SelfReplay: Adapting Self-Supervised Sensory Models via Adaptive Meta-Task Replay

Hyungjun Yoon
KAIST
hyungjun.yoon@kaist.ac.kr

Jaehyun Kwak
KAIST
jaehyun98@kaist.ac.kr

Biniyam Aschalew Tolera
KAIST
binasc@kaist.ac.kr

Gaole Dai
Nanyang Technological University
gaole001@e.ntu.edu.sg

Mo Li
HKUST
lim@cse.ust.hk

Taesik Gong
UNIST
taesik.gong@unist.ac.kr

Kimin Lee
KAIST
kiminlee@kaist.ac.kr

Sung-Ju Lee
KAIST
profsj@kaist.ac.kr

Abstract

Self-supervised learning enables effective model pre-training on large-scale unlabeled data, which is crucial for user-specific fine-tuning in mobile sensing applications. However, pre-trained models often face significant domain shifts during fine-tuning due to user diversity, leading to performance degradation. To address this, we propose *SelfReplay*, an adaptive approach designed to align self-supervised models to different domains. *SelfReplay* consists of two stages: MetaSSL, which leverages meta-learning with self-supervised learning to pre-train domain-adaptive weights, and ReplaySSL, which further adapts the pre-trained model to each user's domain by replaying the meta-learned self-supervised task with a few user-specific samples. This produces a personalized model tailored to each user. Evaluations on mobile sensing benchmarks demonstrate that *SelfReplay* outperforms existing baselines, improving the F1-score by 9.4%p on average. On-device analyses on a commodity smartphone show the efficiency of *SelfReplay*'s adaptation step, required just once after deployment, with SimCLR completing in only 10 seconds while using less than 100MB of memory.

CCS Concepts

• **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**; • **Computing methodologies** → **Machine learning**.

Keywords

Self-Supervised learning, Mobile Sensing, Domain Adaptation

ACM Reference Format:
Hyungjun Yoon, Jaehyun Kwak, Biniyam Aschalew Tolera, Gaole Dai, Mo Li, Taesik Gong, Kimin Lee, and Sung-Ju Lee. 2025. *SelfReplay*: Adapting Self-Supervised Sensory Models via Adaptive Meta-Task Replay. In *The 23rd ACM Conference on Embedded Networked Sensor Systems (SenSys '25), May 6–9, 2025, Irvine, CA, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3715014.3722066

SenSys '25, May 6–9, 2025, Irvine, CA, USA

ACM ISBN 979-8-4007-1479-5/25/05
https://doi.org/10.1145/3715014.3722066

1 Introduction

Large-scale pre-training has become fundamental in developing models that generalize across diverse applications. Self-supervised learning [24], in particular, enables models to leverage vast amounts of unlabeled data, making it a powerful approach for building foundation models. In mobile sensing applications such as contactless authentication [34, 64], sign language translation [20, 43], and mobile health monitoring [55, 70], labeled data is often scarce and costly to acquire, which makes self-supervised methods especially valuable. Self-supervised techniques such as predictive coding [16, 18], contrastive learning [57], and multi-task learning [52] have demonstrated effectiveness in sensory applications by reducing reliance on labeled data in pre-training.

However, challenges arise once pre-trained models are deployed for fine-tuning across different environments. In mobile sensing, data collected in diverse environments varies widely due to differences in users and device settings (*e.g.*, sensor placement and sampling rate) [58]. These variations introduce a *domain shift*, where models trained in one domain underperform when applied to others [56]. To highlight this challenge in the self-supervised setting, we conducted an empirical analysis using a self-supervised pre-training method [18] for human activity recognition [14] (details in Section 4.6.9). Figure 1 compares the performance of a pre-trained model when fine-tuned within the same source domain versus on a different target domain. The results show that performance declines substantially when fine-tuning is conducted on a different domain. This underscores the challenge of deploying self-supervised models across diverse mobile sensing environments.

Training a domain-specific model using target domain data is straightforward but infeasible due to the cost of gathering sufficient data from each user. Existing research includes domain generalization [36, 45, 47] that trains models with domain-invariant features, and domain adaptation [4, 14, 60, 73] that leverages a small amount of target domain data to achieve domain-specific performance. However, they primarily focus on supervised learning scenarios, making applying to domain shifts arising from self-supervised pre-training difficult.

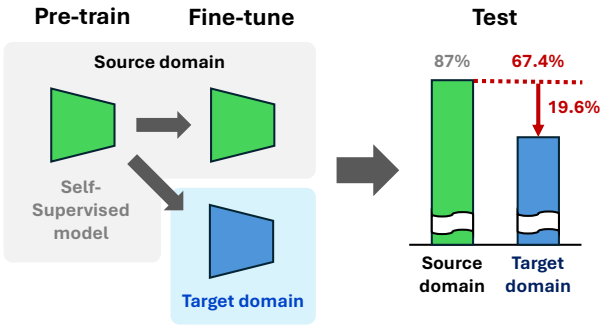

Figure 1: Illustration of domain shift on a self-supervised model pre-trained in one domain and fine-tuned in another for human activity recognition. Fine-tuning on the target domain results in a 19.6% F1-score drop (87% vs. 67.4%).

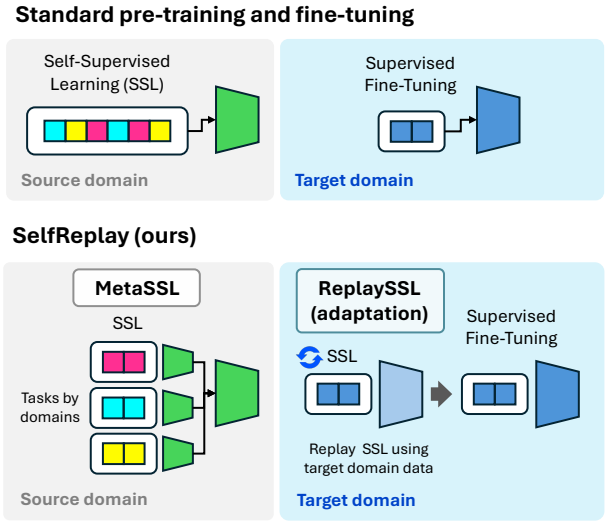

Figure 2: A comparison between the standard pre-training and fine-tuning (top) and *SelfReplay* (bottom). Components in the grey box are performed in the source domain, while those in the blue box are performed in the target domain.

To address the problem, we propose *SelfReplay*, an *adaptive meta-task replay* approach for adapting self-supervised models to different domains. Figure 2 illustrates *SelfReplay* alongside a standard pre-training and fine-tuning setting. We design *SelfReplay* with two key components: (i) MetaSSL generates domain-adaptive pre-trained models and (ii) ReplaySSL adapts the pre-trained model to the target domain. MetaSSL leverages meta-learning over self-supervised objectives, structuring pre-training into multiple few-shot tasks by domain. This enables models to "learn to self-supervise" on only a few domain-specific data, following meta-learning's concept of "learning to learn." After pre-training, ReplaySSL adapts the model to the target domain by replaying the meta-learned self-supervised task with a few samples, creating a *personalized* model. The adapted model is then fine-tuned for the final application task.

We evaluate *SelfReplay* on mobile sensing datasets [1, 14, 50, 56] by simulating domain shifts across different users and devices. Our experiments show that when self-supervised models are fine-tuned on heterogeneous domains, *SelfReplay* consistently outperforms domain generalization and adaptation baselines [30, 71], achieving an average F1-score improvement of 9.4%p. Importantly, *SelfReplay* is agnostic to self-supervised learning objectives, making it compatible with various approaches. We apply *SelfReplay* to contrastive learning [57], predictive coding [18], and multi-task learning [52], demonstrating its effectiveness across different self-supervised methods. To assess practical feasibility, we also measure *SelfReplay*'s computational overhead on three edge devices. On a standard smartphone, ReplaySSL with SimCLR completes in just 10 seconds while using less than 100MB of memory. As the adaptation step runs only once after obtaining the pre-trained model, it introduces minimal computational load for the user.

We summarize our main contributions as follows:

- We investigate the domain shift issue when diverse users deploy and fine-tune self-supervised models. We reveal that the domain shift leads to performance degradation.
- We propose *SelfReplay*, a method for adapting self-supervised models to different domains via adaptive meta-task replay.
- We perform evaluations using mobile sensing datasets, showing that *SelfReplay* outperforms domain generalization and adaptation baselines, achieving an average F1-score improvement of 9.4%p.
- We assess the computational overhead of deploying *SelfReplay* on mobile devices, demonstrating that the adaptation step completes in 10 seconds with less than 100MB of memory usage for SimCLR on an off-the-shelf smartphone.

2 Related Work

2.1 Self-Supervised Learning

Self-supervised learning trains models using an auxiliary task that can be defined without labels, which enables learning generalized features of the data. Among numerous approaches, we focus on the methods applied for mobile sensing [17]. multi-task Learning [52] utilized multiple types of synthetic augmentations on the data and trained task prediction networks to infer the occurrence of the augmentation. Sensor-specific augmentations were selected to make the model learn sensory properties. Recent work focuses on using contrastive learning [23], which generates augmented views of data and trains the model intending to maximize the similarity between the augmented views. Existing methods such as MoCo [19] and SimCLR [6] were applied to mobile sensing [57, 62] by using sensory augmentations to generate views. The temporal property of time-series data is utilized for generating views in a recent work [9]. Taking into account the multi-modality, Cosmo [42], COCOA [8], and ColloSSL [22] utilized contrastive learning to maximize the similarity between the embeddings driven from different modalities in the same context. Contrastive predictive coding (CPC) [16, 18] defined another type of task, predicting the embedding of future segments within the data based on the previously aggregated embeddings. In a similar context, masked-reconstruction-based methods [15, 66] have been explored for mobile sensing, using the task of reconstructing the synthetically masked segment within the data.

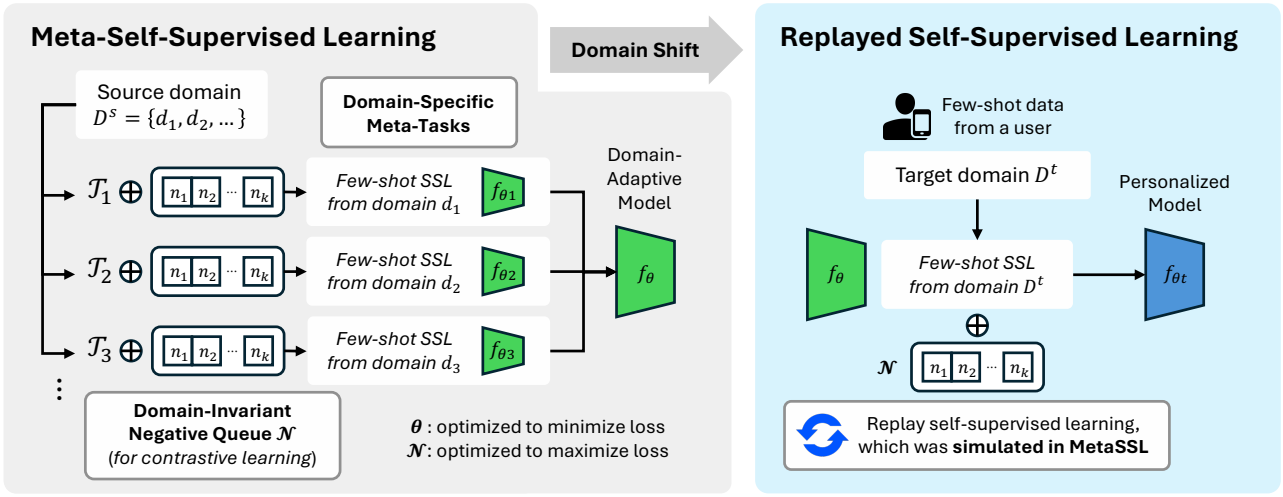

Figure 3: Overview of *SelfReplay*. The model is first pre-trained through MetaSSL in the source domain to develop domain adaptability, followed by ReplaySSL to adapt the model to the target domain.

