CoDEm: *Co*nditional *D*omain *Em*beddings for Scalable Human Activity Recognition

Abu Zaher Md Faridee*, Avijoy Chakma*, Zahid Hasan*, Nirmalya Roy*, Archan Misra**
*Information Systems, University of Maryland, Baltimore County
**Computing & Information Systems, Singapore Management University
Email: *{faridee1, achakma1, zhasan3, nroy}@umbc.edu, **archanm@smu.edu.sg

2022 IEEE International Conference on Smart Computing (SMARTCOMP) | 978-1-6654-8152-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/SMARTCOMP55677.2022.00017

***Abstract*—We explore the effect of auxiliary labels in improving the classification accuracy of wearable sensor-based human activity recognition (HAR) systems, which are primarily trained with the supervision of the activity labels (e.g. running, walking, jumping). Supplemental meta-data are often available during the data collection process such as body positions of the wearable sensors, subjects' demographic information (e.g. gender, age), and the type of wearable used (e.g. smartphone, smart-watch). This information, while not directly related to the activity classification task, can nonetheless provide auxiliary supervision and has the potential to significantly improve the HAR accuracy by providing extra guidance on how to handle the introduced sample heterogeneity from the change in domains (i.e positions, persons, or sensors), especially in the presence of limited activity labels. However, integrating such meta-data information in the classification pipeline is non-trivial – (i) the complex interaction between the activity and domain label space is hard to capture with a simple multi-task and/or adversarial learning setup, (ii) meta-data and activity labels might not be simultaneously available for all collected samples. To address these issues, we propose a novel framework *Co*nditional *D*omain *Em*beddings (*CoDEm*). From the available unlabeled raw samples and their domain meta-data, we first learn a set of domain embeddings using a contrastive learning methodology to handle inter-domain variability and inter-domain similarity. To classify the activities, *CoDEm* then learns the label embeddings in a contrastive fashion, conditioned on domain embeddings with a novel attention mechanism, enforcing the model to learn the complex domain-activity relationships. We extensively evaluate *CoDEm* in three benchmark datasets against a number of multi-task and adversarial learning baselines and achieve state-of-the-art performance in each avenue.**

***Index Terms*—human activity recognition, domain embedding, attention, multi-task learning, adversarial learning, meta-data**

I. Introduction

Wearable sensor-based human activity recognition (HAR) systems have enjoyed meteoric popularity in the past few years in a number of application areas such as health-care, depression prediction [1], sleep [2] and fitness monitoring [3], cognitive [4] and mental health assessment [5] due to their widespread availability (in the form of smartwatches, smartphones, etc), flexible and unobtrusive nature, and the ability capture and infer users activity in real-time. The recent advancement in data-driven supervised deep learning algorithms have significantly lessened the need for training HAR models with complex signal processing and hand-crafted feature engineering pipelines. The performance of these supervised models is, however, highly dependent on the amount and quality of the labels provided which makes developing scalable HAR models a challenging task. Compared to the traditional use cases of deep learning (e.g. NLP and computer vision), human motion data is much more nuanced in nature. Due to the wider variation introduced by the modality (sensor type), placement (on-body or contact-less), and the personal and temporal variability (resulting from the users' lifestyle and contextual differences), both the marginal and conditional distributions of the data can experience a significant shift. Traditional methods of building a supervised classifier only focus on learning the mappings between the raw samples and the activity labels. However, the meta-data information (e.g. position, device, or user demographic) is often not utilized in building these classifiers. We argue that these auxiliary labels carry significant supervision information and when utilized in a regulated manner, can considerably improve the performance of the activity classifiers, especially when limited activity labels are present. More importantly, these meta-data are often easily available during the data collection process, hence enabling one to independently assign these auxiliary labels to the raw data samples irrespective of the availability of the corresponding activity labels.

The straightforward way of integrating these auxiliary supervisions is to build a multi-task model that simultaneously predicts the activity labels and the meta-labels (e.g. position, gender, device), assuming a globally *cooperative* relationship. However, as we will see later in Section VI, this assumption is not completely valid. One might also assume that learning a set of features that are invariant to these heterogeneities (e.g. with adversarial multi-task learning, assuming a globally *adversarial* relationship) will result in better performance, but in reality that also results in performance degradation. This leads us to believe that there exists a more complex, local (per sample) relationship between the primary (from activity label) and auxiliary (i.e. from domain/meta-data label) feature space and special care must be taken to learn that relationship to improve the performance of the activity classifiers. These multi-task learning setups often require careful tuning of the weights of the individual task losses so that the auxiliary tasks do not overwhelm the primary one, which requires either (a) a time-consuming manual hyper-parameter search regiment or (b) incorporating complex loss balancing algorithms [6]–[8], especially when multiple auxiliary tasks need balancing against each other in addition to the primary task. Moreover,

2693-8340/22/$31.00 ©2022 IEEE
DOI 10.1109/SMARTCOMP55677.2022.00017

these models assume that both primary and auxiliary labels are available for each training sample, thus they are unable to exploit any easily available unlabeled samples that only contain auxiliary (domain/meta) labels.

To address these challenges, we propose *Co*nditional *D*omain *Em*beddings (*CoDEm*). From the unlabeled samples and their domain meta-data, *CoDEm* first learns a set of domain-specific low dimensional embeddings. It is trained with a contrastive learning methodology so that samples originating from similar domains (e.g. same body position, person, or device) reside in the same neighborhood in the learned embedding space while at the same time samples from dissimilar domains are pushed apart. This effectively enables us to learn unique *encoding* for each meta-label from the raw data samples such that the knowledge from these encodings can later be used as input into another model, removing the need to have meta-labels for the labeled samples that will be used to train the activity classifier. To train the classifier from the activity labels, we again employ a contrastive learning methodology during feature extraction to induce clustering behavior in the low dimensional learned space, so that activities become easily separable by the classifier. We then introduce a novel attention mechanism to learn the complex interaction between the pre-trained domain feature space (learned from the meta-labels) and activity label feature space – allowing the network to choose per sample basis how much domain information to retain (and discard) to improve the task label classification – removing the need for complicated loss balancing or choosing between complementary/adversarial relationship between the primary and auxiliary tasks. By efficiently utilizing meta-data label information and conditioning the activity classifier on it, *CoDEm* significantly improves HAR classification accuracy beyond single/multi-task baselines.

