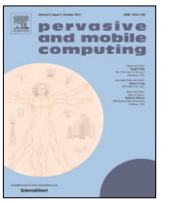
Pervasive and Mobile Computing 87 (2022) 101698

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc



# *STAR-Lite*: A light-weight scalable self-taught learning framework for older adults' activity recognition

Sreenivasan Ramasamy Ramamurthy [a,*], Indrajeet Ghosh [a], Aryya Gangopadhyay [a], Elizabeth Galik [b], Nirmalya Roy [a]

[a] Department of Information Systems, University of Maryland, Baltimore County, Baltimore, United States
[b] Department of Nursing, University of Maryland, Baltimore, United States

ARTICLE INFO

*Article history:*
Received 5 January 2022
Received in revised form 6 August 2022
Accepted 16 September 2022
Available online 21 September 2022

*Keywords:*
Unsupervised learning
Self-taught learning
Pre-training
Human activity recognition
Model compression
Pruning
Quantization

ABSTRACT

Recognizing activities for older adults is challenging as we observe a variety of activity patterns caused due to aging (e.g., limited dexterity, limb control, slower response time) or/and underlying health conditions (e.g., dementia). However, existing literature with deep learning methods has successfully recognized activities when the dataset contains high-quality annotations and is captured in a controlled environment. On the contrary, data captured in a real-world environment, especially with older adults exhibiting memory-related symptoms, varying psychological and mental health status, reliance on caregivers to perform daily activities, and unavailability of domain-specific annotators, makes obtaining quality data with annotations challenging; leaving us with limited labeled data and abundant unlabeled data. In this paper, we hypothesize that projecting the labeled data representations comprising a specific set of activities onto a new representation space characterized by the unlabeled data comprising activities beyond the limited activities in the labeled dataset would help us rely less on the annotated data to improve activity detection performance. Motivated by this, we propose *STAR-Lite*, a self-taught learning framework that involves a pre-training framework to prepare the new representation space considering activities beyond the initial labels in the labeled dataset. *STAR-Lite* projects the labeled data representations on the new representation space characterized by unlabeled data labels and learns higher-level representations of the labeled dataset while optimizing inter- and intra- class distances without explicitly using a computation hungry similarity-based approach. We demonstrate that our proposed approach, *STAR-Lite* (a) improves activity recognition performance in a supervised setting and (b) is feasible for real-world deployment. To enhance the feasibility of deploying *STAR-Lite* on devices with limited memory resources, we explore model compression techniques such as pruning and quantization and propose a novel layer-wise pruning-rate optimization technique that effectively compresses the network while preserving the model performance. The evaluation was performed using the Alzheimer's Activity Recognition dataset (AAR) captured from 25 individuals living in a retirement community center with IRB approval (#Y18NR12035) using an in-house *SenseBox* infrastructure while concurrently assessing the clinical evaluation of

* Corresponding author.
*E-mail addresses:* rsreeni1@umbc.edu (S. Ramasamy Ramamurthy), indrajeetghosh@umbc.edu (I. Ghosh), gangopad@umbc.edu (A. Gangopadhyay), galik@umaryland.edu (E. Galik), nroy@umbc.edu (N. Roy).

https://doi.org/10.1016/j.pmcj.2022.101698

the participants for dementia, and independent living. Our extensive evaluation reveals that *STAR-Lite* can detect activities with an F1-score of 85.12% despite 62% reduction in model size and 5% improvement of execution time on a resource constrained device.


## 1. Introduction

Recent innovative designs and developments of the Internet of Things (IoT) devices have caused an increase in the research and commercialization of technologies that use wearable and ambient sensor modalities. Such consumer-grade IoT devices have helped develop various state-of-the-art methods and frameworks in the domain of human health monitoring relying on activities of daily living (ADLs) and instrumental activities of daily living (IADLs) for early detection of physical mobility and cognitive health impairments. Monitoring health conditions and diagnosing the precursor to a specific functional and/or behavioral health symptom is usually performed by conducting various clinical and survey-based assessments, which involve multiple trips to the hospital and always turn out to be a hassle, especially for older adults. We believe an automated system that can (i) *sense activities*, (ii) *identify anomalies or abnormalities in an older adult's habitual activities or behavior*, and (iii) *alert the caregiver to take an appropriate action just in time* could greatly improve the quality of life of older adults while allowing them to live more independently. A variety of commercial wearable devices[1],[2] and sensors recently has flooded the market for healthcare applications. Other ambient sensors that have been researched for healthcare applications include passive infrared sensors [1–3], magnetic sensors, radars [4], acoustic sensors [5,6], everyday smart objects, and device-free sensing (WiFi [7]).

Several smart-home sensor systems [8,9] have shown their potential in detecting activities through which symptoms of an underlying health impairment were detected [3,10,11]. However, these state-of-the-art systems exhibit challenges while scaling the system for multiple people with varying ages and health statuses, different activities in a wider community, and various houses with different house layouts. Our overarching objective is to identify such challenges through our real-world deployment in community-dwelling apartments in a retirement community center and postulate appropriate solutions to mitigate underlying system and data-related problems. Specifically, the target population for this work is older adults living with neurocognitive disorders such as *Dementia*. Activity recognition for older adults living with *Dementia* is a much challenging problem as a result of their underlying cognitive, functional, and behavioral changes and aging-related functional inability. For instance, let us consider *Preparing a Sandwich* activity, which constitutes several micro-level activities such as *picking up a knife, butter, ingredients, applying butter on the bread*, and so on. Due to cognitive impairment, a person may perform the activity of *Preparing a Sandwich* in a *different order and repeating several micro-level activities*. Such inherent variations in the data cause the older adult's data to experience high inter-class similarity (two activity patterns look similar) and intra-class variability (same activity patterns look different) [12]. Older adults, in particular, exhibit such variations more frequently when compared to young adults. Due to the presence of such inherent variations in the data, a machine learning model for activity recognition *requires abundant data comprising of all such variations* in order to be robust. Besides, this abundant data will be useful only when paired with *high-quality and large-scale annotated labels*, which is challenging to acquire due to the following reasons: (i) *unavailability of annotators* as most older adults live a caretaker and annotating data would hamper their job, (ii) *lack of domain expertise* for annotators may provide irrelevant labels, (iii) *relying on camera-based systems causes privacy concerns*, (iv) *aversion towards cameras* causes the older adults to live in fear. Acquiring labels being a herculean task, it is imperative to circumvent these challenges by leaning on algorithms that require minimal labeled data instances. On the other hand, it is relatively easier to acquire unlabeled data as it would only involve sensors (wearables and ambient) used by older adults daily, especially in a retirement community center. Thus, the primary requirement for the algorithm would be to perform well with abundant unlabeled data and minimal labeled data.

Besides posing labeling challenges, the wider range of inherent variations in the data for older adults severely impacts the adaptability of the developed AR model in the real world, i.e., while scaling the AR model. Scaling refers to adapting an already developed sensing system/model to new scenarios. For example, the new scenarios can be a different and bigger population, young adults vs. older adults, or a population with different underlying health conditions. Therefore, designing an AR model for a general population and adapting it for older adults may cause the AR model to fail. Furthermore, due to the additional variations introduced by older adults due to the evolving physical and cognitive health changes due to aging, developing a generalized and scalable AR model for older adults is non-trivial.

Several machine learning paradigms such as *Active Learning*, *Transfer Learning*, *Self-Supervised Learning* have attempted to address the problem of limited availability of labeled data. Active learning benefits by acquiring labels of most informative data instances from an oracle [12]. However, it involves higher computation complexity as the algorithm

[1] https://www.fitbit.com/home

[2] https://www.empatica.com/en-eu/research/e4/

requires online training. Besides, active learning also relies on crowdsourcing to obtain the labels, which require a quality annotator with domain-specific knowledge to provide quality labels. On the other hand, transfer learning involves transferring meaningful information from a source domain to a target domain. We believe transfer learning is one of the most appropriate techniques due to the challenges involved in acquiring data from older adults. Some semi- or self-supervised techniques have been found useful where partial label information from the ground truth is leveraged in fine-tuning the algorithm's parameters. However, these methodology requires to form pairs, triplets, and N-pair [13–17] to optimize their loss functions that drastically increases the computational complexity.

The overall objective of this paper is to develop a light-weight AR model in comparison to our prior work [18] that can be readily deployable in a real-world scenario for older adults with the following characteristics: (a) *capable of scaling with minimal labeled and abundant unlabeled data*, (b) *optimize inter-class similarity and intra-class variations*, (c) *has minimal memory footprint and average model inference time*. Considering these requirements in mind, we propose a self-taught learning framework called *STAR-Lite*. Our proposed framework, *STAR-Lite* is an unsupervised activity recognition model that can leverage the benefits of both minimal labeled data and abundant unlabeled data and learn the generic feature representations. The learned representations are autonomously optimized for inter-class similarities and intra-class variations without using a computation-hungry similarity-based approach. The **key contributions** of the paper are enumerated below.