While self-supervised models are known to be generalizable across diverse tasks, the potential performance decline when applied to different domains (shown in Section 4.6.9) is overlooked. Our work differs from the prior research in exploring the domain shift problem between self-supervised pre-training and fine-tuning.

2.2 Domain Generalization and Domain Adaptation

Domain generalization (DG) [61] mitigates domain shifts by learning domain-invariant features via adjusted objectives [32, 38], adversarial discrimination [33], and domain-information minimization [63, 72]. Recent work incorporated meta-learning [31, 46] and self-supervised learning [29, 67] to define domain-invariant training objectives. In mobile sensing, GILE [45] disentangled domain-specific information, while SDMix [36] employed semantic-aware augmentations to achieve DG tailored to activity recognition. For placement shifts, position- and orientation-agnostic methods [59, 69] rely on data preprocessing and feature transformations but remain limited to a single domain factor and are less adaptable to user differences or other domain variations. Domain adaptation (DA) [65] is more suitable for our scenario, as it allows fine-tuning with user-collected data. Common approaches rely on unlabeled or limited-labeled target data [11, 49] and employ strategies such as feature matching, confusion maximization [4], and transfer learning for activity recognition [26, 35, 60]. MetaSense [14] introduced a meta-learning approach followed by few-shot adaptation to create domain-specific models. DAPPER [13] is another line of research for estimating the expected performance of DA in mobile sensing.

However, these approaches assume the availability of source-domain labels and thus are incompatible with our unsupervised pre-training. DARLING [71] addresses domain shift without source labels by integrating conditional optimization that modifies the contrastive loss per domain. Likewise, ContrastSense [7] targets unsupervised pre-training by introducing a contrastive loss that minimizes inter-domain discrepancies. While these methods effectively generalize across domains, our approach differs by training *domain-adaptive* pre-trained models (MetaSSL), and then leveraging available target-domain data (*i.e.*, fine-tuning data) to train domain-specific models, thereby achieving superior performance through an additional adaptation step (ReplaySSL).

2.3 Unsupervised Meta-Learning

We consider unsupervised meta-learning (UML) [27, 28, 30] methods due to their effectiveness in few-shot adaptation, which is also applicable to our unsupervised pre-training scenario. Traditional methods employ pseudo-labeling data through augmentation [27] or generative methods [28], followed by supervised meta-learning [2] using the generated labels. Set-SimCLR [30], during pre-training, trains a set encoder by creating sets of augmented samples from the same data, employing contrastive learning to maximize agreement between set embeddings. In fine-tuning, it composes sets of data by classes, generating class prototypes using the set encoder to initialize the classifier's parameters. These prototypes enable rapid adaptation for further few-shot fine-tuning. However, our approach differs in that we perform the adaptation to refine the encoder for the target domain, while Set-SimCLR primarily focuses on making the following classifier adaptable to few-shot fine-tuning. Our evaluation (Section 4.6.1) demonstrates the superior performance of our approach in mobile sensing scenarios.

3 Method

We present *SelfReplay*, an approach for adapting self-supervised models to diverse domains via adaptive meta-task replay. In Section 3.1, we formulate the domain shift problem that arises when self-supervised models are deployed to different target domains. Our solution, illustrated in Figure 3, incorporates two key strategies: (i) Meta-Self-Supervised Learning (MetaSSL) to pre-train domain-adaptive weights, and (ii) Replayed Self-Supervised Learning (ReplaySSL) to adapt pre-trained models to the target domain with only few samples. We detail the design of MetaSSL in Section 3.2 and ReplaySSL in Section 3.3.

3.1 Problem Formulation

Domain Shift from Pre-Training. Pre-training generates representation models using large-scale data, but obtaining corresponding labels is often expensive or restricted. For example, in activity recognition, asking users to report their activity every hour is costly. In health applications, labels often contain sensitive personal information (*e.g.*, medical diagnoses and mental health assessments), sometimes making label acquisition infeasible. To address this, models are pre-trained on unlabeled data before being adapted to specific tasks. Existing large-scale efforts have produced models trained solely on unlabeled datasets, such as Google's 40 million hours of physiological data [39] and UK-Biobank's 700,000 person-days of accelerometer data [68].

We target scenarios where users utilize the pre-trained models for their own applications. Pre-trained models provide rich feature representations, enabling users to train models with only a small amount of data. For example, when setting up a fitness app, a smartphone might prompt the user to stay, walk, or run for 30 seconds, allowing the model to fine-tune running detection with just a few labeled samples. Similarly, for gesture recognition, leveraging pre-trained models trained on UK-Biobank, a user might define and repeat a custom hand gesture a few times for fine-tuning.

To summarize, our scenario consists of three steps: (i) a model is pre-trained on a large amount of unlabeled data, (ii) it is fine-tuned with only a few labeled samples, and (iii) it is evaluated on test data. Pre-training occurs on a source domain D^s, which is distinct from the target domain D^t used for fine-tuning and testing.

Availability of Domain Labels. During pre-training, we assume access to domain labels within D^s. Domain labels (*e.g.*, device type, or anonymized user identifiers) are generally available as metadata, allowing us to distinguish data from different sources without exposing sensitive information.

Few-Shot Fine-Tuning. After deployment, users fine-tune the pre-trained model using their own data from the target domain D^t. We target scenarios where users can easily collect a small amount of labeled data, such as by repeating a few representative actions per class, to customize the model for their specific needs. Therefore, we assume that only a few labeled samples (*e.g.*, 10 per class) are available for fine-tuning.

3.2 Meta-Self-Supervised Learning

We propose an approach to adapt self-supervised models to a target domain, even when only a few data samples are available. To enable this, we enhance self-supervised pre-training to produce domain-adaptive weights, allowing the model to align with a specific domain using minimal data.

Our approach, *Meta-Self-Supervised Learning (MetaSSL)*, prepares models for adaptation by combining meta-learning with self-supervised objectives. Meta-learning [2], often referred to as "learning to learn," trains models to be fine-tuned effectively in new conditions with a few data. Inspired by its efficacy in traditional supervised settings, we design MetaSSL as a method for "learning to self-supervise." MetaSSL produces a model that is adaptive to few-shot self-supervised learning. To further adapt the model, we introduce an additional step that replays self-supervised training using data from the target domain (Section 3.3).

Algorithm 1 Domain-Specific Task Generation

Inputs: Pre-train dataset X, number of tasks M, and number of domain-specific tasks $M^{\text{dom}} < M$

Initialize empty task set $\mathcal{T}$
for $i \in \{1, 2, ..., M\}$ **do**
  if $i \leq M^{\text{dom}}$ **then**
    Select domain d_i from D^s uniformly at random
    Select K samples with domain d_i randomly: $\mathcal{S}_i = \{x \in X | \text{dom}(x) = d_i\}$ such that $|\mathcal{S}_i| = K$
    Select another set of K samples with d_i randomly: $\mathcal{Q}_i = \{x \in X | \text{dom}(x) = d_i\}$ such that $|\mathcal{Q}_i| = K$
  else
    Select K samples randomly: $\mathcal{S}_i = \{x \in X\}$ such that $|\mathcal{S}_i| = K$
    Select another set of K samples randomly: $\mathcal{Q}_i = \{x \in X\}$ such that $|\mathcal{Q}_i| = K$
  end if
  $\mathcal{T}_i \leftarrow (\mathcal{S}_i, \mathcal{Q}_i)$
  Update a set of task $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{T}_i$
end for
return Task set $\mathcal{T}$

Our implementation of MetaSSL builds on Model-Agnostic Meta-Learning (MAML) [10], which optimizes a model's initial weights for further updates with minimal data, making it well-suited to our approach. Additionally, MAML's flexibility allows MetaSSL to be applied across various self-supervised learning objectives [6, 41, 52] (Section 4.6.3). We detail the MetaSSL procedure in the next sections.

Domain-Specific Task Generation. MetaSSL simulates few-shot self-supervised learning through a set of *tasks*, $\mathcal{T}$, that emulate training and testing on a limited data. By default, each task $\mathcal{T}_i$ involves optimizing *domain-specific weights* θ_i on a small dataset, the *support set* $\mathcal{S}_i$. The optimized weights are then evaluated on a separate *query set* $\mathcal{Q}_i$. The evaluation results from all tasks are combined to compute a single loss, which guides the optimization of the model's *global weights* θ and reinforces its ability to adapt effectively across tasks.

Each task $\mathcal{T}_i$ is configured to operate within a single domain d_i, selected randomly from the source domain D^s using domain labels. This setup enables each task to simulate few-shot self-supervised learning and test within a specific domain, thus facilitating domain-specific weight optimization. We refer to these as *domain-specific tasks*. Following prior work [12], we introduce a small proportion (*e.g.*, 30%) of *multi-conditioned tasks*—tasks generated from random, domain-agnostic samples—to reduce the risk of overfitting to particular domains. These multi-conditioned tasks, formed from mixed-domain samples, act as synthetic domains that add diversity to the training. Algorithm 1 summarizes our domain-specific task generation process.