Key Contributions: We make the following key contributions:

- **Ability to Utilize Meta-data (Domain) Information from Unlabeled Data to Improve the HAR Classification Accuracy:** We motivate and propose *CoDEm*, a novel HAR classification approach that exploits the often overlooked meta-data information from the unlabeled data to learn unique representations of each meta-data label (e.g. device position, type, user demographics such as gender, hand-dominance) and their associated domain heterogeneities. We propose a methodology to learn low dimensional embedding of the domains – samples of each distinct domain coalesce into their unique neighborhood resulting in unique encoding for each domain/meta-data label. We utilize these pre-trained encoding as a source of auxiliary supervision when training activity classifiers with limited labeled samples and achieve improved accuracy with the same number of labels samples over a normal supervised classification pipeline.
- **Automatic Discovery and Exploitation of the Relationship Between Domain (meta-data) and Label Space with a Hyper-parameter Free Attention Mechanism**: We propose a novel attention mechanism that automatically *learns* the complex relationship between the meta-data/domain embeddings and the activity label embedding, selectively choosing only the relevant features from the domain embeddings per sample basis to improve the activity label classification performance. Instead of having a classical global co-operative/adversarial multi-task learning setup and choosing a loss balancing term with a long hyper-parameter search (especially when the number of tasks becomes more than two), our model *learns* the **type** and **strength** of this relationship per sample basis. Since encodings of the meta-data labels can be learned separately from the activity labels, the *decoupling* allows the classifier to be trained without any meta-data information on its training samples, in stark contrast to traditional multi-task learning setups.
- **Demonstration of *CoDEm's* Efficacy and Robustness:** We demonstrate the efficacy of *CoDEm* in improving the activity classification performance by utilizing meta-data information from unlabeled samples on 3 distinct benchmark datasets (SHAR [9], PAMAP2 [10], and OPPORTUNITY [11]). The 3 datasets capture a range of low-level human activities/gestures (e.g. sitting, standing, lying), more complex short-lived transient activities (e.g. jumping, ascending, descending stairs), and typical ADLs (Activities of Daily Living), and are characterized by sample heterogeneities across users with different body positions. We experimentally establish that *CoDEm* outperforms normal supervised classification baseline (which does not utilize any meta-data information) by $\approx 11\%$ in macro-F1 score. We also show that, when meta-data information is available, *CoDEm* also outperforms traditional multi-task co-operative and adversarial learning setups by $\approx 9.5\%$ in terms of macro-F1 scores. We also visualize and measure the intra-class *cohesion* and iter-class *separation* of the embedding space learned by *CoDEm* – which shows that *CoDEm* generates feature spaces with up to 17.51% higher silhouette score across the datasets, justifying *CoDEm's* improvement over the baselines.

II. Overview

Figure 1 depicts a high-level overview of our proposed *CoDEm* architecture. Our vision for *CoDEm* is to support robust HAR classification with a limited number of training samples by utilizing the meta-data information from unlabeled samples. Unlike the classical multi-task learning approach that provides auxiliary supervision from such-meta data information but requires the activity labels and meta-labels are associated with the same set of data samples, *CoDEm* can work with a completely disjoint set of samples that contain either the meta-data information or the activity label information – thereby making it possible to exploit a larger set of unlabeled of samples to improve model's performance.

In order to achieve this feat, *CoDEm* first trains a set of feature extractors and projects the features into low dimensional embeddings. It trains these *Domain Embeddings* separately with each set of domain labels (i.e. *position*, *device*, and *genders*) with the unlabeled samples, after which all the

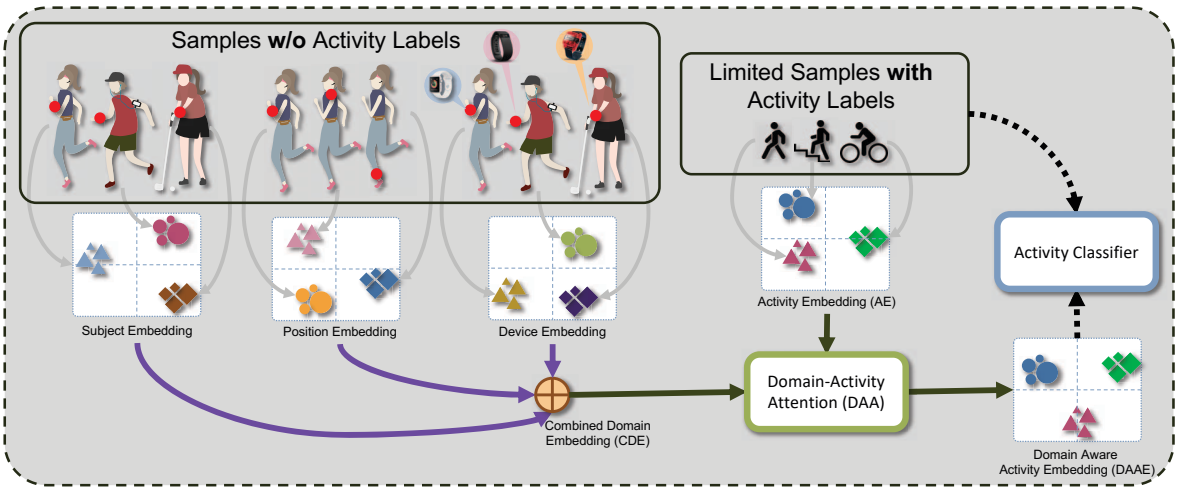


Fig. 1: Overview of *CoDEm* Framework

resulting embeddings are concatenated, resulting in the *Combined Domain Embeddings*. Similarly, with the limited activity labeled samples, *CoDEm* also independently extracts a set of features – the projected *Activity Embedding* vectors, in this case, are assumed to be sub-optimal for downstream classification task due to (i) being trained with limited activity labeled samples, and (ii) not containing any supervision on how to handle the potential heterogeneity (e.g. difference in position, gender or device) in the input samples. Hence, the main challenge becomes finding the optimal way to integrate only the salient features from the *Combined Domain Embeddings* into the *Activity Embedding*. *CoDEm* introduces a *Domain-Activity Attention* block to automatically learn the optimal interaction between these two embeddings space through back-propagation, resulting in a *Domain Aware Activity Embedding* space where the activity labels are optimally distributed. Final classification results are obtained by training a *Classifier* on top of the *Domain Aware Activity Embedding* with the same set of limited activity samples.

One key ingredient of *CoDEm*'s success is the utilization of contrastive learning [12], [13] to generate the *Activity* and *Domain Embeddings* and induce clustering characteristics in the learned unit normalized embedding spaces. This design choice has multiple benefits. First, this enables a geometric interpretation of the embedding space. The *Domain Aware Activity Embedding* is essentially a corrected re-projection of the *Activity Embedding* space and the *Domain-Activity Attention* block provides the necessary parameters for this re-projection learned from the *Domain Embedding*. As each of the embeddings resides in euclidean space, the *Domain-Activity Attention* block can be modeled with a simple attention mechanism (later detailed in Section IV-B3) greatly reducing the extra number of parameters to be learned. The clustering behavior in the embeddings also makes both the domain and activity labels easily separable with a simple classifier.

We would like to note that *CoDEm* is not intended as a replacement for other frameworks that aim to leverage unlabeled data [14]–[16] to improve the HAR classification performance. However, these frameworks never intentionally take advantage of the meta-data information available in the unlabeled data samples, and *CoDEm* is specially designed to address that gap. Hence, we anticipate *CoDEm* to be compatible with the majority of off-the-shelf HAR classification frameworks.

III. Background

Before delving deeper into the details of each component of *CoDEm*, we provide a quick review of the primary mechanism to learn a compact representation of the domains and activity labels. In *CoDEm*, we primarily employ *Contrastive Center Loss* that aims to learn representations with large inter-class separability and minimal intra-class variability.