- **Self-Taught Learning Framework**: To address the inherent variations in the activity data of older adults, we propose a novel methodology involving a pre-training phase that helps us learn a new representation space characterized by unlabeled data labels, and a self-taught learning framework that projects that labeled data representations in the newly learned representation space. This framework learns labeled data representations better from limited data while optimizing inter-class similarities and intra-class variations and helps improve the classification performance drastically.
- **Data Collection from Multiple Smart Homes**: We presented a data collection system for activity recognition in a smart home setting deployed and used in this study. We described the data collection procedure and the dataset (namely *Alzheimer's Activity Recognition (AAR) dataset*, which comprises sensor data and survey-based data that can help assess an older adult's functional and behavioral health.
- **Evaluation in Practical Setting**: We demonstrated that the proposed framework, *STAR-Lite* helps reduce the misclassifications related to the system-level faults and data collection errors. Besides, we showcased that a simpler CNN network is sufficient to boost the performance of the downstream task using *STAR-Lite*. Finally, we evaluated *STAR-Lite* using the in-house Alzheimer's Activity Recognition (AAR) dataset collected from 25 apartments in a retirement community center with IRB approval (IRB #Y18NR12035) and contrasted the performance to a publicly available dataset that comprises a different age-group population.
- **Model Compression:** We evaluate the adaptability of *STAR-Lite* across multiple smart homes and show how post-training model compression techniques, such as pruning and quantization, can help reduce the memory and computational footprints of the *STAR-Lite* model. We also propose an iterative layer-wise pruning rate optimization technique to determine an optimum value to preserve the model performance while minimizing the memory and computational footprints of *STAR-Lite* model.

Our experiments reveal that the self-taught learning methodology outperforms the state-of-the-art self-supervised learning algorithm [19] in detecting activities by 17% and a LSTM based model [20]. Besides, we also show a 62% reduction in the model size and 28% reduction in average execution time on a single-core Intel processor and 5% reduction on a resource constrained device, indicating that *STAR-Lite* can conveniently be deployed on resource-constrained devices.

The reminder of the manuscript is organized as follows. In Section 2, we discuss the related work while Section 3 discusses the methodology of the unsupervised pre-training phase, supervised classification and the model compression techniques. Section 4 describes the dataset used in our experimentation pipeline (Section 5) and Section 6 shows the experimental results. We report the feasibility study for resource constrained device implementation in Section 7. Finally, we discuss and conclude in Section 8 and 9

## 2. Related work

This section focuses on the literature of activity recognition, self-taught learning and model compression techniques that help us motivate our work.

### 2.1. Activity recognition

Activity Recognition is one of the heavily investigated topics in the past decade, especially after deep learning gained its popularity [12,21]. Typically, the sensors used in AR research are Inertial Measurement Units (Accelerometer, Magnetometer, Gyroscope) and ambient sensors. Several datasets, such as OPPORTUNITY, PMAP2, WISDM, etc., have been collected using these sensors. Besides, most work attempts to classify activities from the sensor data using deep learning approaches such as Convolutional Neural Networks [22,23], Recurrent Neural Networks [22,23], Deep Belief Networks [24],

and but not limited to Autoencoders [25]. In contrast to the existing literature, our approach involves collecting data from a real-world deployment from a population of older adults living with neurocognitive disorders.

Several works exist that helps improve classification performance with a minimal amount labeled dataset. Active learning, a popular method that falls under the former category, determines the most uncertain data instances and queries the label from an annotator. Such methods drastically reduce the labeling efforts, time and help improve the classification performance in a supervised setting. Another popular method, Transfer learning, lets us transfer knowledge from a source domain to a target domain, thus minimizing the training effort. The target domain could be a different person, scenario, surroundings, and so on. Transfer learning can be broadly classified as Inductive, unsupervised, and transductive based on the similarity between the source and target domains. *STAR-Lite* falls under the category of the inductive transfer. Several algorithms have been proposed for activity recognition using transfer learning to boost the classification performance in the target domain [1,26,27]. In a study, the authors have proposed a framework that leverages autoencoder features from unlabeled data and transfer the first two layers to the target domain to recognize unseen activities in the target domain [28]. In such works, the assumption remains that the source and target domain label space follows the same distribution in contrast to our approach, *STAR-Lite*, which assumes otherwise.

### 2.2. Self-taught learning

Self-taught learning, which is a special case of inductive transfer learning, where the model learns from the unlabeled data. Besides, the primary assumption is that the labeled data and the unlabeled data follow different generative conditional distributions (a different subset of activities are present in both labeled and unlabeled data domains). Such an assumption has proven to learn good representations of the input data from the unlabeled data and help aid the classification task in a supervised setting [29]. Self taught learning has shown to be effective in various fields such as audio classification [30,31], E-Nose in Wound Infection Detection [32], facial beauty prediction [33], image classification [29]. In activity recognition, the authors created codebooks of features called basis vectors corresponding to each activity using sparse encoding from the unlabeled data [25]. Further, a distance-based methodology was employed to determine the most appropriate basis vector, and the basis vector-label pair were classified in a supervised setting. In contrast to this methodology, although our work involves a pre-training phase from unlabeled date, our method does not involve additional computational complexity related to identifying appropriate basis vectors. We believe, restricting the number of basis vectors negatively impacts the quality of the latent representation space learned. In another work, the authors proposed a feature similarity search algorithm in conjunction with self-taught learning to learn features from unlabeled data [34]. However, this work is limited to a very preliminary analysis using a dataset with one participant and four activities performed over a 6-hour period. In contrast, our work is evaluated on a dataset with 25 participants performing 14 activities.

Another paradigm of machine learning that leverages unlabeled data to aid supervised classification is self-supervised learning. Self-supervised learning uses representations learned from unlabeled data through unsupervised transformations and further uses transfer learning to assist the downstream task in a supervised setting similar to our approach. However, *STAR-Lite* does not leverage any label information during the representation learning phase. Additionally, we believe that the augmentations such as jittering, cropping, rotation [19] of the data do not represent the true variation in the activity recognition dataset. We believe that the feature engineering step of using wavelet coefficients would expose the variations in the time–frequency domain to the pre-training and classification phases. In AR, authors of [19] proposed a multi-task learning framework to learn auxiliary tasks (augmentation of the unlabeled data) during the pre-training phase and used the primary task to perform classification, which makes it a semi-supervised approach.

### 2.3. Model compression

Model compression is a salient approach that enables us to perform real-time inference on-edge devices and hardware accelerators. Model compression can be broadly classified into four major techniques: pruning, quantization, knowledge distillation, low-rank factorization, and compact convolutional filters [35]. Among the above-listed techniques, pruning and quantization gained much popularity due to their robustness, support for both training & post-training settings, reducing model network complexity, and mitigating overfitting problem [36] characteristics. Furthermore, post-training pruning and quantization techniques will be appropriate for evaluating *STAR-Lite* implementation due to the model-friendly characteristics. In paper [37], the authors proposed an architecture-agnostic approach, *Degree-Quant* (DQ), which helped enable an improvement to the existing baselines of the quantization-aware training for graph neural networks. Furthermore, they used a stochastic protective nodes method to achieve higher and better weight update accuracy for quantization, where a protective node mask is generated for each layer.

Furthermore, the authors of [38] proposed a novel post-training weight pruning for deep learning algorithms. The proposed pipeline comprises three major modules: layer-wise sparsity rate selection, weight, and activation bias correctness, and local layer-wise fine-tuning method using auxiliary knowledge distillation (KD) losses. The KD mean-squared error loss measures the difference of the activation features space of the unpruned and prune layers. Similarly, in paper [39], the authors proposed a pruning method EagleEye, where the proposed evaluation pipeline is based on an adaptive batch normalization (BN) approach. The proposed pruning pipeline is classified into three major components: filter pruning, strategy generation, and adaptive BN. They performed a quantitative analysis to compute the correlation coefficient between pruning candidates (pruning strategy generation) and their corresponding accuracy.