MetaSSL Optimization. MetaSSL pre-trains domain-adaptive weights by simulating few-shot self-supervised learning tasks within domains. Each domain-specific task $\mathcal{T}_i$, generated from a single domain, trains domain-specific weights θ_i on a support set $\mathcal{S}_i$ and evaluates performance on a corresponding query set $\mathcal{Q}_i$. Both training and evaluation use a self-supervised loss function

Algorithm 2 Meta-Self-Supervised Learning (MetaSSL)

Inputs: Pre-train dataset X, number of tasks M, number of domain-specific tasks M^{dom}, model weights θ, and self-supervised loss function $\mathcal{L}_{\text{SSL}}$

1: **for** epochs **do**
2: $\mathcal{T}$ = TaskGeneration(X, M, M^{dom}) ▷ Algorithm (1)
3: **for** $\mathcal{T}_i = (\mathcal{S}_i, \mathcal{Q}_i) \in \mathcal{T}$ **do**
4: Optimize task-specific weights using $\mathcal{S}_i$: $\theta_i \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}_{\text{SSL}}(\theta; \mathcal{S}_i)$
5: Evaluate loss of updated model on $\mathcal{Q}_i$: Compute $\mathcal{L}_{\text{SSL}}(\theta_i; \mathcal{Q}_i)$
6: **end for**
7: Update weights $\theta \leftarrow \theta - \beta \nabla_\theta \sum_i \mathcal{L}_{\text{SSL}}(\theta_i; \mathcal{Q}_i)$
8: **end for**

$\mathcal{L}_{\text{SSL}}$ to optimize task-specific weights to the domain. For example, when contrastive learning [6] is used, the loss function applied to $\mathcal{S}_i$ in a domain-specific task (same for $\mathcal{Q}_i$) is:

$$\mathcal{L}_{\text{SSL}}(\theta_i; \mathcal{S}_i = \{x \mid \text{dom}(x) = d_i\}) = - \sum_{x_j \in \mathcal{S}_i} \log \left(\frac{e^{\text{sim}(f_{\theta_i}(x'_j), f_{\theta_i}(x''_j))}}{e^{\text{sim}(f_{\theta_i}(x'_j), f_{\theta_i}(x''_j))} + \sum_{k \neq j} e^{\text{sim}(f_{\theta_i}(x'_j), f_{\theta_i}(x_k))}} \right), \tag{1}$$

where the parameterized encoder f_{θ_i} is optimized to maximize the similarity between two augmented views of the same data point (x'_i and x''_i) while minimizing similarity with other samples x_k, *i.e.*, negative samples. Here, $\mathcal{S}_i$ contains samples from a specific domain d_i when it is from a domain-specific task, ensuring that the contrastive loss is computed within a single domain. Note that our framework is agnostic to specific self-supervised methods, allowing $\mathcal{L}_{\text{SSL}}$ to be replaced with various objectives (*e.g.*, contrastive predictive coding [41] and multi-task learning [52]).

After evaluating the domain-specific weights on each $\mathcal{Q}_i$, results are aggregated across tasks to optimize the global model weights θ. This iterative process results in global model weights that are highly adaptable for few-shot self-supervised learning across different domains. Algorithm 2 outlines the full procedure of MetaSSL. α and β denote the learning rates for domain-specific training of θ_i and global model weights θ update.

Domain-Invariant Negative Queue. In MetaSSL, self-supervised learning is performed within domain-specific tasks, which results in small batch sizes. This poses a challenge when using contrastive learning [6], a popular self-supervised learning approach, as the objective. In contrastive learning, model performance depends on sufficient negative samples. Thus, it relies on large batch sizes (*e.g.*, 1024) to ensure effective training [19]. However, our MetaSSL involves very small batches (*e.g.*, 128), limiting the available negative samples.

To address this, we propose a *Domain-Invariant Negative Queue* to supplement the small batch sizes across domain-specific tasks. We implement a shared negative queue $\mathcal{N} = \{n_1, n_2, ..., n_k\}$ accessible to each task, where k is defined as a size sufficient for effective contrastive learning (*e.g.*, 1024). When contrastive learning is applied within a task, it draws negatives not only from the current task batch but also from the shared queue $\mathcal{N}$, enriching the pool of negatives for training. Building on Equation 1, we define the updated contrastive loss function with the Domain-Invariant Negative Queue as follows:

$$\mathcal{L}_{\text{SSL}}(\theta_i; \mathcal{S}_i = \{x \mid \text{dom}(x) = d_i\}, \mathcal{N}) = - \sum_{x_j \in \mathcal{S}_i} \log \left(\frac{e^{\text{sim}(f_{\theta_i}(x'_j), f_{\theta_i}(x''_j))}}{e^{\text{sim}(f_{\theta_i}(x'_j), f_{\theta_i}(x''_j))} + Z_j} \right), \quad \text{where } Z_j = \sum_{k \neq j} e^{\text{sim}(f_{\theta_i}(x'_j), f_{\theta_i}(x_k))} + \sum_{n \in \mathcal{N}} e^{\text{sim}(f_{\theta_i}(x'_j), f_{\theta_i}(n))}. \tag{2}$$

The queue $\mathcal{N}$ must provide effective negative samples across different domain-specific tasks. Hard negatives improve contrastive learning [19], introducing challenging examples that strengthen model robustness. Thus, we design the elements in $\mathcal{N}$ to function as *domain-invariant hard negatives*. This introduces a challenge, as randomly sampling negatives from the source domain D^s fails to consistently yield hard negatives for different domains, and identifying hard negatives for each domain-specific task is computationally costly.

To overcome this, we define the negative samples in $\mathcal{N}$ as trainable variables [21] and propose an adversarial training approach to optimize them as domain-invariant hard negatives. In this setup, the elements of $\mathcal{N}$ are updated adversarially during each iteration of MetaSSL to maximize domain-specific losses, effectively serving as *general* challenging examples across different domains. The elements in $\mathcal{N}$ are updated when the global weights θ are updated, following the aggregation of evaluation results from each query set $\mathcal{Q}_i$. The optimization of $\mathcal{N}$ follows a min-max problem:

$$\arg\max_{\mathcal{N}} \min_{\theta} \sum_i \mathcal{L}_{\text{SSL}}(\theta_i; \mathcal{Q}_i = \{x \mid \text{dom}(x) = d_i\}, \mathcal{N}). \tag{3}$$

Our Domain-Invariant Negative Queue enables contrastive learning within each domain-specific task by using an enriched pool of negative samples, even with small batch sizes. Later, in the target-side adaptation step, where target domain data is limited, the trained queue $\mathcal{N}$ is deployed alongside the model to serve as additional negative samples for the adaptation.

3.3 Replayed Self-Supervised Learning

The models pre-trained with MetaSSL are deployed to users, where each user's domain is considered a target domain D^t. Although these models are designed to be domain-adaptive, standard fine-tuning alone fails to adapt them to target domains. Standard fine-tuning relies on few-shot supervised learning, whereas MetaSSL pre-trains models specifically for few-shot self-supervised learning, which does not align with the supervised objective. Consequently, an additional adaptation step is needed to align the pre-trained model with the target domain.

We propose *Replayed Self-Supervised Learning (ReplaySSL)* as the adaptation method for aligning models to target domains. ReplaySSL replays the meta-learned self-supervised task as an adaptation step before supervised fine-tuning using the few-shot data set aside for fine-tuning. This procedure aligns the model more closely with the target domain, creating a *personalized* model.

ReplaySSL adapts the pre-trained weights θ using the few-shot fine-tuning data S from the target domain D^t with the same self-supervised learning objective used during MetaSSL, $\mathcal{L}_{\text{SSL}}$. This adaptation is completed in a few steps (*e.g.*, 10) with a fixed learning rate α, as formulated in the following equation:

$$\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}_{\text{SSL}}(\theta; S = \{x \mid \text{dom}(x) = D^t\}). \tag{4}$$

If $\mathcal{L}_{\text{SSL}}$ is defined as a contrastive learning loss, using the negative queue $\mathcal{N}$ trained from MetaSSL, the following optimization is applied:

$$\theta \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}_{\text{SSL}}(\theta; S = \{x \mid \text{dom}(x) = D^t\}, \mathcal{N}). \tag{5}$$

Importantly, ReplaySSL is effective only when coupled with MetaSSL. Replaying self-supervised learning without MetaSSL risks damaging the pre-trained model, as training on few-shot data often leads to an overfitted representation and fails to produce a generalizable model. MetaSSL addresses this by making the model adaptive to few-shot self-supervised settings, mitigating overfitting through domain-specific tasks structured within meta-learning. This synergy between MetaSSL and ReplaySSL is the key aspect of *SelfReplay*, and we validate their combined effectiveness through detailed ablation studies in Section 4.6.7.

After ReplaySSL, supervised learning fine-tunes the model on the downstream task. We apply linear evaluation protocol for this fine-tuning: only the classification head is trained, while the encoder remains frozen. This final, fine-tuned model is ready for use in the user application.

4 Experiments

4.1 Datasets

We evaluate *SelfReplay* on mobile sensing benchmarks across human activity recognition, gesture recognition, and stress level detection tasks. Each dataset has different domains of users or device positions. Our experiments assess performance under domain shift when self-supervised models are fine-tuned on different target domains. The following datasets are used:

ICHAR [14] comprises inertial measurement unit (IMU) data for classifying nine types of daily activities, such as walking, running, and stair climbing. Data was collected from ten participants using various mobile devices (seven smartphones and three watches). Each participant is treated as a unique domain.

HHAR [56] is designed for classifying six human activities collected from nine users with a combination of four smartwatches and eight smartphones. We define domains based on distinct user-device pairs.

PAMAP2 [50] classifies 12 different activity types using data collected from IMUs placed on three body locations: the wrist, chest, and ankle. Domains are divided by device positions.

DSA [1] encompasses a wide array of 19 daily and sports activities data, gathered from eight participants wearing five IMUs on the torso, arms, and legs. We define domains based on device positions.

NinaproDB5 [44] classifies hand gestures using 16-channel Surface Electromyographic (sEMG) signals. We use twelve gestures from Exercise A, focusing on basic finger movements as in prior work [5]. Domains are defined based on individual users (Session 4.6.1) and session variations (Session 4.6.4), where sessions represent different recordings that introduce temporal variability.

WESAD [53] classifies stress levels across neutral, stress, and amusement states. Physiological and motion data were collected from fifteen participants using wrist- and chest-worn devices. We used only the chest Electrocardiogram (ECG) signals, the most reliable modality for stress detection [53]. Domains are defined based on individual users.

Opportunity [51] captures motion sensor recordings of users performing daily activities. We use accelerometer data from the right wrist to classify four primitive activities: standing, walking, sitting, and lying. To evaluate temporal domain shift, we define domains based on session variations, where each session corresponds to a distinct recording instance.

4.2 Data Preprocessing

For the human activity recognition datasets, we segmented data using a fixed window size of 256 with an overlap of 128. We standardized each dataset to a range of -1 to 1, following data processing settings from prior work [14]. We focused exclusively on 3-channel accelerometer data from all sources to reproduce existing baselines [17] and excluded domains with fewer than 500 samples to ensure sufficient training data. As a result, we used 20 domains for HHAR, representing combinations of five users and four devices. For WESAD, we applied a window size of 10 seconds with a sliding interval of 0.25 seconds, creating approximately 7,000 windows to reduce the computational cost, following a prior study [3]. For NinaproDB5, we used a window size of 60 (0.3 seconds) with a 50% overlap, following the common practice of using small windows for sEMG data [25, 48].

To split data for self-supervised pre-training, fine-tuning, and testing, we followed the approach of previous studies on self-supervised learning for sensing [17]. Specifically, we allocated 70% of the data for pre-training—7% for validation and 63% for training—and used the remaining 30% for few-shot fine-tuning. In the few-shot setting, we sampled a few instances per class (*e.g.*, 1, 2, 5, or 10 samples). The remaining data was split evenly for validation and testing. We ensured no temporal overlap between samples in different splits, preserving data independence.

For our evaluation, we composed pre-training in domains separate from fine-tuning and testing. We prepared fine-tuning and testing sets for each target domain and composed a pre-training dataset by sampling exclusively from other domains. This process was repeated across all target domains to ensure a consistent setup.