Wen et al. [17] first proposed an auxiliary loss function to supervised Softmax loss called *center-loss* that introduces a prototypical class center for each of the classes and adds a criterion that penalizes the distance of each classes representation from their respective class centers, which can be described by the following equation.

$$\mathcal{L}^c = \frac{1}{2}\sum_{i=1}^{m} \|z_i - c_{y_i}\|_2^2 \tag{1}$$

Here $z_i \in \mathbb{R}^d$ denotes the learned deep representation for input sample x_i and $c_{y_i} \in \mathbb{R}^d$ denotes the y_i-th activity class center. d denotes the feature dimension and m denotes the number of training samples in a batch.

While the center loss function described in Equation 1 penalizes large intra-class distances, it does not consider inter-class separability. In the resulting embedding space, the class centers might still end up close by to each other – hence reducing the discriminative power of the classifier. To address this issue, Qi et al. [13] proposed an extension to it called *contrastive center-loss* which additionally penalizes the closeness of the class centers.

$$\mathcal{L}^{ct-c} = \frac{1}{2}\sum_{i=1}^{m}\frac{\|z_i - c_{y_i}\|_2^2}{(\sum_{j=1,j\neq y_i}^{k}\|x_i - c_j\|_2^2) + \delta} \tag{2}$$

Here $\mathcal{L}^{ct-c}$ denotes the contrastive center loss, k denotes the number of classes, and $\delta = 1$ as default. The modification in the denominator of Equation 2 allows simultaneous minimization of intra-class variability and maximization of inter-class separability. By optimizing the ratio of intra-class separation and inter-class separation (contrastive part), the network ensures a higher relative distance between different class prototypes.

IV. Methodology

A. Preliminary Assumptions

As stated previously, instead of jointly learning a common feature embedding from the domain and activity labels, *CoDEm* opts for a decoupled architecture where the domain and activity-specific features are learned separately from unlabeled and labeled sets respectively. A labeled training set, $\mathbf{L} = \{(x_l^{(i)}, y_l^{(i)})\}_{i=1}^{N}$ consist of N number of training samples $x_l^{(i)}$ and their associated activity labels $y_l^{(i)}$. A traditional supervised approach will model $\mathbb{E}(y_l^{(i)})$ as $P(y_l^{(i)}|x_l^{(i)})$. Let us assume that an *encoding* of the associated subject label $\hat{s}_l^{(i)}$, device label $\hat{d}_l^{(i)}$ and position label $\hat{p}_l^{(i)}$ can be inferred. *CoDEm* postulates that modeling $\mathbb{E}(y_l^{(i)})$ as $P(y_l^{(i)}|x_l^{(i)}, \hat{s}_l^{(i)}, \hat{d}_l^{(i)}, \hat{p}_l^{(i)})$ leads to increased performance, especially when N is small. Since, the labeled set $\mathbf{L}$ does not provide these *encodings* of the domain labels $\hat{s}_l^{(i)}, \hat{d}_l^{(i)}, \hat{p}_l^{(i)}$, we take advantage of the meta-data labels from the an unlabeled training set, $\mathbf{U} = \{(x_u^{(i)}, s_u^{(i)}, d_u^{(i)}, p_u^{(i)})\}_{i=1}^{M}$ to model $\mathbb{E}(s_u^{(i)})$, $\mathbb{E}(d_u^{(i)})$, $\mathbb{E}(p_u^{(i)})$ which can be used later to infer $\hat{s}_l^{(i)}, \hat{d}_l^{(i)}, \hat{p}_l^{(i)}$ given $x_l^{(i)}$. $\mathbf{U}$ consists of M training samples and is fully disjoint from $\mathbf{L}$. We also assume that $N \ll M$.

B. Functional Components

Next we provide an breakdown different components of our proposed *CoDEm* framework including the **Activity Embedding** ($\mathsf{E}_{\mathtt{act}}$), **Domain Embedding** (E_{D}), the **Domain Activity Attention** ($\mathsf{A}_{\mathsf{D}-\mathtt{act}}$) between them and the **Activity Classifier** (C), and explain how these components are utilized in both the learning and inference stages of *CoDEm* (as illustrated in Fig. 1).

1) Activity Embedding, $\mathsf{E}_{\mathtt{act}}$*:* Given the labeled training set $\mathbf{L}$, Activity Embedding $\mathsf{E}_{\mathtt{act}}$ is a learnable parameterized projection function [18] which maps each input sample $x_l^{(i)} \in \mathcal{X}_l$ to a vector

$$z_{\mathtt{act}(l)}^{(i)} = \mathsf{E}_{\mathtt{act}}(x_l^{(i)}) \in \mathbb{R}^n \tag{3}$$

We optimize the parameters of $\mathsf{E}_{\mathtt{act}}$ with $\mathcal{L}^{ct-c}(x_l^{(i)}, y_l^{(i)})$ where the objective is to *minimize* intra-class distance and *maximize* inter-class distance (as shown in Figure 2).

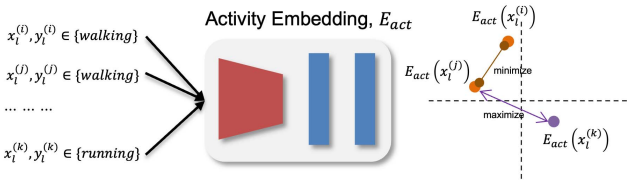


Fig. 2: Activity Embedding, $\mathsf{E}_{\mathtt{act}}$

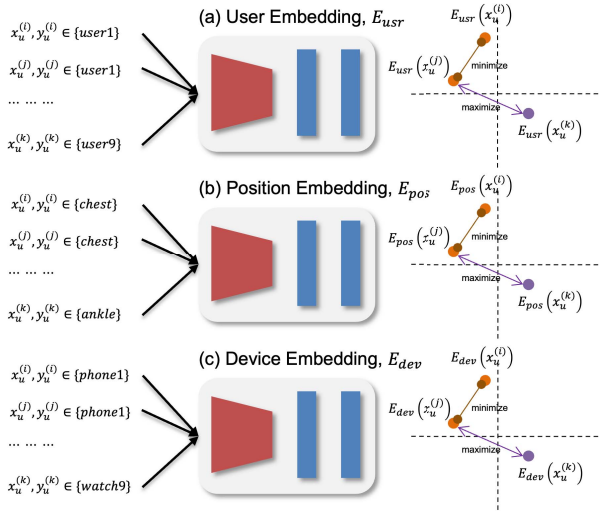


Fig. 3: Learning (a) User, $\mathsf{E}_{\mathtt{usr}}$, (b) Position $\mathsf{E}_{\mathtt{pos}}$ and (c) $\mathsf{E}_{\mathtt{dev}}$ Embeddings Independently

2) Domain Embedding, E_{D}*:* From the unlabeled training set $\mathbf{U}$, we fist learn the following embeddings independently (i) **User Embedding**, $\mathsf{E}_{\mathtt{usr}}$, (ii) **Position Embeddings** $\mathsf{E}_{\mathtt{pos}}$, (iii) **Device Embedding**, $\mathsf{E}_{\mathtt{dev}}$. Here $\mathsf{E}_{\mathtt{usr}}$, $\mathsf{E}_{\mathtt{pos}}$ and $\mathsf{E}_{\mathtt{dev}}$ are three learnable parameterized projection functions [18] which maps each input sample $x_u^{(i)} \in \mathcal{X}_u$ to three vectors

$$z_{\mathtt{usr}(u)}^{(i)} = \mathsf{E}_{\mathtt{usr}}(x_u^{(i)}) \in \mathbb{R}^p \tag{4}$$

$$z_{\mathtt{pos}(u)}^{(i)} = \mathsf{E}_{\mathtt{pos}}(x_u^{(i)}) \in \mathbb{R}^q \tag{5}$$

$$z_{\mathtt{dev}(u)}^{(i)} = \mathsf{E}_{\mathtt{dev}}(x_u^{(i)}) \in \mathbb{R}^r \tag{6}$$