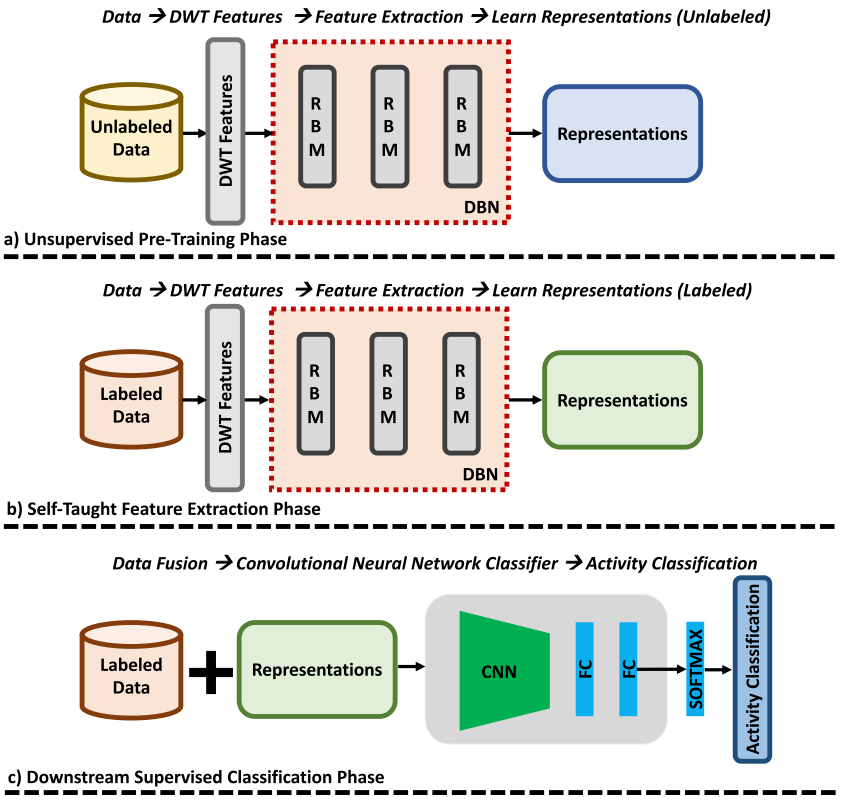



**Fig. 1.** Overall Architecture to illustrate the pre-training phase, the weight transfer and how it aids the classification in supervised setting.

## 3. Methodology

### 3.1. Self taught learning

This section describes the proposed methodology that comprises an unsupervised feature learning phase, a feature fusion component, and the supervised classification phase. A detailed pictorial representation of the pipeline is described in Fig. 1.

#### 3.1.1. Unsupervised pre-training phase

We propose to leverage the unlabeled data to learn the parameters using generative stochastic neural networks in the pre-training phase. The notion for using the unlabeled data is because the unlabeled data comprises activities outside of the labels included in the labeled dataset. In this study, we propose to use Deep Belief Networks (DBN) to learn high-level representations (from unlabeled data) and further use the learned parameters (from unlabeled data) to generate high-level representations for labeled data for the classification task. Let us now describe the mathematical formulation of the pre-training phase using DBN. A DBN is a probabilistic generative network that is formed by stacking Restricted Boltzmann Machines (RBM). RBMs are probabilistic neural networks that comprise of neurons in two groups (visible and hidden) and form a bipartite graph between them, with a restriction that the neurons are not connected to each other in the same group. Let us denote the visible layers as $v_i$ and hidden layers as $h_j$, where $i$ and $j$ represent the neurons in the visible and hidden layers, respectively. The weight update required during the training phase of the RBM is performed using gradient descent and is given by Eq. (1).

$$w_{ij}(t+1) = w_{ij}(t) + \eta \frac{\partial P(v)}{\partial w_{ij}} \tag{1}$$

In Eq. (1), $\eta$ represents the learning rate, $w_{ij}$ is the weight vector connecting the visible layer and the hidden layer, and $P(v)$ is the probability distribution over the visible vectors defined using an energy function is shown in Eq. (2):

$$P(v) = \frac{1}{Z} \sum_{h} exp(-E(\mathbf{v}, \mathbf{h})) \tag{2}$$

**Algorithm 1** : Pseudocode of Contrastive Divergence, *CD*().

**Input:** unlabeled data ($X_u$)
**Output:** RBM trained weights ($W_{RBM}$)
*Initialization* : visible units, $v \leftarrow X_u$
**for** epoch = 1 to $E$, total epochs **do**
  **for** k = 1 to $N$, total number of instances in $X_u$ **do**
    update hidden units, $p(h_j = 1|v)$
    reconstruction step: $p(v_i = 1|h)$
    update hidden units, $p(h_j = 1|v)$
    update weight, $w_{ij}(t+1) = w_{ij}(t) + \eta \frac{\partial P(v)}{\partial w_{ij}}$
  **end for**
**end for**
**return** $W_{RBM}$

Here, $E(\mathbf{v}, \mathbf{h})$ is the energy function. A lower value of energy function is desirable and is thus minimized during training by adjusting the weights and biases. The derivative of the log probability in Eq. (1) with respect to the weights are defined as

$$\frac{\partial P(v)}{\partial w_{ij}} = \langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model} \tag{3}$$

where $\langle\rangle$ denotes the expectations under the distribution by the subscript. Since there is no connection among the hidden layers and among visible layers, it is possible to obtain $\langle v_i h_j \rangle_{data}$ for the visible layer given the hidden layer (using Eq. (5)) and hidden layer given visible layer (using Eq. (4)).

$$p(h_j = 1|v) = \sigma(b_j + \sum_i v_i w_{ij}) \tag{4}$$

$$p(v_i = 1|h) = \sigma(a_i + \sum_j h_j w_{ij}) \tag{5}$$

However, obtaining $\langle v_i h_j \rangle_{model}$ is cumbersome as it requires alternating Gibbs Sampling. Hence, we use [40] to train RBM using Contrastive Divergence (as explained in algorithm 1). Besides, training DBNs is a greedy process; each RBMs are individually trained before moving on to the next RBM. The activations of the first RBM (after training) are fed as an input to the second RBM, and so on (as explained in algorithm 2).

### 3.1.2. Self-taught feature leaning phase

After the pre-training phase, the next step is to feed the labeled data to the pre-trained DBN network to obtain representations. These representations of the labeled data are projected in a representation phase obtained with respect to the unlabeled data representations. The unlabeled data representations are characterized by activity labels that are not encountered in the labeled space. To obtain these representations of the labeled data, we subject the labeled data to the same feature extraction methodology (using DBN) proposed in the unsupervised pre-training step initialized with the parameters learned using the unlabeled data. Let us assume to have $m$ training data instances denoted by $X_l$ with class labels $y_l$. The unlabeled data be denoted as $X_u$. Besides, let us denote the DBN feature extraction function as $f_{DBN}(.)$, so, the representation of the labeled data in the new representation space, $R_l$ is defined as $R_l = f_{DBN}(X_l)$ as shown in Fig. 1b. Now, the new data instance for the supervised classification task is obtained by fusing the input data, $X_l$ and the representation, $R_l$, i.e., $X_l^{fused} = [X_l, R_l]$. The data-label fed to the supervised classification pair is denoted as $(X_l^{fused}, yl)$.

Careful examination of this methodology reveals that there are two crucial steps: (i) representation learning in a new feature space and (ii) fusing input data and the learned representations. Fig. 2 illustrates the effect of these two steps. As the pre-training phase for the labeled data learns representations with respect to the activities in the unlabeled data, it tries to maximize the distance between the labeled and unlabeled data in the representation space. On the other hand, the second step for fusing the input data with the representations introduces bias to the fused data by introducing the inherent variations of different activities in the raw data, which help maximize the intra-class variations. Collectively, the pre-training and the self-taught feature learning phases help us enhance both intra-class and intra-class distances, which consequently helps with the classification performance.

### 3.1.3. Computational complexity

It is evident that the unsupervised pre-training phase is performed in addition to the classification pipeline (discussed in the following section). Therefore, it is critical to evaluate and discuss the computational complexity of the unsupervised pre-training phase. As explained in the algorithm 1, each epoch comprises of a forward pass and a backward pass both involving weight updation. For k data instances, the forward pass involves matrix multiplication (Eq. (4)) with a

**Algorithm 2** : Pseudocode of extracting self-taught features.

**Input:** unlabeled data ($X_u$), labeled data ($X_l$), $CD(X_u)$
**Output:** self-taught data, $X_{self}$
1: **for** each *RBM* in *DBN* **do**
2: $W_{RBM} \leftarrow$ training_cd($X_u$)
3: $W_{DBN} \leftarrow$ stack ($W_{RBM}$)
4: **end for**
5: **for** m = 1 to $M$, total number of instances in $X_l$ **do**
6: $R_t \leftarrow$ Compute forward pass, $f_{DBN}(X_l^m)$
7: $temp \leftarrow$ append($X_l^m$, $R(t)$)
8: **end for**
9: $X_{self} \leftarrow temp$
10: **return** $X_{self}$