4.3 Baselines

We benchmark *SelfReplay* against baselines chosen for their efficacy in mitigating domain shift between the unsupervised pre-training and the following fine-tuning. Note that most existing domain generalization [36, 45, 47] and adaptation [4, 14, 60, 73] methods assume labeled data for pre-training and thus they do not apply to our scenario. We found two approaches that fit our scenario.

DARLING [71] is a domain generalization approach tailored for contrastive learning. DARLING optimizes the loss by using intra-domain negative samples, encouraging discrimination within each domain. This process enables the model to learn domain-invariant features that can be fine-tuned across different domains.

Table 1: F1-scores of *SelfReplay* and baseline methods for 10-shot fine-tuning across six datasets, with the highest scores in bold and the second-highest underlined.

		Domain: User				Domain: Position		
Pre-train	Fine-tune	ICHAR	HHAR	NinaproDB5	WESAD	PAMAP2	DSA	Avg.
SimCLR [57]	Linear eval.	0.745 ± 0.024	<u>0.866</u> ± 0.008	<u>0.446</u> ± 0.012	0.848 ± 0.024	0.549 ± 0.016	0.391 ± 0.006	<u>0.641</u> ± 0.015
	End-to-end	0.663 ± 0.028	0.836 ± 0.029	0.405 ± 0.032	0.869 ± 0.023	<u>0.589</u> ± 0.046	0.253 ± 0.022	0.602 ± 0.030
Set-SimCLR [30]	Linear eval.	<u>0.758</u> ± 0.010	0.814 ± 0.004	0.154 ± 0.012	0.813 ± 0.012	0.487 ± 0.011	0.283 ± 0.007	0.552 ± 0.009
	End-to-end	0.747 ± 0.029	0.848 ± 0.016	0.244 ± 0.034	<u>0.882</u> ± 0.016	0.573 ± 0.015	0.165 ± 0.012	0.577 ± 0.020
DARLING [71]	Linear eval.	0.749 ± 0.019	0.831 ± 0.003	0.303 ± 0.013	0.789 ± 0.051	0.551 ± 0.012	<u>0.399</u> ± 0.008	0.604 ± 0.018
	End-to-end	0.656 ± 0.019	0.844 ± 0.026	0.324 ± 0.028	0.860 ± 0.030	0.580 ± 0.042	0.258 ± 0.024	0.587 ± 0.028
SelfReplay (ours)		**0.839** ± 0.023	**0.912** ± 0.009	**0.464** ± 0.021	**0.883** ± 0.018	**0.680** ± 0.027	**0.632** ± 0.014	**0.735** ± 0.019

Set-SimCLR [30] is an unsupervised meta-learning method that employs a set encoder to enhance the agreement between augmented sample sets originating from identical sources. Both an instance encoder and the set encoder are trained through contrastive learning. In fine-tuning, the set encoder generates class prototypes from sample sets by class and is used to set the initial weights of the classifier. The classifier, adjusted by the prototypes, facilitates rapid adaptation to novel conditions with the initial weights.

4.4 Implementation

While *SelfReplay* serves as a model- and method-agnostic approach applicable to various self-supervised learning methods, our primary implementation was based on SimCLR [6] to ensure a fair comparison with baselines. This selection aligns with DARLING and Set-SimCLR, which were also presented based on contrastive learning. All baseline models use the same network architecture as *SelfReplay* to maintain consistency.

Our backbone network is implemented with 1D convolutional neural networks (CNNs), followed by a projection head consisting of a fully connected layer. The architecture and hyperparameters are based on SimCLR practices for sensing tasks as described in a previous study [17]. We optimized hyperparameters through a grid search: for pre-training, we explored learning rates from $\{0.0001, 0.0005, 0.001, 0.005\}$ and weight decays from $\{0, 0.0001\}$. Baseline batch sizes were tested at $\{1024, 2048, 4096\}$, while *SelfReplay* batches were constructed within each domain-specific task, resulting in smaller batch sizes. During fine-tuning, we used a fixed learning rate of 0.005 for linear evaluation and 0.001 for end-to-end fine-tuning. Our main evaluation used $\{1, 2, 5, 10\}$-shot samples per class, with additional tests on smaller sample sizes. Using the Adam optimizer, we trained models for 100 epochs in pre-training and 20 epochs in fine-tuning.

For MetaSSL, we optimized the meta-learning rate (β) and weight decay within the same range as the baselines, with additional tuning for task-specific learning rates (α) from $\{0.001, 0.005, 0.01\}$ and inner iteration steps from $\{10, 20, 30\}$. ReplaySSL followed the same parameter setup as MetaSSL. We fixed the number of domain-specific tasks at eight, multi-conditioned tasks at four, and task size at 128. Since meta-learning requires extended training for convergence, we ran MetaSSL for 5,000 epochs. For the Domain-Invariant Negative Queue, we tested sizes of $\{1024, 2048, 4096\}$, selecting the optimal size to align with the SimCLR baseline batch size. Negative queue elements were optimized using an adversarial objective (Equation 3) with the Adam optimizer and a learning rate of 1. We implemented all methods in PyTorch and conducted training on eight NVIDIA TITAN Xp GPUs.

4.5 Evaluation Protocols and Metric

We employed a leave-one-domain-out setting [45]. For each domain in the dataset, we designated it as the target domain for fine-tuning and testing, while all other domains were used for pre-training. This evaluation was conducted across all domains, with each domain rotated as the target, and the results were averaged. We selected a few samples per class (*e.g.*,1, 2, 5, and 10) for fine-tuning and evaluated performance within the same target domain.

We applied two fine-tuning protocols: linear and end-to-end fine-tuning. Linear evaluation served as the primary protocol for *SelfReplay*, treating the pre-trained encoder as a frozen feature extractor and training only a linear classification layer. Since *SelfReplay*'s ReplaySSL refines the encoder parameters, we also conducted end-to-end fine-tuning for baseline methods, updating the entire network without freezing the encoder. This ensured a fair comparison by enabling encoder parameter updates in both cases.

All evaluations were performed using five random seeds, and the results are reported as the mean and standard deviation. To assess performance, we used the macro-averaged F1-score, which is well-suited for handling class imbalances in the data.

4.6 Results

4.6.1 Main Evaluation. Table 1 shows the performance of *SelfReplay* compared with domain generalization and domain adaptation baselines, using 10-shot fine-tuning. Bolded values indicate the highest scores in each column. *SelfReplay* consistently achieves the highest F1 scores across all datasets. The results indicate that baseline models struggle to capture domain-specific features as they rely on weights pre-trained in the source domain. While end-to-end fine-tuning occasionally improves performance, its impact varies widely depending on the dataset, likely due to few-shot fine-tuning's sensitivity. Baselines do not fully benefit from end-to-end fine-tuning without an adaptive pre-training design. In contrast, *SelfReplay* achieves an average F1-score improvement of 9.4%p, establishing it as a robust domain adaptation method.

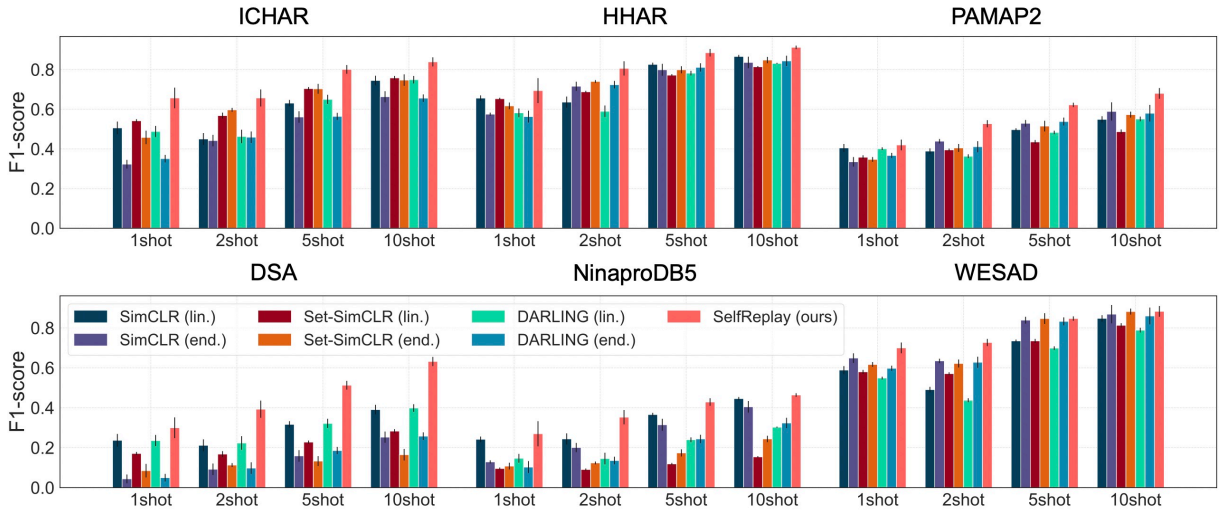

Figure 4: Average F1-scores of *SelfReplay* and the baselines across different shot numbers (1, 2, 5, and 10).

Table 2: F1-scores of *SelfReplay* based on different self-supervised learning methods (SimCLR, CPC, and Multi-Task Learning). The highest scores are in bold.

		Domain: User		Domain: Position		
Pre-train	Fine-tune	ICHAR	HHAR	PAMAP2	DSA	Avg.
SimCLR [57]	Linear eval.	0.745 ± 0.024	0.866 ± 0.008	0.549 ± 0.016	0.391 ± 0.006	0.638 ± 0.014
	End-to-end	0.663 ± 0.028	0.836 ± 0.029	0.589 ± 0.046	0.253 ± 0.022	0.585 ± 0.031
SelfReplay$_{\texttt{SimCLR}}$ (ours)		**0.839** ± 0.023	**0.912** ± 0.009	**0.680** ± 0.027	**0.632** ± 0.014	**0.735** ± 0.019
CPC [18]	Linear eval.	0.765 ± 0.016	0.846 ± 0.005	0.379 ± 0.017	0.371 ± 0.005	0.590 ± 0.011
	End-to-end	0.816 ± 0.013	0.849 ± 0.021	0.484 ± 0.026	0.352 ± 0.017	0.625 ± 0.019
SelfReplay$_{\texttt{CPC}}$ (ours)		**0.826** ± 0.008	**0.871** ± 0.005	**0.527** ± 0.017	**0.419** ± 0.008	**0.661** ± 0.009
Multi-Task [52]	Linear eval.	0.716 ± 0.010	0.877 ± 0.003	0.630 ± 0.003	0.456 ± 0.004	0.670 ± 0.005
	End-to-end	0.718 ± 0.019	0.865 ± 0.030	0.636 ± 0.015	0.378 ± 0.021	0.649 ± 0.021
SelfReplay$_{\texttt{MultiTask}}$ (ours)		**0.794** ± 0.015	**0.891** ± 0.005	**0.659** ± 0.016	**0.578** ± 0.011	**0.731** ± 0.012

4.6.2 Performance across Different Shots. We evaluated the performance of *SelfReplay* compared with the baselines by fine-tuning with a smaller number of samples (*e.g.*, 1, 2, and 5 shots per class). Figure 4 shows the results. For all datasets and shot settings, *SelfReplay* consistently achieves the best performance, with improvements of 5.9%, 14.3%, 12.1%, and 9.4% over the second-best method, respectively. These results demonstrate the robustness of our approach across varying few-shot scenarios, indicating that *SelfReplay* remains effective in data-scarce mobile sensing applications.