All of these embeddings are trained in the same fashion as Activity Embedding (Section IV-B1), but with their respective domain labels. We then concatenate the three vectors to derive the final domain encoding vector $z_{\mathsf{D}(u)}^{(i)}$.

$$z_{\mathsf{D}(u)}^{(i)} = z_{\mathtt{usr}(u)}^{(i)} \oplus z_{\mathtt{pos}(u)}^{(i)} \oplus z_{\mathtt{dev}(u)}^{(i)} \tag{7}$$

For simplicity of exposition, we interpret $z_{\mathsf{D}(u)}^{(i)}$ as outcome of applying a Combined **Domain Embedding** function $\mathsf{E}_{\mathsf{D}} \in \mathbb{R}^m$ (where $m = p + q + r$) so that the following holds,

$$\begin{aligned}\mathsf{E}_{\mathsf{D}}(x_u^{(i)}) &= z_{\mathsf{D}(u)}^{(i)} \\ &= \mathsf{E}_{\mathtt{usr}}(x_u^{(i)}) \oplus \mathsf{E}_{\mathtt{pos}}(x_u^{(i)}) \oplus \mathsf{E}_{\mathtt{dev}}(x_u^{(i)})\end{aligned} \tag{8}$$

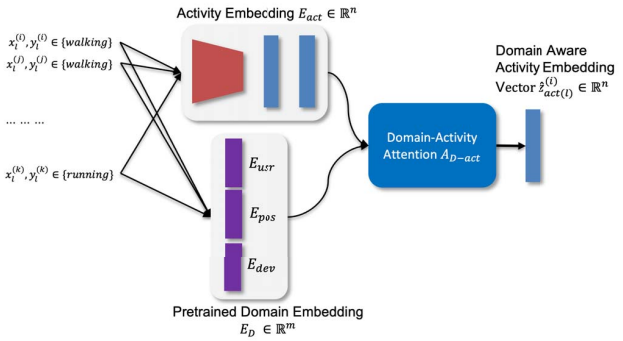


Fig. 4: Conditional label embedding

3) Domain Activity Attention, $\mathsf{A}_{\mathtt{D-act}}$*:* With $\mathsf{E}_{\mathtt{act}}$ trained on a labeled training set $\mathbf{L}$ and $\mathsf{E}_{\mathtt{D}}$ trained on unlabeled training set $\mathbf{U}$, the challenge now lies in finding a way to incorporate domain heterogeneity information from $\mathsf{E}_{\mathtt{D}}$ into $\mathsf{E}_{\mathtt{act}}$ to create a *Domain Aware Activity Embedding Vector* $\hat{z}^{(i)}_{\mathtt{act}(l)}$. To that end, the labeled samples in $\mathbf{L}$ are passed through $\mathsf{E}_{\mathtt{D}}$ to get the corresponding domain encoding vector $z^{(i)}_{\mathtt{D}(l)}$.

$$z^{(i)}_{\mathtt{D}(l)} = \mathsf{E}_{\mathtt{usr}}(x^{(i)}_l) \oplus \mathsf{E}_{\mathtt{pos}}(x^{(i)}_l) \oplus \mathsf{E}_{\mathtt{dev}}(x^{(i)}_l) \tag{9}$$

The *Domain Activity Attention* block $\mathsf{A}_{\mathtt{D-act}}$ takes both $z^{(i)}_{\mathtt{act}(l)}$ and $z^{(i)}_{\mathtt{D}(l)}$ as input and produces our *Domain Aware Activity Embedding Vector* $\hat{z}^{(i)}_{\mathtt{act}(l)}$. Since our intention is to incorporate small domain-specific corrections from $z^{(i)}_{\mathtt{D}(l)}$ into $\hat{z}^{(i)}_{\mathtt{act}(l)}$, we can first learn a projection of $z^{(i)}_{\mathtt{D}(l)}$ into the same dimension as $z^{(i)}_{\mathtt{act}(l)} \in \mathbb{R}^n$ and then take the dot product between the two vectors to derive the *attention score* [19].

$$s^{(i)}_{\mathtt{act}(l)} = z^{(i)}_{\mathtt{act}(l)} \cdot z^{(i)}_{\mathtt{D}(l)} \times w \tag{10}$$

This score can then be used to provide necessary correction to $z^{(i)}_{\mathtt{act}(l)}$ to get the final $\hat{z}^{(i)}_{\mathtt{act}(l)}$. Instead of fully overwriting $z^{(i)}_{\mathtt{act}(l)}$ with the updated values, we employ a residual connection as it allows the network to pick and choose which features to update.

$$\hat{z}^{(i)}_{\mathtt{act}(l)} = z^{(i)}_{\mathtt{act}(l)} + z^{(i)}_{\mathtt{act}(l)} s^{(i)}_{\mathtt{act}(l)} \tag{11}$$

Here $w \in \mathbb{R}^{m \times n}$ is a learnable learnable parameters. Hence $\mathsf{A}_{\mathtt{D-act}}$ can be rewritten as following,

$$\mathsf{A}_{\mathtt{D-act}}(z^{(i)}_{\mathtt{act}(l)}, z^{(i)}_{\mathtt{D}(l)}) = z^{(i)}_{\mathtt{act}(l)} + z^{(i)}_{\mathtt{act}(l)}(z^{(i)}_{\mathtt{act}(l)} \cdot z^{(i)}_{\mathtt{D}(l)} \times w) \tag{12}$$

4) Activity Classifier, C*:* Once we get the *Domain Aware Activity Embedding Vector*, the next task is to apply a classification head with a `Softmax` activation.

$$\hat{y}^{(i)}_{act(l)} = \mathsf{C}(\hat{z}^{(i)}_{\mathtt{act}(l)}) \tag{13}$$