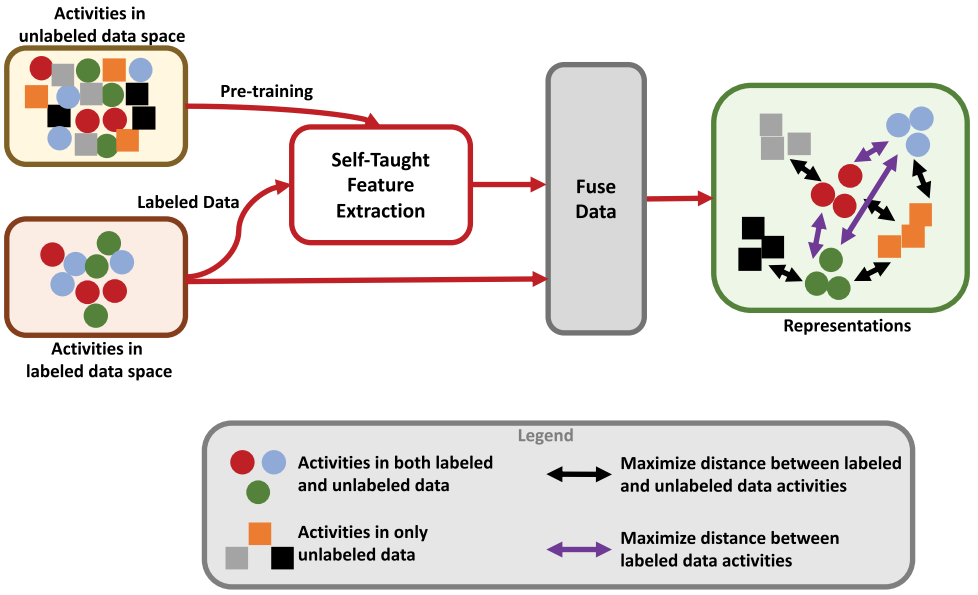



**Fig. 2.** Illustration of labeled data representations learned by self-taught feature extraction phase.

computational complexity of $\mathcal{O}(i * j * k)$. The weight updation equation's computational complexity can be written as $\mathcal{O}(i * j * k + exp(i + j + k))$. As the backward pass involves the same steps as the forward pass, the overall complexity can be written as $\mathcal{O}(2 * [i * j * k + exp(i + j + k)])$ and simplified to $\mathcal{O}(i * j * k + exp(i + j + k))$.

### 3.1.4. Supervised classification phase

The fused data obtained from the self-taught feature learning phase is then fed to a Convolutional Neural Network (CNN) that comprises 1-D convolutional layers and fully connected layers for classification. The objective of this phase is to learn a model, $\mathcal{M}$, that maps the fused data, ($X_l^{fused}$ and the label, $y_l$).

## 3.2. Model compression

Consider a scenario of a retirement community center where multiple older adults live as a community. If the center decides to implement a system for the older adults to monitor their activities to provide them an assisted living concierge service, we believe a system like *STAR-Lite* would be ideal. In such a scenario, an instance of *STAR-Lite* for each older adult would be running on a remote server operated by the service provider. In an attempt to provide a real-time inference of activities, especially for detecting fall, or to determine if the older adults need help, *STAR-Lite* must execute in real-time, i.e., the latency in processing the input data and provide the model inference should be minimal. In this section, we postulate how to compress the already trained AR model to reduce memory usage and inference time. We evaluate the performance of our proposed compressed AR model using two compression techniques with two types of model compression techniques: *pruning* and *quantization*.

**Algorithm 3** : Pseudocode of layer-wise pruning rate optimization.

**Input:** Pre-trained Model ($\mathcal{M}$), Fused Data ($X_l^{fused}$), Labels ($y_l$)
**Output:** Layer-wise prune rate, $\mathcal{P}$
**Initialize:** $min \leftarrow 0$, $max \leftarrow 1$, prune rate:$\rho \leftarrow 0$, $BestCost \leftarrow MinCost$

**for** each *layer* in $\mathcal{M}$ **do**
  $MinCost \leftarrow \mathcal{M}(X_l^{fused})\forall\rho \leftarrow min$
  $MaxCost \leftarrow \mathcal{M}(X_l^{fused})\forall\rho \leftarrow max$
  $\mathbb{P} = (min + max)/2$
  $PruneCost \leftarrow \mathcal{M}(X_l^{fused})\forall\rho \leftarrow \mathbb{P}$
  **if** $|MaxCost - PruneCost| > |PruneCost - MinCost|$ **then**
    $min \leftarrow \mathbb{P}$
  **else**
    $max \leftarrow \mathbb{P}$
  **end if**
  **if** $PruneCost > BestCost$ **then**
    $BestCost \leftarrow PruneCost$
    $Append(\mathcal{P}, \mathbb{P})$
  **else**
    $max \leftarrow \mathcal{P}$
  **end if**
**end for**
**return** $\mathcal{P}$

#### 3.2.1. Network pruning

Pruning refers to removing *unimportant* connections in a neural network. In a deep architecture, the initial layers provide low-level features of the data. The information represented by these features are highly localized and very specific to a certain characteristic to a certain data instance. As we move deeper, the features obtained are the generalized representations of those low-level information. As each layer in a deep architecture contributes to various levels of information, we believe each layer must be pruned accordingly. Thus, we propose to prune the model in a layer-wise fashion and propose a simple optimization method to reduce the computational complexity as explained in algorithm 3.

#### 3.2.2. Quantization

We also employ quantization to compress our proposed AR model. Quantization refers to computing and storing network parameters at a lower bit-width (e.g., 8-bit integer instead of 32-bit floating-point). The perks of quantizing the network include (i) the network structure does not change, (ii) lower model size, and (iii) lower inference time, which are precisely the type of characteristics we desire for the proposed *STAR-Lite* model. From the literature, we chose the most desired 8-bit integer quantization for our work as it can potentially provide up to 4x improvement in model compression [37].

#### 3.2.3. Pruning + quantization

Model Compression using pruning alters the network structure. On the other hand, quantization does not alter the network structure. However, both pruning and quantization individually help with compressing the models without affecting the model performance. The contribution of each weight or activation in the network towards the classification performance may not be discretized as important/unimportant (in the case of pruning) or assumed important in the case of quantization (and reduce bit-width). This is because each weight and activation (despite being zero) helps with the computation of the logits. Thus, we hypothesize that combining quantization with layer-wise pruning can help retain such weights/activations that do not have a higher value but can be present in the compressed model at a lower precision to help with retaining the model performance.

## 4. Alzheimer's Activity Recognition (AAR) dataset

This section presents the Alzheimer's Activity Recognition (AAR) dataset created to study the relationship between activities and behavioral health, especially Dementia. AAR dataset was acquired from a population of 25 older adults living in a retirement living facility and possessing symptoms consistent with Dementia from a mix of individuals who are healthy, mild cognitive impairment, and cognitive impairment. The dataset has two components, sensor-based and survey-based data. The aim of capturing the sensor-based dataset was to record the movement patterns (activities performed on a daily basis: ADLs and IADLs; Table 1). In contrast, the survey-based questionnaire aimed to acquire the current state (clinical evaluation) of the individual's functional and behavioral health. Below, we describe the data collection procedure, system architecture, and details of both the sensor and survey-based datasets.

**Table 1**
List of activities in AAR dataset.

| Categories | Activities | Abbreviation |
|---|---|---|
| Activities of Daily Living(ADLs) | Sitting | Sit |
| | Standing | St |
| | Walking | Wa |
| Instrumental Activities of Daily Living (IADLs) | Brushing Hair | BH |
| | Folding Laundry | FL |
| | Phone | Ph |
| | Preparing Sandwich | PS |
| | Sweeping | Sw |
| | Taking Trash Out | TT |
| | Using Toothbrush | UT |
| | Wash Hands | WH |
| | Wear Jacket | WJ |
| | Wear Shoes | WS |
| | Writing | Wr |

### 4.1. Sensor system architecture

To collect the AAR dataset (especially the sensor-based data), we developed a raspberry-pi based system to integrate various sensors (extended from SenseBox [8]). Raspberry-pi is a Linux-based miniaturized computer that consists of Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit processor, which clocks at 1.4 GHz with 1 GB LPDDR2 SDRAM onboard. The role of the raspberry-pi in a smart-home is to act as a hub, connect to an existing network, provide internet connectivity to various sensors in the smart-home, and finally log the data. The notion of such a system was to develop a system that can be readily deployable in a new home. Further, the system is comprised of two categories of sensors: wearable and ambient. Empatica-E4 was used as the wearable sensor to record the movement (through accelerometry; 32Hz sampling frequency), Electro-Dermal Activity (EDA), skin temperature, and heart-rate variability. Besides, ambient sensors such as passive infrared sensors (PIR; in each room), reed switches (on each door), and object tags (on objects used on a daily basis) were connected to the hub. The hub was in continuous connection with a server (present in the author's laboratory) via a reverse-ssh tunnel for constant monitoring of the state of the hub and the connected sensors. The overview of the sensor system architecture is depicted in Fig. 3a and the placement of various ambient sensors are shown in Fig. 3b. Finally, we integrated cameras with the hub (for limited time usage) to acquire the ground truth and the cameras were place in each room. We further elaborate the dataset collection in the following sections.