4.6.3 Integration with Self-Supervised Learning Methods. We designed *SelfReplay* to be agnostic to specific self-supervised learning methods. For evaluation, we implemented *SelfReplay* with two additional self-supervised objectives: contrastive predictive coding (CPC) [41] (*SelfReplay*$_{\texttt{CPC}}$) and multi-task learning [52] (*SelfReplay*$_{\texttt{MultiTask}}$). Each version replaces the self-supervised loss function $\mathcal{L}_{\texttt{SSL}}$ (from Equation 1) with the objective corresponding to its respective method, CPC or multi-task learning. CPC defines an objective for predicting the embedding of a future window based on past embeddings, contrasting the true future window against other candidates. Multi-task learning assigns the model several tasks focused on identifying data transformations and promoting shared feature learning across tasks.

In implementing *SelfReplay*$_{\texttt{CPC}}$ and *SelfReplay*$_{\texttt{MultiTask}}$ along with their baselines, we followed the same architecture and parameter tuning settings from prior assessments [17]. For parameter tuning, we used the same search settings as outlined in Section 4.4. The only adjustment was in batch size, which was reduced for CPC and multi-task learning, searching across {64, 128, 256} for optimal results. Since CPC and multi-task learning were primarily designed for human activity recognition [17], we focused our evaluations on four relevant datasets.

Table 2 presents the results across different self-supervised learning methods. *SelfReplay* consistently improved performance with average gains of 4.4%p and 6.1%p for CPC and multi-task learning, respectively. Performance gains varied depending on the self-supervised method used on the same dataset, which we analyze further in Section 4.6.9. Additionally, Figure 5 illustrates *SelfReplay*'s effectiveness across k-shot settings, underscoring its robustness. These results demonstrate that *SelfReplay* can be applied flexibly across different applications, adapting to the most effective self-supervised learning method for each setting.

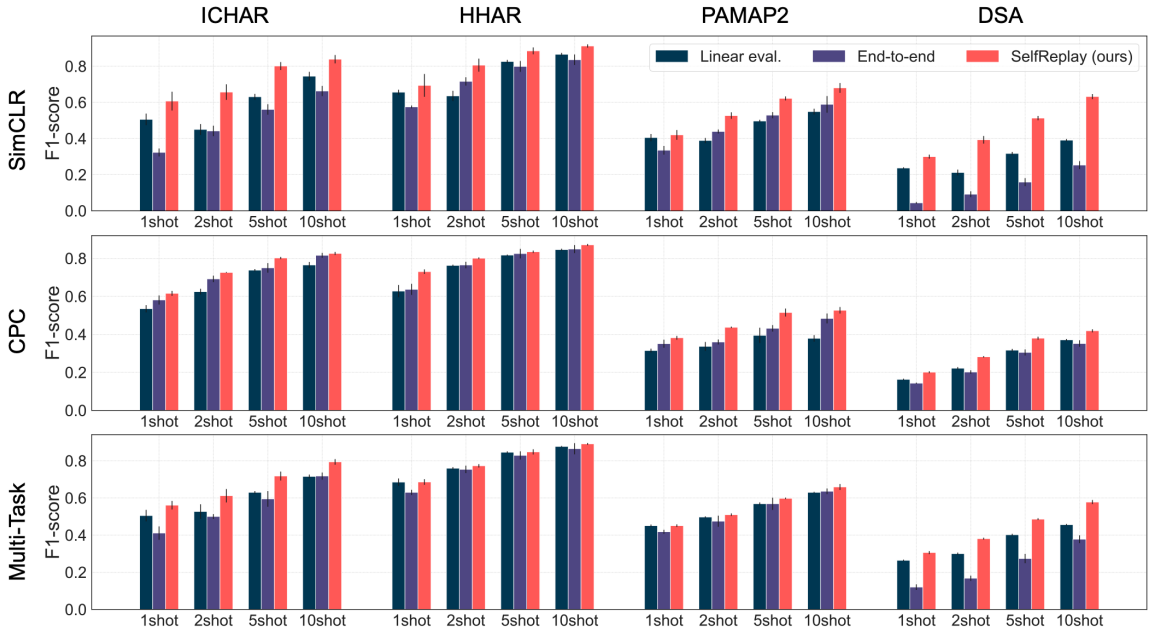

Figure 5: Average F1-scores of *SelfReplay* based on different self-supervised learning methods (SimCLR, CPC, and multi-task learning) across different shot numbers (1, 2, 5, and 10).

Table 3: F1-scores of *SelfReplay* and baseline methods for 10-shot fine-tuning on two datasets, where domains are defined by sessions to evaluate temporal domain shift. The highest scores are in bold and the second-highest are underlined.

		Domain: Session	
Pre-train	Fine-tune	Opportunity	NinaproDB5
SimCLR [57]	Linear eval.	0.427 ± 0.007	0.676 ± 0.006
	End-to-end	<u>0.458</u> ± 0.001	<u>0.693</u> ± 0.004
Set-SimCLR [30]	Linear eval.	0.438 ± 0.009	0.206 ± 0.003
	End-to-end	0.445 ± 0.009	0.287 ± 0.007
DARLING [71]	Linear eval.	0.424 ± 0.004	0.657 ± 0.005
	End-to-end	0.456 ± 0.010	0.686 ± 0.003
SelfReplay (ours)		**0.476** ± 0.008	**0.724** ± 0.003

4.6.4 Robustness to Temporal Domain Shift. We evaluate an extended domain scope by testing *SelfReplay* on temporal shifts within the same user and device. In this setting, *sessions* serve as temporal domains, capturing changes in user behavior and environmental conditions over time [54].

We used two datasets, Opportunity and NinaproDB5, providing session separation with distinct measurement times. Since we focus on sessions within a single user, the available pre-training data was limited. To address this, we reduced the window generation step size to 12.5% and expanded the batch size search space to {64, 128, 256, 1024, 2048, 4096} to find an optimal configuration. Experiments were conducted on a randomly selected user.

Table 3 shows *SelfReplay* outperformed all baselines on both datasets. In NinaproDB5, using sessions as domains yielded higher performance than using users as domains (Section 4.6.1). Still, temporal shifts persisted, and *SelfReplay* demonstrated the highest robustness. Our findings indicate that *SelfReplay* can be effectively deployed in continuous sensing systems, dynamically adapting to temporal changes through fine-tuning with recent data.

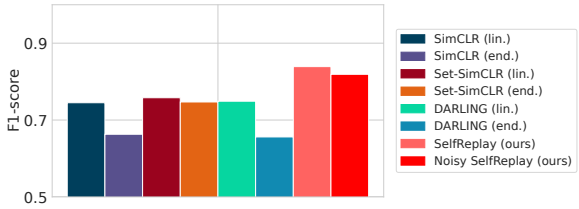

Figure 6: Average F1-scores of *SelfReplay* and baselines with and without noisy domain labels (50%) during pre-training.

4.6.5 Robustness to Noisy Domain Labels. We assume that MetaSSL has access to domain labels during pre-training. To assess the impact of domain labels, we introduced noise into 50% of the domain labels by assigning incorrect values and evaluated MetaSSL on the ICHAR dataset with SimCLR as the base self-supervised method.

Figure 6 shows that with noisy domain labels, *SelfReplay*'s F1 score decreased modestly from 0.839 to 0.819, indicating that label quality does influence performance. Nonetheless, *SelfReplay* still outperformed baseline methods, whose highest score was 0.758. These findings suggest that while accurate domain labels are preferable, the meta-learning framework based on small meta-tasks keeps *SelfReplay* robust even under noisy conditions.

4.6.6 Robustness to Model Size. To assess how larger models behave under domain shift, we evaluated the effect of model size in our setting by increasing the number of layers in our model, which expanded the parameter count from 148.6K to 1.4M. Figure 7 illustrates how larger models affect fine-tuning performance in a new domain compared with smaller models on the ICHAR dataset.

Table 4: F1-score comparison of *SelfReplay* against 1) the baseline self-supervised learning method, 2) *SelfReplay* without meta-learning, and 3) *SelfReplay* without replay. 10-shot fine-tuning is performed.

Components		Domain: User				Domain: Position		
MetaSSL	ReplaySSL	ICHAR	HHAR	NinaproDB5	WESAD	PAMAP2	DSA	Avg.
✗	✗	0.745 ± 0.024	0.866 ± 0.008	0.446 ± 0.012	0.848 ± 0.024	0.549 ± 0.016	0.391 ± 0.006	0.641 ± 0.014
✗	✓	0.731 ± 0.029	0.638 ± 0.044	0.436 ± 0.018	0.853 ± 0.021	0.464 ± 0.063	0.294 ± 0.011	0.569 ± 0.037
✓	✗	0.792 ± 0.047	0.880 ± 0.022	**0.464 ± 0.021**	0.882 ± 0.018	0.677 ± 0.021	0.620 ± 0.012	0.719 ± 0.026
✓	✓	**0.839 ± 0.023**	**0.912 ± 0.009**	**0.464 ± 0.021**	**0.883 ± 0.018**	**0.680 ± 0.027**	**0.632 ± 0.014**	**0.735 ± 0.019**

Table 5: F1-score comparison of *SelfReplay* (w/ Negative Queue) against the baseline self-supervised learning method and *SelfReplay* without the Domain-Invariant Negative Queue (w/o Negative Queue). Results are based on 10-shot fine-tuning.

	Domain: User				Domain: Position		
	ICHAR	HHAR	NinaproDB5	WESAD	PAMAP2	DSA	Avg.
Baseline	0.745 ± 0.024	0.866 ± 0.008	0.446 ± 0.012	0.848 ± 0.024	0.549 ± 0.016	0.391 ± 0.006	0.641 ± 0.014
w/o Negative Queue	0.836 ± 0.011	0.903 ± 0.004	0.449 ± 0.018	0.879 ± 0.020	0.639 ± 0.030	0.526 ± 0.019	0.705 ± 0.017
w/ Negative Queue	**0.839 ± 0.023**	**0.912 ± 0.009**	**0.464 ± 0.021**	**0.883 ± 0.018**	**0.680 ± 0.027**	**0.632 ± 0.014**	**0.735 ± 0.019**

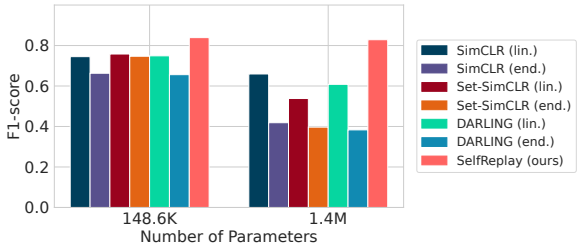

Figure 7: Average F1-scores of *SelfReplay* and baselines for a small (148.6K) model versus a large (1.4M) model.

Our findings reveal a notable trend: larger models exhibited increased overfitting to the source domain for all baselines, leading to worse fine-tuning performance on the target domain. In contrast, *SelfReplay* maintained the performance, demonstrating robustness to domain shift even in a large model.