The full training procedure is summarized in Algorithm 1

Algorithm 1: Learning Algorithm for *CoDEm*

Input: Labeled Dataset, $\mathbf{L} = \{x^{(i)}_l, y^{(i)}_l\}_{i=1}^N, N = |\mathbf{L}|$
Input: Unlabeled Dataset, $\mathbf{U} = \{x^{(i)}_u, s^{(i)}_u, d^{(i)}_u, p^{(i)}_u\}_{i=1}^M, M = |\mathbf{U}|$
Assumption: $N \ll M$
Output: Activity Embedding, $\mathsf{E}_{\mathtt{act}}$
Output: Domain Embedding, $\mathsf{E}_{\mathtt{D}}$
Output: Domain Activity Attention, $\mathsf{A}_{\mathtt{D-act}}$
Output: Activity Classifier, C
for $x^{(i)}_u \in \mathcal{X}_u$ **do**
1 Train $\mathsf{E}_{\mathtt{usr}}, \mathsf{E}_{\mathtt{pos}}, \mathsf{E}_{\mathtt{dev}}$ using Equation 2 so that
$$z^{(i)}_{\mathtt{D}(u)} = \mathsf{E}_{\mathtt{D}}(x^{(i)}_u) = \mathsf{E}_{\mathtt{usr}}(x^{(i)}_u) \oplus \mathsf{E}_{\mathtt{pos}}(x^{(i)}_u) \oplus \mathsf{E}_{\mathtt{dev}}(x^{(i)}_u)$$
for $x^{(i)}_l \in \mathcal{X}_l$ **do**
2 Train $\mathsf{E}_{\mathtt{act}}$ using Equation 2 so that
$$z^{(i)}_{\mathtt{act}(l)} = \mathsf{E}_{\mathtt{act}}(x^{(i)}_l) \in \mathbb{R}^n$$
3 Infer $z^{(i)}_{\mathtt{D}(l)} = \mathsf{E}_{\mathtt{usr}}(x^{(i)}_l) \oplus \mathsf{E}_{\mathtt{pos}}(x^{(i)}_l) \oplus \mathsf{E}_{\mathtt{dev}}(x^{(i)}_l)$
4 Train $\mathsf{A}_{\mathtt{D-act}}$ using Equation 12 so that
$$\hat{z}^{(i)}_{\mathtt{act}(l)} = \mathsf{A}_{\mathtt{D-act}}(z^{(i)}_{\mathtt{act}(l)}, z^{(i)}_{\mathtt{D}(l)})$$
5 Train C so that $\hat{y}^{(i)}_{act(l)} = \mathsf{C}(\hat{z}^{(i)}_{\mathtt{act}(l)})$

V. Experiment

In the following section, we discuss the details of a number of representative Activities-of-Daily-Living (ADL) datasets, which we use to demonstrate the efficacy of our proposed *CoDEm* approach. We also summarize alternative approaches that help provide competitive baselines.

A. Baselines

To demonstrate the effectiveness of *CoDEm*, we compare its performance with 3 different baselines: (i) Single Task Learning/Classification (STL), (ii) Co-operative Multi-task Learning (CoOp-MTL), (iii) Adversarial Multi-task Learning [8] (Adv-MTL). By STL, we refer to training the model with the activity labels only. During the CoOp-MTL setup, a single feature extractor is shared with two classification heads – one for activity label classification and the other for meta-label classification. This setup is then modified with a gradient reversal layer [8] to create the Adv-MTL baseline. These baselines serve to demonstrate different ways the auxiliary labeling information can be integrated into the activity classification pipeline.

Table I shows the primary difference between the baselines and *CoDEm*. As we can see, *CoDEm* is the only framework that can work with a fully disjoint set of labeled (activity) and unlabeled (with meta-label) training sets. In addition, *CoDEm* is the only framework that does not require an additional hyper-parameter to balance the primary (activity classification) and auxiliary (domain classification) losses which makes it

possible to gain optimal classification results without a time-consuming hyper-parameter search.

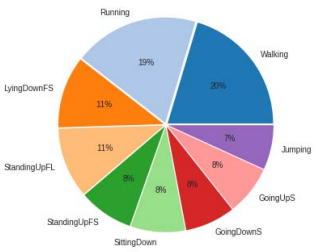
As previously stated, we are mainly interested in exploring the effect of auxiliary supervision on activity classification in this work, hence our benchmark selection purposefully does not include models that use purely unlabeled samples [14]–[16], [20]. *CoDEm's* novel contributions (*contrastive domain* and *activity embedding* and *residual attention* between them) are generalizable enough to be compatible with the above-mentioned frameworks but the study of that behavior is beyond the scope of this paper.

TABLE I: Feature Matrix Comparison of the Baselines

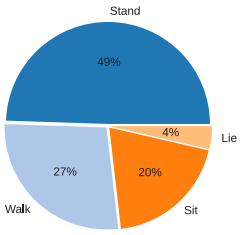
Model	Single-Task Learning	Multi-Task Learning	Adversarial Multi-Task Learning	*CoDEm*
Works with fully disjoint labeled and unlabeled dataset	N/A	No	No	Yes
Does not need loss-balancing hyper-parameter tuning	N/A	No	No	Yes

B. Datasets

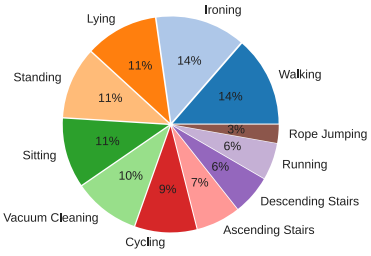
We showcase the effectiveness of our proposed framework on three publicly available datasets: (i) PAMAP2 Physical Activity Monitoring Dataset [10], (ii) OPPORTUNITY Activity Recognition Dataset [11], and (iii) UniMiB SHAR [9]. A summary of the datasets is provided in Table II and the label distributions (full dataset) are shown in Figure 5. Pre-processing details for the datasets are provided in Section V-C.



(a) SHAR



(b) OPPORTUNITY



(c) PAMAP2

Fig. 5: Label distribution of the three benchmark datasets

C. Implementation Details

1) Runtime Environment: We conducted our experiments on a Linux Server (Ubuntu 18.04) running on Intel Core i7-6850K CPU and 64GB DDR4 RAM, with an Nvidia 1080Ti Graphics card (11GB VRAM). `Python` was used for all coding tasks except. For the signal processing, filtering, we used `scikit − learn`, `scipy` and `numpy` libraries. For deep learning tasks, we used `PyTorch`.

2) Pre-processing: For each dataset, we first take the data stream of each trial through a multi-variate iterative data imputer [21] to deal with missing values. We isolated the gravity component from the accelerometer signals with a Butterworth low-pass filter (with a cutoff frequency of 0.3Hz) and then negated this component from the original signal. We then re-sampled the data from all the sensors to 30Hz to have a common sampling rate among all datasets. We divided the accelerometer data into individual windows of 90 samples, with a sliding window offset of 45; this results in 50% overlap between consecutive windows and a 3 seconds per window. The choice of a 3-second window allows for a full cycle of many of the activities (e.g. 2 steps) to be captured – since the cadence of an average person walking is $90 - 130$ steps/min [22].

3) Evaluation: During our experiments, we split the samples into two equal user groups: A and B, each containing half of the total number of users in a dataset while maintaining an equal ratio of gender and position between the two groups. To train the Activity Embedding part of *CoDEm*, we treat group A as having the activity labeled training set and group B as the test set. To learn the Domain Embeddings in *CoDEm*, we treat group B as the training set containing the meta-data information and test on group A. This ensures that the activity and domain embeddings are trained on disjoints set of samples, reflecting a more realistic scenario (as mentioned in Section II). However, the multi-task learning baselines do not support training with a disjoint set of samples for activity and meta-data labels. Hence, for those baselines, we use both activity and meta-data labels from group A.

Within each training subset, we employ an 80/20 train/validation split. To emphasize the statistical significance of our results, we repeat run our experiment 10 times with different user/trial combinations in groups A and B and report the average metrics (Section V-D). We run a paired Wilcoxon signed-rank test between the 10 runs of each baseline, calculate the p value *CoDEm* and only report the result if $p < 0.01$ in Section VI, emphasizing that all the reported findings are statistically significant.