### 4.2. Sensor-based data collection

As soon as the hub and the ambient sensors are placed in strategic positions, we request the participating individuals to wear the wearable device on their dominant hand (preferably, although not compulsory). Now, we divide our data collection duration into two phases: scripted activities and unscripted activities. For the first two hours, the participants are requested to perform scripted activities that involve both ADLs and IADLs with a camera online for ground truth collection. For the following 2 h, the participants are monitored with a camera for any activity (unscripted); however, none of the team members of data collection will be present in the house. Further, we leave the ambient sensors and the wearable device to operate in the house (no cameras) to collect unlabeled data for the next 20 h. This procedure was repeated for 25 different participants in their own homes, where the home's layout was different from each other.

### 4.3. Survey-based data collection

During the scripted activities collection (described in the previous section), we requested the participants to perform certain activities based on the scales used for clinical evaluation of functional and behavioral health assessment (described in this section). First, the survey-based questionnaire collects the basic demographics of the participants. The inclusion criteria to participate in the study includes (i) must be above 65 years of age, (ii) must be a candidate for the symptoms of dementia due to old age or any underlying neurodegenerative disorders. Hence, the first survey we conducted uses the Saint Louis University Mental Status (SLUMS) Examination to help screen for Alzheimer's or any other type of dementia. The advantage of SLUMS is that it can identify people with milder cognitive problems even if it has not risen to the level of dementia. The second survey we collected was related to mental health. We used the Geriatric Depression Rating Scale to measure the participant's mental depression state. Besides, we used the Geriatric Anxiety Scale to measure older adults' anxiety levels. Next, we used the Barthel Scale and the Lawton Instrumental Activities of Daily Living Scale to measure the performance of ADLs and IADLs, respectively. These measures were ideal for this study as they were performance-based and would provide us with the current state of functional health (can be compared to sensor-based data). We would ask the participants to perform the scripted activities, and based on their performance, we would score them, which we believe can help us measure their ability to carry out everyday activities required for independent living.

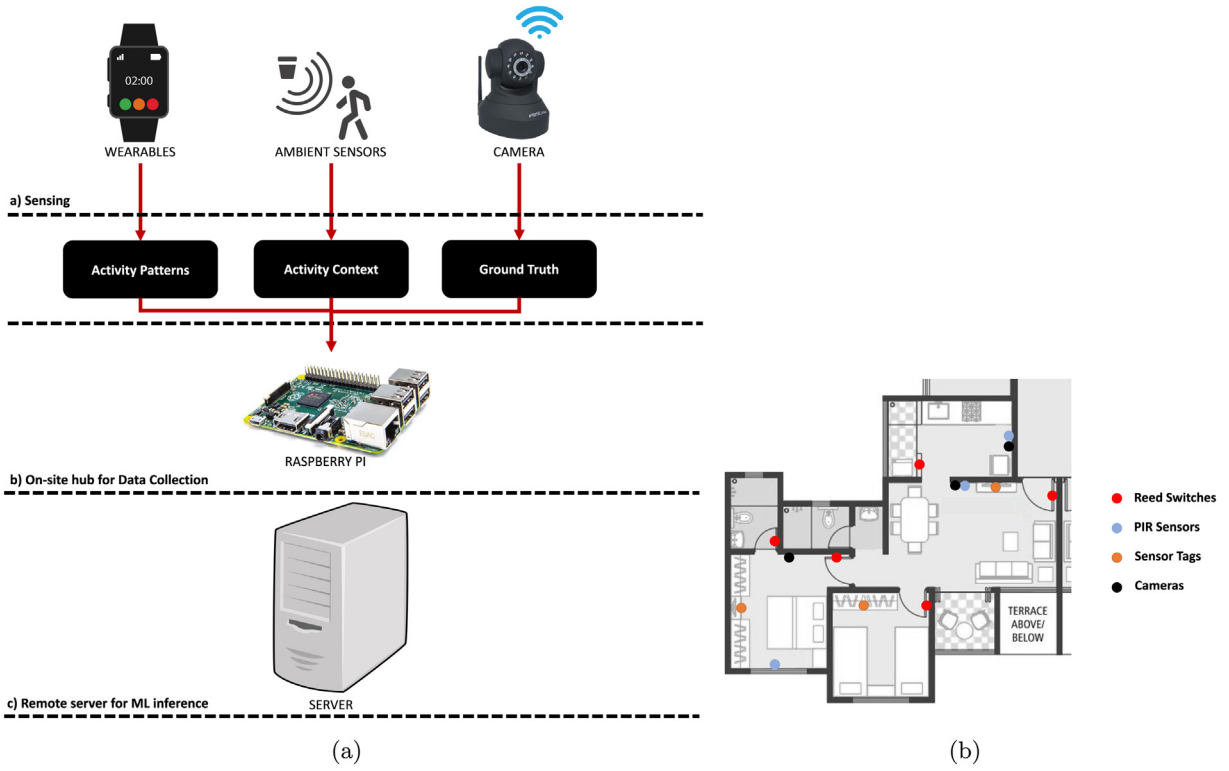



**Fig. 3.** (a) Sensor System Architecture Overview, (b) Ambient Sensors Layout.

## 5. Experimentation pipeline

The experiments were conducted on a Linux server consisting of i7-6850K with 4x NVIDIA GeForce GTX 1080Ti GPUs and 64 GB RAM. Data preprocessing, feature extraction, classification, deep learning techniques were implemented and visualized using python. The deep learning tasks were implemented using PyTorch libraries.

### *5.1. Annotations*

The raw accelerometer data was annotated by two different annotators by matching the time stamps on the wearable device and the captured video. The annotators provided the (start time, end time) of each activity block. We computed the intersection of (start time, end time) provided by both the annotators and used them to separate out labeled data. The remaining data (not considered in the labeled data/ data captured when no camera was present) were considered as unlabeled data in this work. The labeled data, and the unlabeled data were then subjected to preprocessing as discussed below. The set of labels chosen for the annotations are listed in Table 1. These activities were chosen as the labels for this study as these activities were supplemented with a survey-based assessment score using either Barthel or Lawton scales. When the participants remained idle, those data instances were labeled as "Idle" and were ignored from this study.

### *5.2. Preprocessing*

This study only considers the body-worn sensor data for analysis (only accelerometer for AAR dataset). Body-worn sensors are affected by motion artifacts. For instance, if the Empatica E4 is not tight enough, then the devices' slight movements on the wrist can cause motion artifacts. Motion artifacts are nothing but high-frequency noise. Hence, we employ a median filter of window size four, which acts as a low pass filter. We compared the effect of 2nd order Butterworth filter and Kalman filter and found the results comparable. Hence, we chose the median filter as it was computationally faster. Further, a windowing approach was employed to prepare the data of the AAR dataset. The body-worn sensors provide continuous time-series, and a patch of the series represents annotation an activity takes, thus requiring a windowing approach. A window size of 1 second with an overlap of 0.5 s was used for each axis and then concatenated. Additionally, we performed a feature extraction step after the windowing phase. We extracted discrete wavelet transform coefficients (using Daubechies 5(db5) scaling function) for each window and used that as the input data for the unsupervised feature learning phase.

### 5.3. Pipeline and model parameters

In the pre-training phase, we feed the unlabeled data (widowed (1 s) with overlap (0.5 s)) after wavelet transform to the feature extraction function that uses DBN to learn the representations. The raw input data instance vector's size was $(3 * 32)$, where 3 is the number of accelerometer channels and 32 is the window size. Besides, the wavelet coefficients remains with the same dimension. We stacked three RBMs with 90, 80, 70 neurons in each RBM to form DBN. We opted to use a decreasing number of neurons so that the representations are both sparse and with a reduced dimension. Following the pre-training phase using the unlabeled dataset, we preserved the network structure and froze the weights. The labeled dataset was then passed into the preserved pre-trained network to acquire the representations, making the data instance to a size of $(3 * 32 + 70)$ for the supervised classification task.

### 5.4. Evaluation strategy

To evaluate the effectiveness of the proposed methodology, we first visualize the representations learned from the DBNs for the labeled data. Further, we evaluate the performance of the classifiers using classification accuracy, precision, recall, f1-score, markedness and Informedness [41]. The choice of evaluation metrics relied on factors such as the number of classes in the classification problem and class imbalance in the AAR dataset. When a dataset consists of class imbalance (i.e., varying number of data instances in each activity class), the evaluation metric must equally represent the correct and missed detections. Often, the classifiers tend to be biased towards the majority class (more correct detections) while ignoring the minority class (more missed detections). Metrics such as accuracy, precision, and recall do not effectively represent the minor class's missed detections effectively. Thus, in this work, we rely on evaluation metrics such as Informedness and Markedness. Informedness tells us about how well the model has learned the correct and missed detections of classification while markedness tells us the trustworthiness of those correct and missed detections [41,42]. We also employ a 70-20-10 split for training, testing, and validation for shallow learning and deep learning approaches. The validation set for deep learning algorithms was used to tune the hyper-parameters and select the best model. The validation set and test set were truly held out data and were neither a part of the training phase nor the feature extraction phase. We have a massive set of unlabeled instances (50936470 instances) compared to 651844 labeled instances for the AAR dataset before windowing.