These results suggest that *SelfReplay* effectively mitigates domain shift regardless of model size. In particular, as foundation models for sensing applications are expected to be large, our findings indicate that *SelfReplay* can enhance their generalizability across domains.

4.6.7 Ablation Study: MetaSSL and ReplaySSL. We conducted an ablation study to examine the contributions of the key components of *SelfReplay*: MetaSSL and ReplaySSL. We assessed their individual and combined impact by comparing *SelfReplay*'s performance with and without these components.

Table 4 shows the results. A key observation is that using ReplaySSL alone, without MetaSSL, leads to a significant performance drop, even falling below the baseline. This suggests that applying self-supervised learning on a limited target domain dataset can lead to overfitting, distorting the pre-trained model weights. This finding underscores MetaSSL's role as the essential component that enables effective adaptation through ReplaySSL.

With MetaSSL, *SelfReplay* outperforms the baseline, even without ReplaySSL. This result indicates that self-supervised meta-learning promotes meaningful feature learning, which we attribute to meta-learning's capacity to capture transferable features that support effective adaptation. By effectively "learning to learn," MetaSSL enhances the model's ability to leverage generalizable features across domains, which provides a foundation for rapid adaptation with limited data.

Combining MetaSSL and ReplaySSL ultimately achieves the best performance, showing that the adaptation step further enhances the model. This highlights the synergistic value of using MetaSSL and ReplaySSL together.

4.6.8 Ablation Study: Domain-Invariant Negative Queue. We conducted an ablation study to assess the impact of Domain-Invariant Negative Queue in $SelfReplay_{\texttt{SimCLR}}$. Without the domain-invariant negative queue, negative samples are drawn exclusively from each domain-specific task, resulting in a limited set of negatives. By contrast, using the negative queue allows access to over 1024 diverse negative samples.

Table 5 shows the results. Although performance improves without the negative queue relative to the baseline (SimCLR), the Domain-Invariant Negative Queue further enhances performance, with an average improvement of 3%p. This result indicates the effectiveness of the negative queue, as it mitigates the limitations of small batch sizes within *SelfReplay*.

4.6.9 Domain Effect of Self-Supervised Learning Methods. Our evaluation of *SelfReplay* across different self-supervised learning methods (Section 4.6.3) revealed varying impacts depending on the method. To investigate this difference, we examined how domain shifts affect each method. We pre-trained models using SimCLR, CPC, and multi-task learning in a leave-one-domain-out setup and fine-tuned them in a novel domain (*out-of-domain*) setting. For comparison, we created an *in-domain* baseline, where pre-training and fine-tuning occur within the same domain. We measure each method's sensitivity to domain shifts by comparing performance

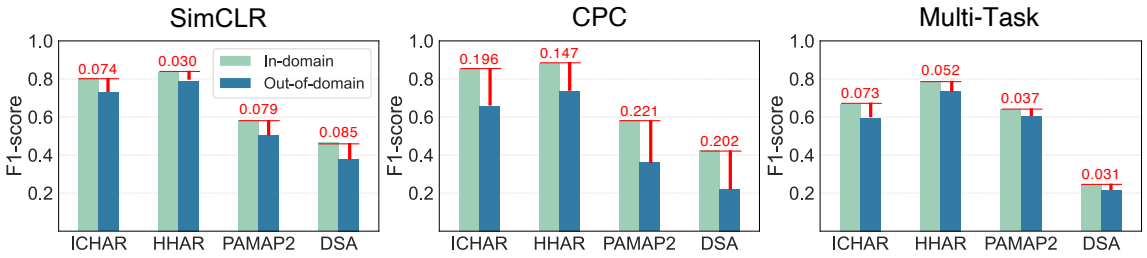

Figure 8: Fine-tuning performance comparison between models pre-trained in-domain and out-of-domain settings. 10-shot fine-tuning is performed for all settings. Performance drops between the settings are shown in red.

drops from in-domain with out-of-domain. We equalized data sizes across both settings to ensure fair comparisons.

Figure 8 illustrates the results. Each self-supervised method experiences a drop in performance when fine-tuned and tested on out-of-domain data, highlighting the challenge of domain shifts. However, the level of decline differs by method: CPC shows the largest drop, with an average F1-score reduction of 19.15%p, while SimCLR and multi-task learning have moderate decreases of 6.7%p and 4.95%p, respectively.

These results indicate that the self-supervised learning method influences a model's sensitivity to domain shifts. CPC, which trains the model to predict future segments, often learns patterns specific to the source domain. For example, a model pre-trained from younger and active users might focus on a pattern of increasing activity over time. When transferred to a domain with older users, whose activity decreases over time, the CPC model struggles because its predictive patterns do not align with the new domain's temporal dynamics. In contrast, multi-task learning's objective of identifying data augmentations is less domain-specific. This knowledge transfers more effectively between user groups, as recognizing augmented characteristics (*e.g.*, rotation) is independent of specific user trends.

In summary, domain shifts generally lead to performance drops during fine-tuning, with the degree of decline varying by a self-supervised method. Our findings underscore the importance of selecting suitable self-supervised methods for effective deployment in heterogeneous mobile environments. We also stress that self-supervised learning methods consider the effects of domain shift across different settings.

4.6.10 MetaSSL Overhead. Unlike standard self-supervised learning methods that process the entire dataset in each epoch, MetaSSL trains on small meta-tasks per epoch (*e.g.*, 12 tasks with a size of 128). As a result, the model observes only a limited portion of the dataset at a time. Therefore, we increased the number of epochs to 5000, which is higher than the standard SSL setting of 100 epochs, to ensure MetaSSL observes enough data and converges. In this setting, we measured the execution time and VRAM usage for MetaSSL and standard SSL methods using a single NVIDIA TITAN XP GPU. Experiments were conducted using the ICHAR dataset.

Although MetaSSL trains smaller data per iteration, its overall training time was longer due to the additional overhead of meta-task construction, aggregation, and serial task-specific gradient updates following the MAML implementation. As a result, the total runtime increased from 3.7 minutes to 3.3 hours. VRAM usage increased slightly from 1.5GB to 1.6GB, as MetaSSL maintains gradients across multiple meta-tasks rather than just within mini-batches. The overhead was more pronounced for CPC, where the runtime increased from 23 minutes to 23 hours, and VRAM usage grew from 0.94GB to 7.7GB. This was due to CPC's reliance on large embedding variables, which MetaSSL maintains across meta-tasks. For multi-task learning, the total runtime increased from 10.9 minutes to 4.3 hours, and VRAM usage grew from 0.4GB to 1GB.

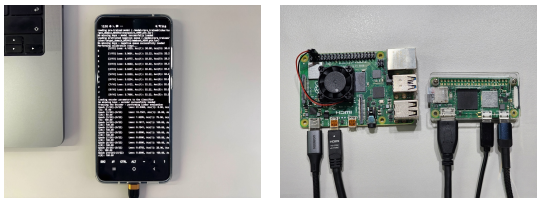

Figure 9: Testbed setups for on-device execution of *SelfReplay* on three devices: Samsung Galaxy S20 Ultra (12GB RAM, left), Raspberry Pi 4 (4GB RAM, middle), and Raspberry Pi Zero 2 W (512MB RAM, right).

The source of overhead is our current MAML-based implementation, which is straightforward but processes meta-tasks sequentially. It is important to note that MetaSSL is conducted on servers where pre-training benefits from abundant resources. Meanwhile, to further optimize training costs, more efficient meta-learning methods such as Reptile [40] or parallelized meta-task updates could improve efficiency. Additionally, refining task generation strategies may further reduce computational overhead.

4.6.11 ReplaySSL Overhead. We aim to enable the practical deployment of pre-trained models to users, particularly those with resource-constrained mobile environments. To this end, we assess the computational feasibility of *SelfReplay* for mobile devices, focusing on its user-side operations, ReplaySSL, and fine-tuning. We performed on-device training with three edge devices (Figure 9): a Samsung Galaxy S20 Ultra, a Raspberry Pi 4, and a Raspberry Pi Zero 2 W. The Samsung Galaxy S20 Ultra had 12GB of RAM, and on-device training was implemented via Termux [37], a Linux terminal emulator for Android, to execute PyTorch-based training code. The Raspberry Pi 4 had 4GB of RAM and runs Ubuntu as its operating system. To simulate a more constrained environment, we used a Raspberry Pi Zero 2 W with only 512MB of RAM; lacking internal storage, we used flash memory as swap space.

Table 6 presents the overhead of ReplaySSL and fine-tuning, measured independently across different self-supervised learning

Table 6: Computational overhead of ReplaySSL followed by fine-tuning across different self-supervised learning methods on three edge devices: Samsung Galaxy S20 Ultra, Raspberry Pi 4, and Raspberry Pi Zero 2 W.

Metric	SimCLR		CPC		Multi-Task	
	Replay SSL	Fine-Tune	Replay SSL	Fine-Tune	Replay SSL	Fine-Tune
Galaxy S20 Ultra						
Time (sec)	9.66	16.31	73.54	21.46	6.06	25.18
CPU (%)	669.29	595.47	589.54	565.60	584.50	480.28
Mem (MB)	83.38	89.29	339.90	36.86	99.84	48.41
Raspberry Pi 4						
Time (sec)	33.62	21.64	330.29	54.87	20.04	40.37
CPU (%)	65.50	62.88	89.39	76.82	53.99	56.41
Mem (MB)	62.31	71.01	714.63	712.62	47.45	57.53
Raspberry Pi Zero 2 W (using flash memory for swap space)						
Time (sec)	88.45	46.43	4539.86	126.35	58.66	73.88
CPU (%)	61.32	66.57	32.48	79.83	57.78	61.49
Mem (MB)	81.69	102.06	762.55	674.00	112.87	73.39

methods. On Samsung Galaxy S20 Ultra, all operations were completed in tens of seconds under the few-shot setting. Notably, ReplaySSL with SimCLR or multi-task learning took under 10 seconds, consuming under 100MB of memory—requiring even less time than fine-tuning. Consequently, all user-side operations of *SelfReplay* could be performed in under 30 seconds on the smartphone. Although CPC required more resources, it was completed within 1.5 minutes while using approximately 340MB of memory. On Raspberry Pi 4, processing times increased by roughly three to four times compared to the smartphone; SimCLR and multi-task learning finished within one minute, and CPC took about six minutes, all while using under 1GB of memory. Even on the highly constrained Raspberry Pi Zero 2 W, SimCLR and multi-task learning remain feasible in under two minutes, whereas CPC's higher memory requirements trigger frequent swapping, prolonging execution to about 1.26 hours. Importantly, the adaptation step (ReplaySSL) is needed only once per user after obtaining the deployed model. Our findings confirm that end-users can conduct all necessary operations of *SelfReplay* on the device with manageable computational overhead.

5 Discussion

5.1 Adapting to Changing Environments

We designed *SelfReplay* with a single domain adaptation step once self-supervised models are deployed to users. However, data characteristics within a single domain can change over time due to changing environments. This implies that the adapted model might not perform optimally as domain characteristics change continuously. We anticipate the potential for domain adaptation through ReplaySSL in such scenarios, as our adaptation step does not require user labels. This allows us to continuously adapt the model using the ongoing data stream from the user. Enhancing the efficiency and effectiveness of this approach in such dynamic scenarios is a direction for future work.