4) Model Architecture & Hyper-parameters: Each of the E_* function is represented by a convolutional `Encoder` followed by multi-layer perceptron `Projector`. To make sure *CoDEm* can be potentially run efficiently in small wearables, we developed a lightweight convolutional architecture inspired by MobileNet [23]. Our `Encoder` consists of three layers of Depth-wise Separable Convolution operations, each having their associated Batch Normalization and 2:1 Max Pooling

TABLE II: Summary of Three Public Datasets Used in Our Experiments

Dataset	Subjects	Positions	Labels	Sampling Frequency
SHAR	30 (24 Female, 6 Male)	2 (Left & Right Trouser Pocket)	9 (standing up from laying, lying down to standing, standing up from sitting, running, sitting down, going downstairs, going upstairs, walking, and jumping)	100Hz
PAMAP2	9 (8 Males, 1 Female)	3 (Wrist, Chest, Ankle)	11 (lying down, sitting, standing, walking, running, cycling, ascending stairs, descending stairs, vacuum cleaning, ironing, rope jumping)	30Hz
OPPORTUNITY	4	5 (BACK, Right Upper Arm (RUA), Right Left Arm (RLA), Left Upper Arm (LUA), Left Lower Arm (LLA))	4 (sitting, standing, walking, lying)	50Hz

operation. The final convolution is followed by Global Average Pooling [24] operation and directly fed to the MLP `Projector`. The output of the `Projector` is $L2$ normalized so that the resulting embedding vector resides in a unit normalized hyper-sphere. Finally, this vector is fed to the classifier `C`, a single fully-connected layer with a `Softmax` activation. The resultant network contains only $53K$ trainable parameters, making it very suitable for small low-powered wearable devices.

We applied Dropout regularization between each layer and used Adam optimizer with weight decay, which provided another form of regularization. We also opted for a learning rate scheduler that reduced the learning rate by a factor of 10 with a patience factor of 10. The batch sizes were set to 256 during all the experiments as we noticed that the large batch size leads to faster convergence for the contrastive center loss. We optimized our model hyper-parameters with Randomized Search on the validation sets, the final parameters used throughout the experiments are listed in Table III. Our implementation of *CoDEm* is available at GitHub[1].

TABLE III: List of Hyper-parameters

Hyper-Parameters			Values
Encoder	Convolution	Layers	3
		Filters	128, 128, 128
		Kernels	9, 9, 9
	MLP	Layers	2
		Neurons	64, 64
Classifier		Neurons	64
Batch Size			256
Dropout Rate			0.1
Epochs			100
Learning Rate			0.001

D. Evaluation Metrics

1) Silhouette Score: To evaluate the quality of the learned *Activity* and *Domain Embeddings*, we use *Mean Silhouette Coefficient* [25] for all the samples in the validation sets. This score measures, in the embedding space, how each sample is similar is to its own class (cohesion) compared to other classes (separation). If the mean distance between embedding vector $z^{(i)}$ and all other vectors in the same class is $a^{(i)}$, and the mean distance between $z^{(i)}$ and all other vectors in the *next nearest* class in the embedding space is $b^{(i)}$, the *Mean Silhouette Coefficient* is defined as,

$$SC = \frac{1}{N}\sum_{i=1}^{N}\frac{b^{(i)} - a^{(i)}}{\max(a^{(i)}, b^{(i)})} \tag{14}$$

The coefficient score ranges between -1 and +1. A high positive value indicates the embedding vectors lie in close proximity to other vectors of their own classes and far away from vectors of other classes. A negative value indicates that embeddings vectors from different classes occupy close neighborhoods which is indicative of bad embedding. Values close to 0 indicate overlapping clusters.

2) F1 Score: Driven by observation of imbalanced class distribution in the three datasets (Figure 5) and accuracy being a misleadingly optimistic metric [26] in such a scenario, we choose to use macro-`F1` score as the primary performance metric. This prevents easily classifiable high support activity labels from dominating the performance metric.

VI. Results

1) Primary Analysis: In this section, we compare the classification performance of *CoDEm* against the baselines with the three benchmark datasets. In Table IV we show the mean macro-`F1` and Silhouette Score from 10 independent runs.

For the Single Task (activity classification) baseline, we show the performance with and without the contrastive center loss. We note that minimizing intra-class variability and maximizing inter-class separability through the contrastive center loss provides 3.24%, 9.98%, 10% improvement in silhouette score in SHAR, PAMAP2, and OPPORTUNITY dataset, corresponding to 4.3%, 3.6%, 4.8% improvement in macro-`F1` scores, respectively. This validates our intuition to use contrastive learning to improve classification performance.

In Multi-task baselines, we evaluate the performance of both CoOp-MTL and Adv-MTL with two types of auxiliary supervision: (i) gender and (ii) position information and their combination (OPPORTUNITY dataset only has position information). We notice that in all cases, both the silhouette and macro-`F1` scores take a considerable hit compared to STL results. There is also inconsistency on the level of performance dip – in CoOp-MTL setup, using the gender and position metadata together results in the biggest performance dip, while in Adv-MTL the degradation is the least. This further hints that the simplistic assumptions of global co-operative or adversarial MTL models are not well suited to integrate the auxiliary

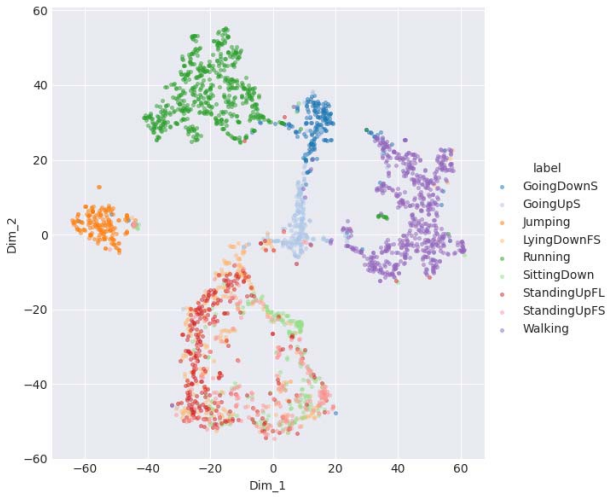
[1] https://github.com/azmfaridee/codem-smartcomp-2022

supervision from the meta-data labels and a learnable, local, per sample relationship should help improve the HAR model's performance.