## 6. Evaluation

This section discusses the various analysis performed using the AAR dataset. Before analyzing the classification task, it is imperative to assess the quality of the representation of the labeled data in the new self-taught space (obtained due to pre-training). Thus, we leverage visualization techniques to assess the quality of the representations. As the representations are high-dimensional, visualizing them requires a dimensionality reduction step to convert the high-dimensional representation to a 2-dimensional matrix without losing information. We use Compression Variational Autoencoders (CVAE), a dimensionality reduction technique built on variational autoencoders with inverse autoregressive flow layers that makes the learned distribution more expressive [43]. Fig. 4 shows the CVAE visualization of the data at various stages of the *STAR-Lite* model for different numbers of activities (row-wise). The left, center, and right columns show the CVAE visualization of the raw data, fused data, and output of the final convolutional layer, respectively. The CVAE visualizations are color-coded with respect to the class labels. We observe that the raw data has the highest overlap between data instances belonging to different classes in the left column of Fig. 4. Also, as the number of classes increases, the separability in the data is not observed, which makes the classification task difficult. Therefore, we can infer that both the inter-class similarity and intra-class variations are high for the raw data. For the visualization of the fused data, we observe marginal clustering behavior, i.e., data instances belonging to the same classes are relatively closer, and those belonging to a different class are also closer to each other, suggesting that the intra-class variations have been minimized, even when the number of classes is increased. Finally, from the CVAE visualization of the output of the final convolution layer (right column), we observe clear clustered data instances. Here, both inter-class similarity and intra-class variations have been optimized, which helps with the downstream classification task. As the cluster formation holds true for an increasing number of activities, we can infer that *STAR-Lite* is invariant to the number of labels in the labeled data and can be considered scalable for an increasing number of activity labels.

In Fig. 5, we present the illustration of the self-similarity matrix for each of the activity labels. The self-similarity matrix is calculated by computing the cosine similarity of each data instance with every other data instance. Fig. 5a shows the self-similarity matrix visualization for the raw features, and Fig. 5b shows that of the self-taught features. We observe that the self-taught features have a higher number of data instances that are similar to other data instances belonging to the same class. Such observation also holds true for every activity class label considered in this study's labeled dataset. We also plotted the probability density functions of the obtained cosine similarities in Figs. 6a, b and observed that the self-taught features consisting of about 70% of the data instances have a cosine similarity value greater than 0.5 compared to 10% for the raw data. This observation helps us conclude that the self-taught features have fewer intra-class variations.

Now that we have obtained a good representation of the labeled data in a new representation space, we now aim to perform the classification task. First, we used some popular shallow learning algorithms such as k-NN, SVM, Random

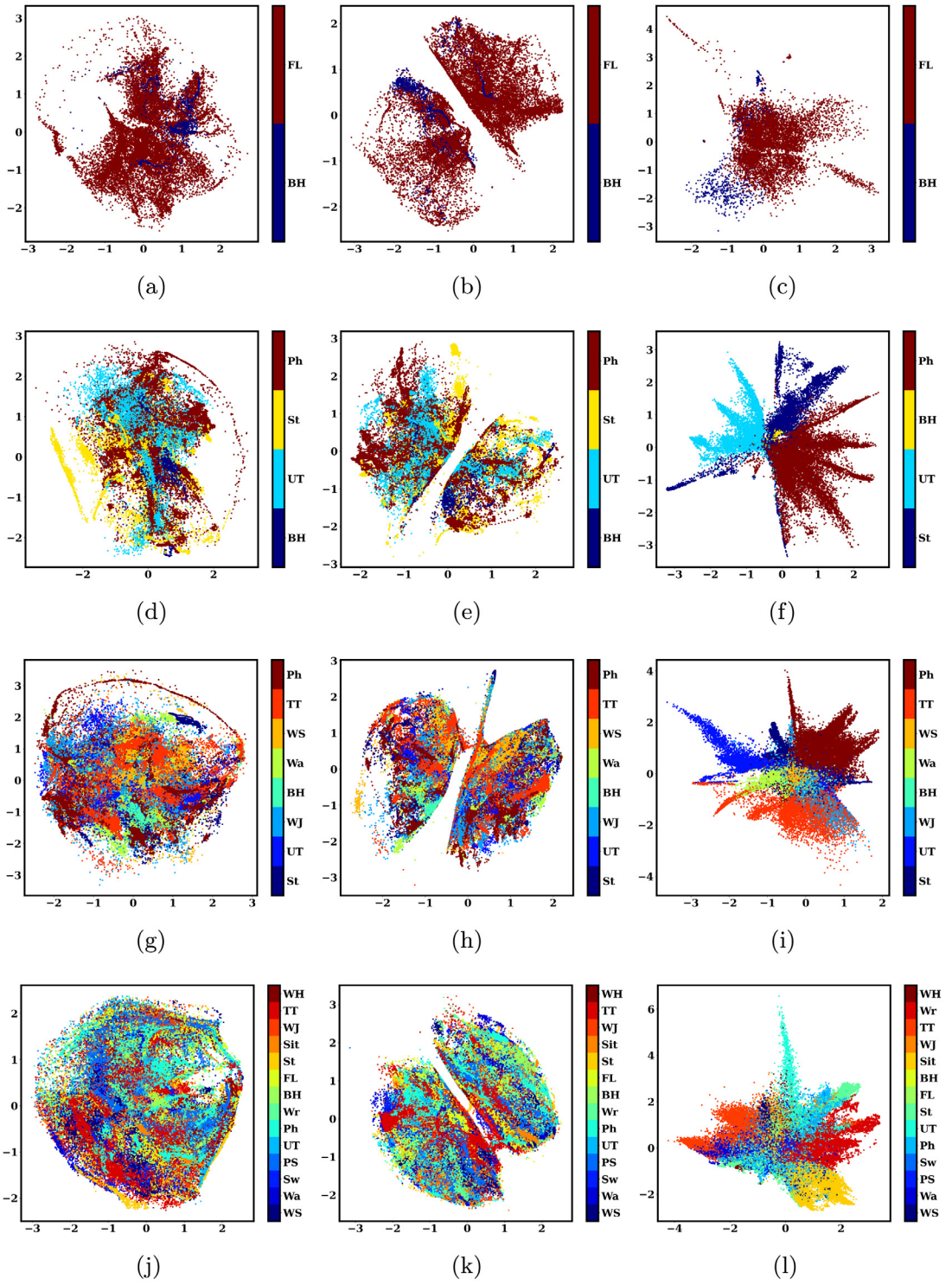



**Fig. 4.** CVAE Visualization of raw data (left column), fused data (center column) and self taught features (right column) for increasing number of activities (row-wise).

forest, and Multi-layer perceptron to perform the classification task. The notion behind this step was to check the efficacy of our pre-training and self-taught learning step for algorithms that highly depend on feature engineering. In contrast, we also investigated if we could extract more meaningful features from the self-taught features using deep learning