5.2 Expanding Domain Coverage

Our domain-specific task generation in MetaSSL is designed to reflect target-domain characteristics by composing tasks based on the same user or device information. In this approach, MetaSSL relies on sufficiently diverse domains during pre-training to learn effective adaptation strategies. If the pre-training data lack variety, the model may fail to generalize, highlighting the need for large-scale, diverse domain coverage.

As a future direction, more advanced meta-task generation approaches could be designed to handle the complexity of real-world applications. For instance, incorporating tasks involving a broader range of domains (*e.g.*, different modalities and user contexts) would enable pre-trained models to adapt to varying deployment environments. Another promising approach involves reducing MetaSSL's reliance on domain diversity by devising methods that maintain generalizability with limited pre-training data, thus enhancing scalability. Ultimately, refining task generation to capture real-world variability will be a priority to ensure that MetaSSL remains robust across real-world applications.

5.3 Extending Self-Supervised Methods

Our findings underscore the impact of domain shift on different self-supervised learning methods. We observed that the improvement from our domain adaptation varies with the type of self-supervised learning method applied. This suggests a need for a deeper understanding of how domain shifts affect various self-supervised learning approaches. Although our results shed light on the domain shift effects for established methods such as SimCLR, CPC, and multi-task learning, the behavior of numerous other self-supervised learning methods [8, 9, 15, 22, 42, 66] under domain shifts remains unexplored. Addressing this as future work is an essential step in the field.

6 Conclusion

We explored the domain shift challenge in mobile sensing, where self-supervised models are fine-tuned to heterogeneous domains. To address this, we proposed *SelfReplay*, an adaptive meta-task replay approach for self-supervised learning. *SelfReplay* combines MetaSSL, which uses meta-learning to produce self-supervised models prepared for domain adaptation, with ReplaySSL, an adaptation step that replays the meta-learned self-supervised task on target domain data. Our evaluation across different mobile sensing tasks demonstrates that *SelfReplay* consistently outperforms existing self-supervised learning and domain generalization methods, achieving an average F1-score improvement of 9.4%p. Additionally, *SelfReplay* is computationally efficient, completing adaptation on a smartphone in under a few minutes. These findings validate *SelfReplay* as a practical framework for enhancing pre-trained models for end-users with minimal overhead.

Acknowledgments

This work is funded in part by the National Research Foundation of Korea (NRF), funded by the Ministry of Science and ICT (MSIT) under grant RS-2024-00464269, the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (RS-2024-00337007), and the Hong Kong GRF 16204224.

References

[1] Kerem Altun, Billur Barshan, and Orkun Tunçel. 2010. Comparative study on classifying human activities with miniature inertial and magnetic sensors. *Pattern Recogn.* 43, 10 (Oct. 2010), 3605–3620. https://doi.org/10.1016/j.patcog.2010.04.019

[2] Marcin Andrychowicz, Misha Denil, Sergio Gómez Colmenarejo, Matthew W. Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando de Freitas. 2016. Learning to learn by gradient descent by gradient descent. In *Proceedings of the 30th International Conference on Neural Information Processing Systems* (Barcelona, Spain) *(NIPS'16)*. Curran Associates Inc., Red Hook, NY, USA, 3988–3996.

[3] Anubhav Bhatti, Behnam Behinaein, Dirk Rodenburg, Paul Hungler, and Ali Etemad. 2021. Attentive Cross-modal Connections for Deep Multimodal Wearable-based Emotion Recognition. In *2021 9th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW)*. IEEE, Nara, Japan, 01–05. https://doi.org/10.1109/ACIIW52867.2021.9666360

[4] Youngjae Chang, Akhil Mathur, Anton Isopoussu, Junehwa Song, and Fahim Kawsar. 2020. A systematic study of unsupervised domain adaptation for robust human-activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–30.

[5] Lin Chen, Jianting Fu, Yuheng Wu, Haochen Li, and Bin Zheng. 2020. Hand gesture recognition using compact CNN via surface electromyography signals. *Sensors* 20, 3 (2020), 672.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning (ICML'20)*. JMLR.org, Virtual Event, Article 149, 11 pages.

[7] Gaole Dai, Huatao Xu, Hyungjun Yoon, Mo Li, Rui Tan, and Sung-Ju Lee. 2024. ContrastSense: Domain-invariant Contrastive Learning for In-the-Wild Wearable Sensing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 8, 4, Article 162 (Nov. 2024), 32 pages. https://doi.org/10.1145/3699744

[8] Shohreh Deldari, Hao Xue, Aaqib Saeed, Daniel V. Smith, and Flora D. Salim. 2022. COCOA: Cross Modality Contrastive Learning for Sensor Data. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 3, Article 108 (Sept. 2022), 28 pages. https://doi.org/10.1145/3550316

[9] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee-Keong Kwoh, and Xiaoli Li. 2023. Self-supervised learning for label-efficient sleep stage classification: A comprehensive evaluation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 31 (2023), 1333–1342.

[10] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70 (ICML'17)*. JMLR.org, Sydney, NSW, Australia, 1126–1135.

[11] Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15)*. JMLR.org, Lille, France, 1180–1189.

[12] Taesik Gong, Yeonsu Kim, Ryuhaerang Choi, Jinwoo Shin, and Sung-Ju Lee. 2021. Adapting to unknown conditions in learning-based mobile sensing. *IEEE Transactions on Mobile Computing* 21, 10 (2021), 3470–3485.

[13] Taesik Gong, Yewon Kim, Adiba Orzikulova, Yunxin Liu, Sung Ju Hwang, Jinwoo Shin, and Sung-Ju Lee. 2023. DAPPER: Label-Free Performance Estimation after Personalization for Heterogeneous Mobile Sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7, 2 (2023), 1–27.

[14] Taesik Gong, Yeonsu Kim, Jinwoo Shin, and Sung-Ju Lee. 2019. MetaSense: Few-Shot Adaptation to Untrained Conditions in Deep Mobile Sensing. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems* (New York, New York) *(SenSys '19)*. Association for Computing Machinery, New York, NY, USA, 110–123. https://doi.org/10.1145/3356250.3360020

[15] Harish Haresamudram, Apoorva Beedu, Varun Agrawal, Patrick L. Grady, Irfan Essa, Judy Hoffman, and Thomas Plötz. 2020. Masked reconstruction based self-supervision for human activity recognition. In *Proceedings of the 2020 ACM International Symposium on Wearable Computers* (Virtual Event, Mexico) *(ISWC '20)*. Association for Computing Machinery, New York, NY, USA, 45–49. https://doi.org/10.1145/3410531.3414306

[16] Harish Haresamudram, Irfan Essa, and Thomas Plötz. 2021. Contrastive predictive coding for human activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 2 (2021), 1–26.

[17] Harish Haresamudram, Irfan Essa, and Thomas Plötz. 2022. Assessing the state of self-supervised human activity recognition using wearables. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 3 (2022), 1–47.

[18] Harish Haresamudram, Irfan Essa, and Thomas Plötz. 2023. Investigating enhancements to contrastive predictive coding for human activity recognition. In *2023 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, Atlanta, GA, USA, 232–241.

[19] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Seattle, WA, USA, 9726–9735. https://doi.org/10.1109/CVPR42600.2020.00975

[20] Jiahui Hou, Xiang-Yang Li, Peide Zhu, Zefan Wang, Yu Wang, Jianwei Qian, and Panlong Yang. 2019. SignSpeaker: A Real-time, High-Precision SmartWatch-based Sign Language Translator. In *The 25th Annual International Conference on Mobile Computing and Networking* (Los Cabos, Mexico) *(MobiCom '19)*. Association for Computing Machinery, New York, NY, USA, Article 24, 15 pages. https://doi.org/10.1145/3300061.3300117

[21] Qianjiang Hu, Xiao Wang, Wei Hu, and Guo-Jun Qi. 2021. AdCo: Adversarial Contrast for Efficient Learning of Unsupervised Representations from Self-Trained Negative Adversaries. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Nashville, TN, USA, 1074–1083. https://doi.org/10.1109/CVPR46437.2021.00113

[22] Yash Jain, Chi Ian Tang, Chulhong Min, Fahim Kawsar, and Akhil Mathur. 2022. Collossl: Collaborative self-supervised learning for human activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 1 (2022), 1–28.

[23] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. 2020. A survey on contrastive self-supervised learning. *Technologies* 9, 1 (2020), 2.

[24] Longlong Jing and Yingli Tian. 2020. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence* 43, 11 (2020), 4037–4058.

[25] David Josephs, Carson Drake, Andy Heroy, and John Santerre. 2020. sEMG Gesture Recognition with a Simple Model of Attention. In *Proceedings of Machine Learning for Health (ML4H)*. Proceedings of Machine Learning Research (PMLR), Virtual Event, 126–138.

[26] Md Abdullah Al Hafiz Khan, Nirmalya Roy, and Archan Misra. 2018. Scaling Human Activity Recognition via Deep Learning-based Domain Adaptation. In *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, Athens, Greece, 1–9. https://doi.org/10.1109/PERCOM.2018.8444585

[27] Siavash Khodadadeh, Ladislau Bölöni, and Mubarak Shah. 2019. Unsupervised meta-learning for few-shot image classification. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Red Hook, NY, USA, Article 909, 11 pages.

[28] Siavash Khodadadeh, Sharare Zehtabian, Saeed Vahidian, Weijia Wang, Bill Lin, and Ladislau Bölöni. 2021. Unsupervised Meta-Learning through Latent-Space Interpolation in Generative Models. In *Proceedings of the Nineth International Conference on Learning Representations ICLR*. ICLR Conference Track, Virtual Event. https://openreview.net/forum?id=XOjv2HxIF6i

[29] Daehee Kim, Youngjun Yoo, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. 2021. Selfreg: Self-supervised contrastive regularization for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. IEEE, Montreal, QC, Canada, 9619–9628.

[30] Dong Bok Lee, Seanie Lee, Kenji Kawaguchi, Yunji Kim, Jihwan Bang, Jung-Woo Ha, and Sung Ju Hwang. 2023. Self-Supervised Set Representation Learning for Unsupervised Meta-Learning. In *Proceedings of the Eleventh International Conference on Learning Representations (ICLR)*. ICLR Conference Track, Virtual Event. https://openreview.net/forum?id=kIAx30hYi_p

[31] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. 2018. Learning to generalize: meta-learning for domain generalization. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence (AAAI'18/IAAI'18/EAAI'18)*. AAAI Press, New Orleans, Louisiana, USA, Article 427, 8 pages.

[32] Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M Hospedales. 2019. Episodic training for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. IEEE, Seoul, Korea (South), 1446–1455.

[33] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. 2018. Domain Generalization with Adversarial Feature Learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, Salt Lake City, UT, USA, 5400–5409. https://doi.org/10.1109/CVPR.2018.00566

[34] Shuqi Liu, Wei Shao, Tan Li, Weitao Xu, and Linqi Song. 2022. Recent advances in biometrics-based user authentication for wearable devices: A contemporary survey. *Digital Signal Processing* 125 (2022), 103120.