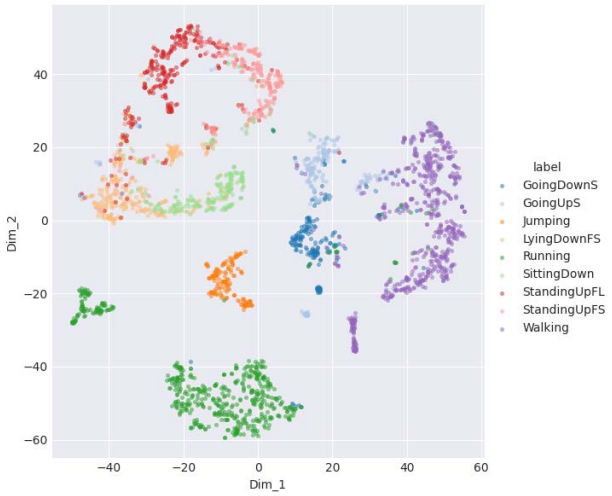
Finally, we analyze the performance of *CoDEm* on similar types of auxiliary supervision. As we can clearly see, using the gender and position information together results in the highest amount of performance improvement by *CoDEm*. *CoDEm* outperforms all baselines both in terms of silhouette and macro-F1 score. More specifically, *CoDEm* achieves 9.63%, 12.07%, 11.94% higher macro-F1 score and 11.22%, 16.66%, 24.65% higher silhouette scores in SHAR, PAMAP2, and OPPORTUNITY, respectively compared to the STL baseline. That is an average of **11.21%** macro-F1 and **17.51%** silhouette score improvement.

Similarly *CoDEm* achieves 13.77%, 13.36%, 8.41% macro-F1 and 21.62%, 13.47%, 13.23% silhouette score improvement compared to CoOp-MTL and 11.57%, 8.98%, 7.85% macro-F1 and 16.95%, 10.46%, 23.65% silhouette score improvement compared to Adv-MTL baselines in SHAR, PAMAP2 and OPPORTUNITY dataset. On average *CoDEm* achieves **9.07%**, **9.46%** macro-F1 score and **16.10%**, **17.02%** silhouette score improvement over CoOp-MTL and Adv-MTL baselines. We would like to note that, both the MTL baselines are trained samples that contain both activity and meta-data labels whereas *CoDEm* is trained with samples that contain either the activity or meta-data label (not both). Hence a macro-F1 score improvement of ≈**9.5%** by *CoDEm* in such a challenging scenario is very encouraging.

2) Visual Analysis: To understand how samples in the *Domain Aware Activity Embedding* vector $\hat{z}^{(i)}_{\texttt{act}(l)}$ are better distributed than the single task embedding vector $z^{(i)}_{\texttt{act}(l)}$ for higher classification performance, we employ 2D TSNE visualization on one of the runs on SHAR dataset, the result of which is shown in Figure 6. As shown in Figure 6a, there is a high amount of overlap between samples of *LyingDownFS*, *StandingUpFS*, *SittingDown* and *StandingUpFL* activities. This results in poor F1 scores in those classes (Figure 8a) as they are confused with each other majority of the time (Figure 7a). *CoDEm*, on the other hand, can learn a higher quality embedding space (11% better silhouette score), which is reflected in Figure 6b, and results in a much higher (35% more) F1 score for those classes (Figure 8b).



(a) Single Task, $SC = 0.54$



(b) *CoDEm* , $SC = 0.65$

Fig. 6: TSNE visualization of Embedding Vector

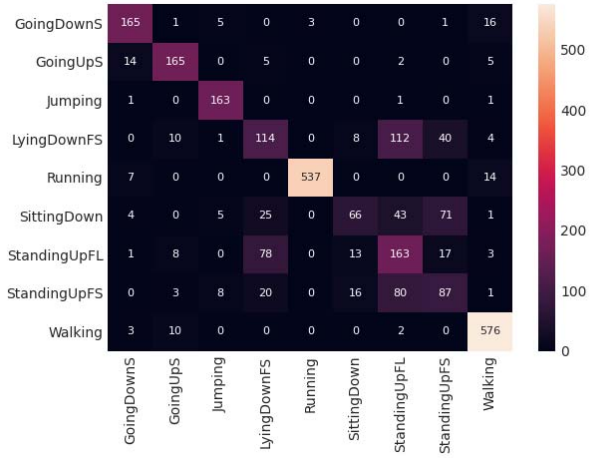
VII. Conclusion and Future Work

In this paper, we presented *CoDEm*, a novel framework to exploit meta-data information (user attributes such as gender, device position, etc.) to improve HAR classification accuracy in three benchmark datasets. In contrast to Co-operative or Adversarial MTL learning methods that learn a shared representation from samples containing both meta-data and activity labels, *CoDEm* offer hyper-parameter free learning of separate embeddings from a disjoint set of samples containing meta-data and activity labels. Combining contrastive learning with a novel residual attention mechanism to learn highly compact representation with an average 17.5% improved silhouette score, *CoDEm* offer an average $9.5-11\%$ improved macro-F1 score over the baselines in the three benchmark datasets.

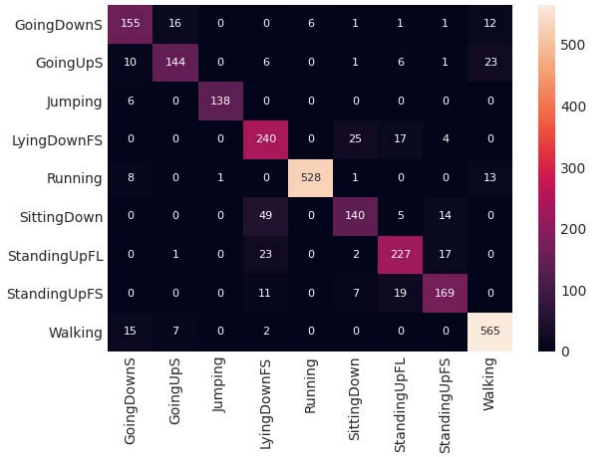
Although we envisioned *CoDEm* as being able to handle device or other types of heterogeneity in addition to person/gender and positions, in this work we did not explore device heterogeneity due to the lack of device labels in the three benchmark datasets. In the future, we would like to conduct extensive experiments to include a diverse set of heterogeneities. Despite being designed to handle multiple-positional heterogeneity, the input to *CoDEm* actually consists of a single positional data at a time which makes *CoDEm*, at its current form unsuitable input streams consisting of multiple positional data [27], [28]. Our future work would investigate means to mitigate this limitation. We would also like to investigate whether the integration of recent contrastive-learning architectures (e.g. SupContrast [12], SimCLR [29])

TABLE IV: F1 and Mean Silhouette Coefficient (in parentheses) in SHAR, PAMAP2, and OPPORTUNITY dataset. 'CC' denotes Contrastive Center loss.

Model	**Single Task**		**Multi-Task**						***CoDEm***		
Relationship	N/A		Co-operative			Adversarial			Learnable		
Loss	Softmax	Softmax+CC	Softmax	Softmax	Softmax	Softmax	Softmax	Softmax	Softmax+CC	Softmax+CC	Softmax+CC
Extra Supervision	N/A	N/A	Gender	Position	Gender+Position	Gender	Position	Gender+Position	Gender	Position	Gender+Position
SHAR	0.7493 (0.5404)	0.7925 (0.5729)	0.7301 (0.4921)	0.7284 (0.4683)	0.7079 (0.4364)	0.7271 (0.4701)	0.7289 (0.4747)	0.7299 (0.4831)	0.8312 (0.6122)	0.8305 (0.6026)	**0.8456 (0.6526)**
PAMAP2	0.6567 (0.3108)	0.6928 (0.4106)	0.6407 (0.3521)	0.6322 (0.3330)	0.6438 (0.3427)	0.6629 (0.3731)	0.6412 (0.3395)	0.6876 (0.3728)	0.7282 (0.4191)	0.7315 (0.4318)	**0.7774 (0.4774)**
OPPORTUNITY	0.7339 (0.4327)	0.7821 (0.5327)	N/A	0.7692 (0.4869)	N/A	N/A	0.7748 (0.4427)	N/A	N/A	**0.8533 (0.6792)**	N/A