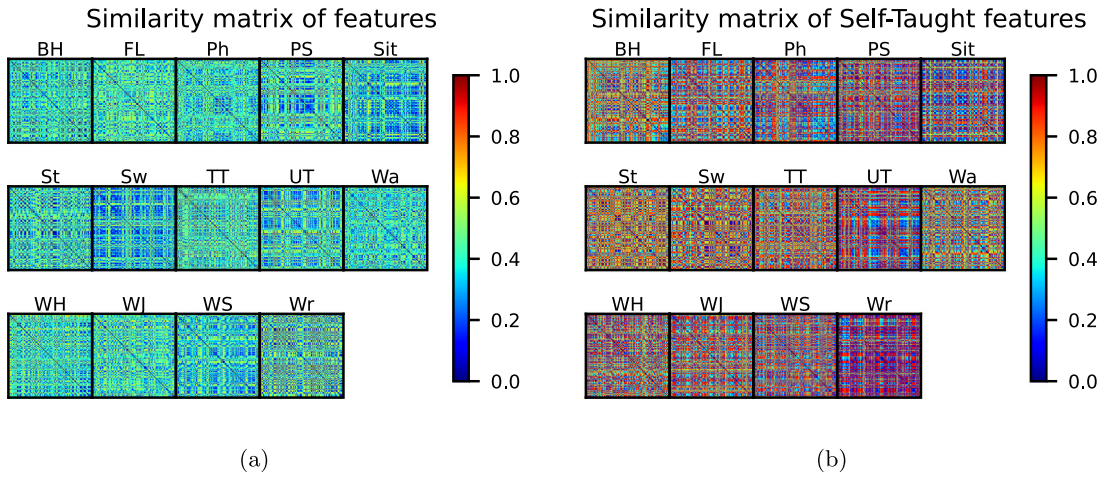



**Fig. 5.** Self-similarity matrix visualization of (a) raw features, (b) self-taught features.

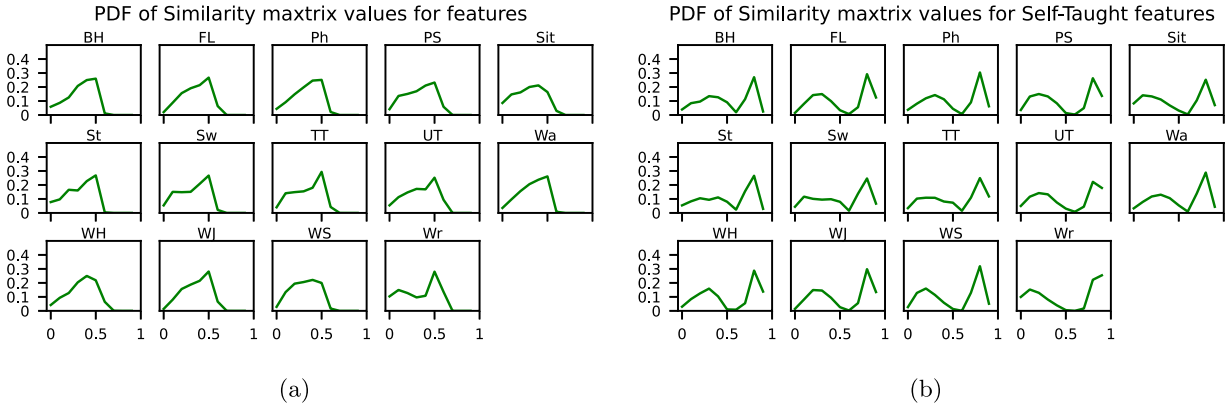



**Fig. 6.** Probability density function of self-similarity matrix for (a) raw features, (b) self-taught features.

**Table 2**
Hyper-parameters of CNN model.

| Hyper-parameters | Values |
|---|---|
| No. of convolution layers | 3 |
| No. of filters in convolution layers | 256, 256, 512 |
| Convolution filter dimensions | 5, 5, 5 |
| No. of maximum fully connected layers | 6 |
| No. of neurons in fully connected layers | 512, 256, 256, 128, 64, 14 |
| Batch size | 32 |

approaches such as Convolutional Neural Networks (CNN). We chose CNNs for the deep learning approach because the convolution operation makes the algorithm shift-invariant. In the AAR dataset, some activities may have been performed in 3 s, and other instances of the same activity may have been performed in 2 s. The shift-invariant property of convolution operation in CNN ensures (at various depths of convolutional layers) to gather this information and represent them as a higher-level representation for the classification task. Table 2 shows the hyper-parameters used to train the convolutional neural network. Besides, the classification metrics have been listed in Table 3. Comparing the shallow learning algorithms for the AAR dataset with/without self-taught features, we see an increase of 2% in both informedness and markedness metrics with a maximum of 44.56% for k-NN using the proposed pre-training methodology. Such a result ascertains the need to improve the feature representation of labeled data and supports our decision for a deep learning-based approach. Performing the same comparison with CNN-based classification, we noticed a 36% improvement in informedness and markedness metrics. Fig. 7 shows the learning curves of the CNN model comparing both with/without self-taught learning. For the AAR dataset, the model converges relatively quickly at ten epochs with higher accuracy when compared to the baseline CNN. Besides, Fig. 8 shows the confusion matrix for *STAR-Lite*. Interestingly, we notice that some of the

**Table 3**
Evaluation metrics: Comparison of shallow learning and deep learning for downstream classification task.

| Classifiers | Without Self-taught learning (in %) | | | | | Using Self-taught learning (in %) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | I | M | P | R | F | I | M |
| k-NN | 46.38 | 46.38 | 46.38 | 44.56 | 44.56 | 48.30 | 48.30 | 48.30 | 44.32 | 44.32 |
| SVM | 21.56 | 21.56 | 21.56 | 15.53 | 15.53 | 40.16 | 40.16 | 40.16 | 35.55 | 35.55 |
| RF | 21.91 | 21.91 | 21.91 | 15.91 | 15.91 | 23.82 | 23.82 | 23.82 | 17.96 | 17.96 |
| MLP | 20.62 | 20.62 | 20.62 | 14.51 | 14.51 | 36.66 | 36.66 | 36.66 | 31.78 | 31.78 |
| Baseline-CNN | 53.23 | 53.23 | 53.23 | 49.63 | 49.63 | | | | | |
| STAR-Lite | | | | | | **86.18** | **86.18** | **86.18** | **85.12** | **85.12** |
| SOTA [19] | | | | | | 69.53 | 69.53 | 69.53 | 68.09 | 68.11 |
| SOTA [20] | | | | | | 49.11 | 49.11 | 49.11 | 49.02 | 49.02 |

P: Precision, R: Recall, F1: F1-score, I: Informedness, M: Markedness

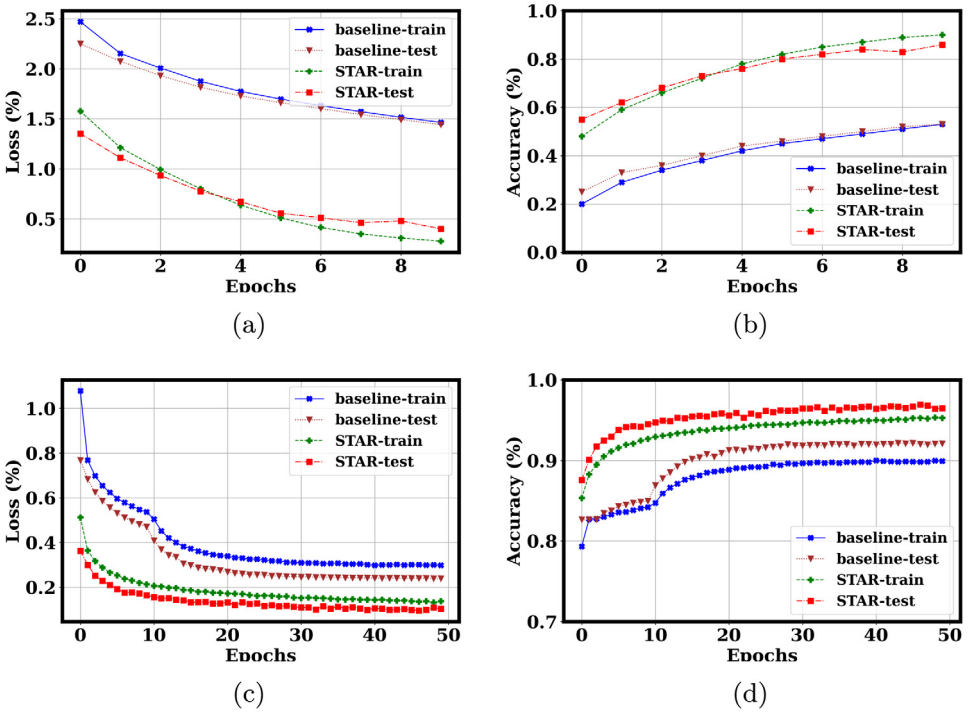



**Fig. 7.** AAR Dataset: (a) loss vs epoch, (b) accuracy vs epoch; OPPORTUNITY Dataset: (c) loss vs epoch, (d) accuracy vs epoch.

misclassifications are related to *Standing* activity. Activities such as *Wash Hands, Walking, Using Tooth Brush, Sweeping, Preparing Sandwich* have been misclassified as *Standing* (2 - 15 %). We believe the reason for this is because each of these activities was performed while standing. A part of these activities constitutes *Standing*, which makes it plausible for some windows labeled as *Wash Hands, Walking, Using Tooth Brush, Sweeping, Preparing Sandwich* activities could contain the pattern of *Standing*. In addition, *STAR-Lite* can successfully mitigate the practical challenges that arise during our data collection drive. For instance, let us consider some activities/sub-activities such as *brushing hair, picking up phone call, writing, wearing shoes*. Most of the time, the wearable device may not detect these activities based on its dominant/non-dominant hand position. Alternatively, as previously discussed, some of these activities may be performed while sitting. However, based on the confusion matrix (Fig. 8), we notice a tremendous improvement in the detection of such activities using the proposed methodology. We believe our approach can also help mitigate the shortcomings of such system-related and practical challenges as well.

### *6.1. Comparison with baseline*

To better highlight the efficacy of our approach, we perform two types of comparisons. First, we compare how the state-of-the-art algorithm performs on the AAR dataset. Secondly, we use a publicly available dataset to show how well the algorithm scales to different scenarios.