[35] Wang Lu, Yiqiang Chen, Jindong Wang, and Xin Qin. 2021. Cross-domain activity recognition via substructural optimal transport. *Neurocomputing* 454 (2021), 65–75.

[36] Wang Lu, Jindong Wang, Yiqiang Chen, Sinno Jialin Pan, Chunyu Hu, and Xin Qin. 2022. Semantic-discriminative mixup for generalizable sensor-based cross-domain activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 2 (2022), 1–19.

[37] Microsoft. 2020. Termux. https://termux.dev/en/.

[38] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. 2013. Domain generalization via invariant feature representation. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28 (ICML'13)*. JMLR.org, Atlanta, GA, USA, I–10–I–18.

[39] Girish Narayanswamy, Xin Liu, Kumar Ayush, Yuzhe Yang, Xuhai Xu, Shun Liao, Jake Garrison, Shyam Tailor, Jake Sunshine, Yun Liu, et al. 2024. Scaling wearable foundation models.

[40] Alex Nichol and John Schulman. 2018. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999* 2, 3 (2018), 4.

[41] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding. arXiv preprint arXiv:1807.03748. https://arxiv.org/abs/1807.03748

[42] Xiaomin Ouyang, Xian Shuai, Jiayu Zhou, Ivy Wang Shi, Zhiyuan Xie, Guoliang Xing, and Jianwei Huang. 2022. Cosmo: contrastive fusion learning with small data for multimodal human activity recognition. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking* (Sydney, NSW, Australia) *(MobiCom '22)*. Association for Computing Machinery, New York, NY, USA, 324–337. https://doi.org/10.1145/3495243.3560519

[43] HyeonJung Park, Youngki Lee, and JeongGil Ko. 2021. Enabling real-time sign language translation on mobile platforms with on-board depth cameras. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5, 2 (2021), 1–30.

[44] Stefano Pizzolato, Luca Tagliapietra, Matteo Cognolato, Monica Reggiani, Henning Müller, and Manfredo Atzori. 2017. Comparison of six electromyography acquisition setups on hand movement classification tasks. *PloS one* 12, 10 (2017), e0186132.

[45] Hangwei Qian, Sinno Jialin Pan, and Chunyan Miao. 2021. Latent Independent Excitation for Generalizable Sensor-based Cross-Person Activity Recognition. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 13 (May 2021), 11921–11929. https://doi.org/10.1609/aaai.v35i13.17416

[46] Fengchun Qiao, Long Zhao, and Xi Peng. 2020. Learning to Learn Single Domain Generalization. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Seattle, WA, USA, 12553–12562. https://doi.org/10.1109/CVPR42600.2020.01257

[47] Xin Qin, Jindong Wang, Yiqiang Chen, Wang Lu, and Xinlong Jiang. 2022. Domain generalization for activity recognition via adaptive feature fusion. *ACM Transactions on Intelligent Systems and Technology* 14, 1 (2022), 1–21.

[48] Elahe Rahimian, Soheil Zabihi, Amir Asif, Dario Farina, Seyed Farokh Atashzar, and Arash Mohammadi. 2021. FS-HGR: Few-shot learning for hand gesture recognition via electromyography. *IEEE transactions on neural systems and rehabilitation engineering* 29 (2021), 1004–1015.

[49] Mohammad Mahfujur Rahman, Clinton Fookes, Mahsa Baktashmotlagh, and Sridha Sridharan. 2020. Correlation-aware adversarial domain adaptation and generalization. *Pattern Recognition* 100 (2020), 107124.

[50] Attila Reiss and Didier Stricker. 2012. Introducing a New Benchmarked Dataset for Activity Monitoring. In *2012 16th International Symposium on Wearable Computers*. IEEE, Newcastle, UK, 108–109. https://doi.org/10.1109/ISWC.2012.13

[51] Daniel Roggen, Alberto Calatroni, Mirco Rossi, Thomas Holleczek, Kilian Förster, Gerhard Tröster, Paul Lukowicz, David Bannach, Gerald Pirkl, Alois Ferscha, Jakob Doppler, Clemens Holzmann, Marc Kurz, Gerald Holl, Ricardo Chavarriaga, Hesam Sagha, Hamidreza Bayati, Marco Creatura, and José del R. Millàn. 2010. Collecting complex activity datasets in highly rich networked sensor environments. In *2010 Seventh International Conference on Networked Sensing Systems (INSS)*. IEEE, Kassel, Germany, 233–240. https://doi.org/10.1109/INSS.2010.5573462

[52] Aaqib Saeed, Tanir Ozcelebi, and Johan Lukkien. 2019. Multi-task self-supervised learning for human activity detection. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 2 (2019), 1–30.

[53] Philip Schmidt, Attila Reiss, Robert Duerichen, Claus Marberger, and Kristof Van Laerhoven. 2018. Introducing WESAD, a Multimodal Dataset for Wearable Stress and Affect Detection. In *Proceedings of the 20th ACM International Conference on Multimodal Interaction* (Boulder, CO, USA) *(ICMI '18)*. Association for Computing Machinery, New York, NY, USA, 400–408. https://doi.org/10.1145/3242969.3242985

[54] Yongjie Shi, Xianghua Ying, and Jinfa Yang. 2022. Deep unsupervised domain adaptation with time series sensor data: A survey. *Sensors* 22, 15 (2022), 5507.

[55] Xingzhe Song, Boyuan Yang, Ge Yang, Ruirong Chen, Erick Forno, Wei Chen, and Wei Gao. 2020. SpiroSonic: monitoring human lung function via acoustic sensing on commodity smartphones. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking* (London, United Kingdom) *(MobiCom '20)*. Association for Computing Machinery, New York, NY, USA, Article 52, 14 pages. https://doi.org/10.1145/3372224.3419209

[56] Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. 2015. Smart Devices are Different: Assessing and MitigatingMobile Sensing Heterogeneities for Activity Recognition. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems* (Seoul, South Korea) *(SenSys '15)*. Association for Computing Machinery, New York, NY, USA, 127–140. https://doi.org/10.1145/2809695.2809718

[57] Chi Ian Tang, Ignacio Perez-Pozuelo, Dimitris Spathis, and Cecilia Mascolo. 2020. Exploring Contrastive Learning in Human Activity Recognition for Healthcare. https://arxiv.org/abs/2011.11542 Presented at Machine Learning for Mobile Health Workshop at NeurIPS 2020, Vancouver, Canada.

[58] Yunus Emre Ustev, Ozlem Durmaz Incel, and Cem Ersoy. 2013. User, device and orientation independent human activity recognition on mobile phones: challenges and a proposal. In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication* (Zurich, Switzerland) *(UbiComp '13 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 1427–1436. https://doi.org/10.1145/2494091.2496039

[59] Yunus Emre Ustev, Ozlem Durmaz Incel, and Cem Ersoy. 2013. User, device and orientation independent human activity recognition on mobile phones: challenges and a proposal. In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication* (Zurich, Switzerland) *(UbiComp '13 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 1427–1436. https://doi.org/10.1145/2494091.2496039

[60] Jindong Wang, Yiqiang Chen, Lisha Hu, Xiaohui Peng, and Philip S. Yu. 2018. Stratified Transfer Learning for Cross-domain Activity Recognition. In *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, Athens, Greece, 1–10. https://doi.org/10.1109/PERCOM.2018.8444572

[61] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip S. Yu. 2023. Generalizing to Unseen Domains: A Survey on Domain Generalization. *IEEE Transactions on Knowledge and Data Engineering* 35, 8 (2023), 8052–8072. https://doi.org/10.1109/TKDE.2022.3178128

[62] Jinqiang Wang, Tao Zhu, Jingyuan Gan, Liming Luke Chen, Huansheng Ning, and Yaping Wan. 2022. Sensor data augmentation by resampling in contrastive learning for human activity recognition. *IEEE Sensors Journal* 22, 23 (2022), 22994–23008.

[63] Yufei Wang, Haoliang Li, and Alex C. Kot. 2020. Heterogeneous Domain Generalization Via Domain Mixup. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Barcelona, Spain, 3622–3626. https://doi.org/10.1109/ICASSP40776.2020.9053273

[64] Zi Wang, Sheng Tan, Linghan Zhang, Yili Ren, Zhi Wang, and Jie Yang. 2021. EarDynamic: An Ear Canal Deformation Based Continuous User Authentication Using In-Ear Wearables. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 5, 1, Article 39 (March 2021), 27 pages. https://doi.org/10.1145/3448098

[65] Garrett Wilson and Diane J Cook. 2020. A survey of unsupervised deep domain adaptation. *ACM Transactions on Intelligent Systems and Technology (TIST)* 11, 5 (2020), 1–46.

[66] Huatao Xu, Pengfei Zhou, Rui Tan, Mo Li, and Guobin Shen. 2021. LIMU-BERT: Unleashing the Potential of Unlabeled Data for IMU Sensing Applications. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems* (Coimbra, Portugal) *(SenSys '21)*. Association for Computing Machinery, New York, NY, USA, 220–233. https://doi.org/10.1145/3485730.3485937

[67] Xufeng Yao, Yang Bai, Xinyun Zhang, Yuechen Zhang, Qi Sun, Ran Chen, Ruiyu Li, and Bei Yu. 2022. Pcl: Proxy-based contrastive learning for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, New Orleans, LA, USA, 7097–7107.

[68] Hang Yuan, Shing Chan, Andrew P Creagh, Catherine Tong, Aidan Acquah, David A Clifton, and Aiden Doherty. 2024. Self-supervised learning for human activity recognition using 700,000 person-days of wearable data. *NPJ digital medicine* 7, 1 (2024), 91.

[69] Aras Yurtman and Billur Barshan. 2017. Activity Recognition Invariant to Sensor Orientation with Wearable Motion Sensors. *Sensors* 17, 8 (2017). https://doi.org/10.3390/s17081838

[70] Hanbin Zhang, Chenhan Xu, Huining Li, Aditya Singh Rathore, Chen Song, Zhisheng Yan, Dongmei Li, Feng Lin, Kun Wang, and Wenyao Xu. 2019. Pdmove: Towards passive medication adherence monitoring of parkinson's disease using smartphone-based gait assessment. *Proceedings of the ACM on interactive, mobile, wearable and ubiquitous technologies* 3, 3 (2019), 1–23.

[71] Xingxuan Zhang, Linjun Zhou, Renzhe Xu, Peng Cui, Zheyan Shen, and Haoxin Liu. 2022. Towards unsupervised domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, New Orleans, LA, USA, 4910–4920.

[72] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. 2021. Domain Generalization with MixStyle. In *Proceedings of the International Conference on Learning Representations (ICLR)*. ICLR Conference Track, Virtual Eent, 1–12. https://openreview.net/forum?id=6xHJ37MVxxp

[73] Zhijun Zhou, Yingtian Zhang, Xiaojing Yu, Panlong Yang, Xiang-Yang Li, Jing Zhao, and Hao Zhou. 2020. XHAR: Deep Domain Adaptation for Human Activity Recognition with Smart Devices. In *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, Como, Italy, 1–9. https://doi.org/10.1109/SECON48991.2020.9158431