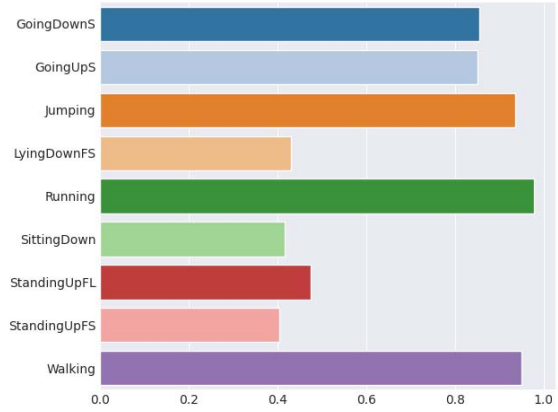


(a) Single Task

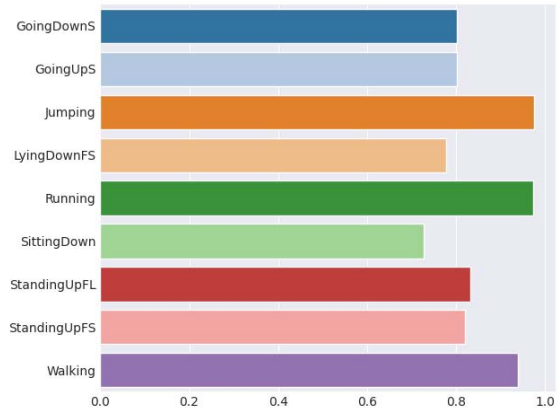


(b) *CoDEm*

Fig. 7: Confusion Matrix



(a) Single Task



(b) *CoDEm*

Fig. 8: F1 values per activity class

would provide further improvements *CoDEm's* performance.

As the *Domain Embedding* module can be trained to learn the embedding of sensitive personal information (e.g., personal identity, gender), a valid question on the practical use case of *CoDEm* is whether *CoDEm* poses any serious privacy risk. As we can see, *CoDEm* uses the *Domain Embedding* information during its intermediate steps (Equation 12) and does not expose such information in its final output of activity labels. Nevertheless, we would like to perform an in-depth investigation along this avenue in our future work.

ACKNOWLEDGMENT

This research is supported by NSF CAREER grant 1750936, U.S. Army grant W911NF2120076 and the National Research Foundation, Singapore under its NRF Investigatorship grant (NRF − NRFI05 − 2019 − 0007).

REFERENCES

[1] S. Ware, C. Yue, R. Morillo, J. Lu, C. Shang, J. Bi, J. Kamath, A. Russell, A. Bamis, and B. Wang, "Predicting depressive symptoms using smartphone data," *Smart Health*, vol. 15, p. 100093, 2020.

[2] S. Bobovych, F. Sayeed, N. Banerjee, R. Robucci, and R. P. Allen, "Resteaze: low-power accurate sleep monitoring using a wearable multi-sensor ankle band," *Smart Health*, vol. 16, p. 100113, 2020.

[3] S. Milanko and S. Jain, "Liftright: quantifying strength training performance using a wearable sensor," *Smart Health*, vol. 16, p. 100115, 2020.

[4] E. Rastegari and H. Ali, "A bag-of-words feature engineering approach for assessing health conditions using accelerometer data," *Smart Health*, vol. 16, p. 100116, 2020.

[5] M. Boukhechba, A. R. Daros, K. Fua, P. I. Chow, B. A. Teachman, and L. E. Barnes, "Demonicsalmon: Monitoring mental health and social interactions of college students using smartphones," *Smart Health*, vol. 9, pp. 192–203, 2018.

[6] S. Liu, Y. Liang, and A. Gitter, "Loss-balanced task weighting to reduce negative transfer in multi-task learning," in *AAAI*, vol. 33, no. 01, 2019, pp. 9977–9978.

[7] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *CVPR*, 2018, pp. 7482–7491.

[8] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," *arXiv:1409.7495*, 2014.

[9] D. Micucci, M. Mobilio, and P. Napoletano, "Unimib shar: A dataset for human activity recognition using acceleration data from smartphones," *Applied Sciences*, vol. 7, no. 10, p. 1101, 2017.

[10] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *ISWC*. IEEE, 2012, pp. 108–109.

[11] D. Roggen and A. Calatroni, "Collecting complex activity datasets in highly rich networked sensor environments," in *INSS*. IEEE, 2010, pp. 233–240.

[12] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," *NeurIPS*, 2020.

[13] C. Qi and F. Su, "Contrastive-center loss for deep neural networks," in *ICIP*. IEEE, 2017, pp. 2851–2855.

[14] A. Faridee, M. Khan, N. Pathak, and N. Roy, "Augtoact: scaling complex human activity recognition with few labels," 2019, pp. 162–171.

[15] A. Z. M. Faridee, A. Chakma, A. Misra, and N. Roy, "Strangan: Adversarially-learnt spatial transformer for scalable human activity recognition," *Smart Health*, vol. 23, p. 100226, 2022.

[16] A. Abedin, M. Ehsanpour, Q. Shi, H. Rezatofighi, and D. C. Ranasinghe, "Attend and discriminate: Beyond the state-of-the-art for human activity recognition using wearable sensors," *ACM IMWUT*, vol. 5, no. 1, pp. 1–22, 2021.

[17] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *ECCV*. Springer, 2016, pp. 499–515.

[18] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *CVPR*, vol. 2. IEEE, 2006, pp. 1735–1742.

[19] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *arXiv:1508.04025*, 2015.

[20] C. I. Tang, I. Perez-Pozuelo, D. Spathis, S. Brage, N. Wareham, and C. Mascolo, "Selfhar: Improving human activity recognition through self-training with unlabeled data," *arXiv:2102.06073*, 2021.

[21] S. Van Buuren and C. G. Oudshoorn, "Multivariate imputation by chained equations," 2000.

[22] C. BenAbdelkader, R. Cutler, and L. Davis, "Stride and cadence as a biometric in automatic person identification and verification," in *IEEE Face & Gesture Recognition*. IEEE, 2002, pp. 372–377.

[23] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv:1704.04861*, 2017.

[24] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv:1312.4400*, 2013.

[25] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[26] Y. Chen, J. Wang, M. Huang, and H. Yu, "Cross-position activity recognition with stratified transfer learning," *PMC*, vol. 57, pp. 1–13, 2019.

[27] A. Z. M. Faridee, S. R. Ramamurthy, H. Hossain, and N. Roy, "Happyfeet: Recognizing and assessing dance on the floor," in *HotMobile*, vol. 2018-Febru. ACM, 2018, pp. 49–54.

[28] A. Z. Md Faridee, S. R. Ramamurthy, and N. Roy, "Happyfeet: Challenges in building an automated dance recognition and assessment tool," *GetMobile*, vol. 22, no. 3, pp. 10–16, 2019.

[29] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*. PMLR, 2020, pp. 1597–1607.