn from a small sample set. With less than 200 exemplars per class (i.e., \( < 256KB \)), PILOTE can reach an accuracy of 93.72% within 20 training epochs, and each epoch costs less than 0.5s. These features make PILOTE highly suitable to be deployed on edge devices.

6.4 Q3: Extreme edge with a few samples

In practice, new activities are recorded on edge devices, the data volume is not as large as that collected via a centralized manner. Very limited data samples can be recorded at the extreme edge for learning new activities, leading to an unbalanced data distribution among classes. To validate the model’s performance at the extreme edge, we vary the amount of new-class samples (i.e., ‘Run’) in the support set, and select 200 representative exemplars for old-class samples. We need to note that the new-class exemplars are randomly selected from new-class samples.

Figure 7 shows the performance comparison between PILOTE and the re-trained model regarding the number of new-class exemplars in the support set. We also show the pre-trained model’s accuracy as a warm starting point [3]. The results suggest that only with 30 exemplars in class ‘Run’, PILOTE can obtain a 90% accuracy score. Globally, PILOTE performs better than the re-trained model, particularly when the exemplar number is small (e.g., \(< 50\)). When the previously learned knowledge is conserved, the model requires less effort/data for updating the embedding space, thus becoming more robust when new-class samples are extremely limited.

7 CONCLUSION AND PERSPECTIVES

In this paper, we proposed PILOTE, an incremental learning model applicable to the extreme edge, which considers both the limited edge resources and the catastrophic forgetting problem. In future work, we aim to launch more campaigns for collecting more complex data with more human physical activities. The enriched data would help validate PILOTE in more practical and challenging scenarios. From a longer-term perspective, one can consider the model’s scaling up or collaborative learning with strong privacy-preserving guarantees, e.g., Federated Learning.

REFERENCES

The 'Run' activity is excluded from the pre-training process. The support set includes 200 representative exemplars for each class.

**Figure 5:** Visualization of the embedding spaces of the learning models. The 'Run' activity is excluded from the pre-training process.

**Figure 6:** Model accuracy regarding the support set's size. The 'Run' activity is excluded from the pre-training process.

**Figure 7:** Model accuracy regarding the exemplar number in 'Run'. Each class includes 200 exemplars in the support set.