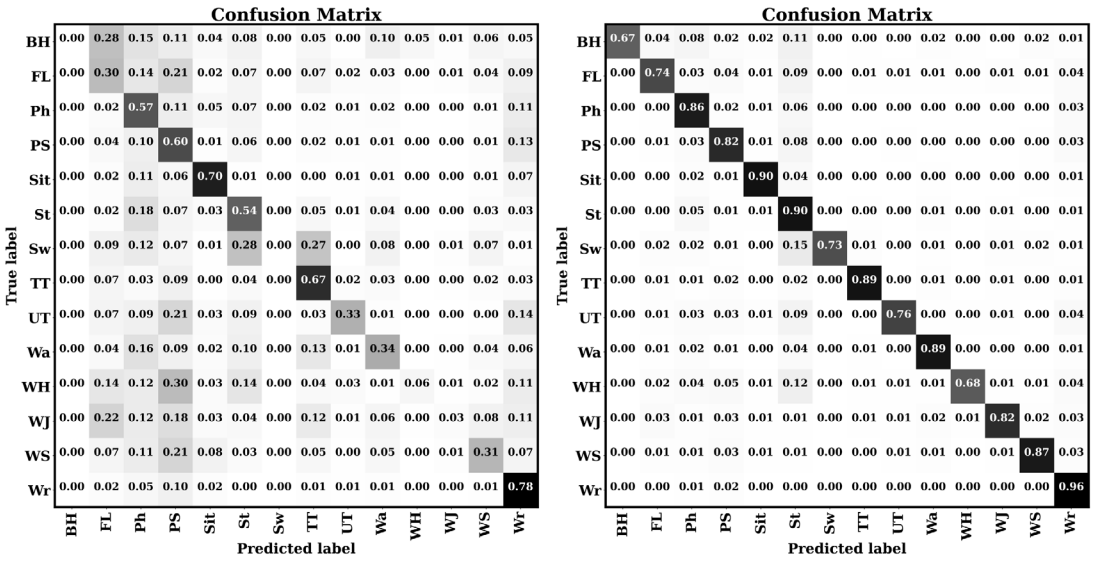



**Fig. 8.** AAR Dataset: Confusion matrix generated for (a)baseline, (b) *STAR-Lite* .

*6.1.1. State-of-the-art algorithms*

The state-of-the-art algorithm for a similar category of data includes a pre-training phase combined with a multi-task learning framework falling under the category of self-supervised learning [19](discussed in related work section). We evaluated the AAR dataset on this algorithm and found the informedness and markedness to be 68.09% and 68.11%. Comparing [19] with *STAR-Lite*, we see that *STAR-Lite* outperforms the state-of-the-art algorithm. The key difference between [19] and *STAR-Lite* lies in the pre-training phase where we use a generative energy-based model while compared to learning a focused variation or augmentation version of the data. Additionally, we also compare our work with other deep architectures known to perform well on activity recognition datasets such as LSTM [20]. With LSTM, we found the informedness and markedness to be 49.02% each with the AAR dataset, which suggests that our framework outperforms LSTM based networks.

*6.1.2. Publicly available dataset*

We chose OPPORTUNITY dataset [44,45] as the publicly available dataset for the analysis. The OPPORTUNITY dataset comprises both wearable sensors and ambient sensor data totaling 145 attributes. In this study, we limit the usage of 77 attributes corresponding to the worn body sensors placed at different locations on the body. The 77 attributes correspond to the 3D accelerometer and 3D inertial measurement data (3D acceleration, 3D rate of turn, 3D magnetic field, and orientation of the sensor), and the sensors were placed at various positions on the body for 4 participants. In this dataset, some high level and low-level activities such as Relaxing, coffee time, early morning, cleanup, sandwich time, unlock the door, stir, lock the door, close door, reach the door, open door, sip, clean, bite, cut, spread, release the door and move are considered. Data corresponding to 2 participants constituted the labeled data, and the remaining unlabeled data of the same participants and all the data of another 2 participants were considered unlabeled data. We also considered the null class data to be a part of the unlabeled data pool from which the features were derived during the pre-training phase. We performed the same experiments as the AAR dataset and reported the analysis below. We found a substantial increase in the accuracies for shallow learning algorithms, approx. 8% for SVM and approx. 7% using random forest. Besides, *STAR-Lite* shows a 6% boost in accuracy for deep learning. We believe that both shallow and deep learning algorithms showed improvement because of the following reasons: (1) the dataset was captured from 4 users only, (2) the dataset comprised of numerous features (77 attributes as discussed earlier) (3) the dataset was captured in a lab environment due to which there is less intra-class variability.

## 7. Resource constrained devices feasibility study

Finally, we evaluated *STAR-Lite* on the SenseBox testbed on a server for two major aspects: Model Size and Model Execution time. The inference was executed on a single core of Intel i7-6850K processor on the remote server of the SenseBox testbed. The model needs to be loaded onto the random access memory (RAM) for execution; it is imperative to benchmark the model size and execution time.

**Table 4**
Comparison of trained model sizes (in MBytes).

| | Original | Pruning | Static quantization | Dynamic quantization | Pruning + Dynamic quantization |
|---|---|---|---|---|---|
| Without self-taught learning | 7.9 | 6.9 | 7.81 | 3.35 | 6.7 |
| Using self-taught learning | 12.98 | 11.05 | 12.87 | 4.90 | 11.02 |

**Table 5**
Average model execution time (in milli-seconds) for Server (CPU).

| | Original | Pruning | Static quantization | Dynamic quantization | Pruning + Dynamic quantization |
|---|---|---|---|---|---|
| Without self-taught learning | 12.59 | 7.83 | 14.22 | 10.07 | 7.85 |
| Using self-taught learning | 17.95 | 12.82 | 17.06 | 16.00 | 14.42 |

**Table 6**
Average model execution time (in milli-seconds) for Jetson Nano.

| | Original | Pruning | Static quantization | Dynamic quantization | Pruning + Dynamic quantization |
|---|---|---|---|---|---|
| Without self-taught learning | 328.99 | 314.46 | 309.84 | 310.07 | 319.90 |
| Using self-taught learning | 399.43 | 381.58 | 378.34 | 396.04 | 393.66 |

### 7.1. Model size

The comparison of model sizes for original *STAR-Lite* model, pruned, quantized, and pruned + quantized models are listed in Table 4. Irrespective of the model compression technique used, we observe that the model size of the *STAR-Lite* is greater than the baseline model. This is because the *STAR-Lite* model has more features (raw features + self-taught features), which increases the number of model parameters, resulting in higher model size. Next, the pruned self-taught model size was 11.05 MBytes compared to the original 12.08 MBytes model and observed only a marginal improvement. The proposed layer-wise pruning technique assumes that the model performance remains the same. Thus, we believe that only the most unimportant and small network connections were pruned, resulting in a small decrease in the model size. Next, we observe that the model size remains almost the same for static quantization. This is because static quantization quantizes the model parameters and involves fusing activations with preceding layers. Therefore, no change in model size suggests that the fusing of the layers did not occur with static quantization. On the other hand, we observe a drastic decrease in model size (12.98 MBytes to 4.90 MBytes) for dynamic quantization while preserving the model performance. Finally, when we combine pruning and quantization, we observe that the model size is reduced by 1.96 MBytes, which is not drastic compared to dynamic quantization but quite modest compared to the base model. As pruning removes the connections in the network and quantization reduces the bitwidth of the network parameters, quantizing pruned networks requires a higher number of network parameters to retain model performance.

### 7.2. Model inference time

The model inference times for all the models are shown in Table 5. Similar to model sizes, we observe that the inference time of the *STAR-Lite* model irrespective of model compression used is higher compared to the baseline due to the higher number of computations involved with the increased feature size. In the case of model inference, both static and dynamic quantized model inference times were comparable to that of the baseline. On the other hand, the pruned model provided the best inference time by improving it by 28%. This improvement can be attributed to the pruned model reducing the number of computations resulting in faster inference times. In the pruned and quantized model, we observe a decrease of 3.53 ms, which is higher than the pruned model, as quantization prevented some network connections from being pruned to retain the model performance.

The model execution time observed while executing the various models on a Jetson Nano is shown in Table 6. We observe that the execution time is much higher compared to that of a server. This mammoth difference is expected due to the availability of enhanced resources such as CPU, RAM, and power available to a server. Similar to our observation of executing the models on the server, we observe a decrease in execution time after compressing the model using pruning, quantization, and pruning+quantization. However, the model execution time decrease is much prominent on resource-constrained devices (e.g., 20 s reduction after static quantization).

The above comparisons conclude that pruning the *STAR-Lite* model provided the best inference time on a server, while static quantization performs better on a resource-constrained Jetson Nano. On the other hand, dynamically quantizing the model provided the least model size. The choice of a specific compressed model can be determined based on the

requirements of the application scenario. For instance, detecting activities from older adults in real-time would require near-real detection, however, such a system can accept a small latency (around 0.5 s or 500 ms). In such a scenario, we could choose the dynamically quantized model for a resource-constrained device due to its smallest model size and relatively better execution time compared to the original model.

## 8. Discussion

### 8.1. Computational overhead of pre-training phase

The pre-training phase that involves learning the new representation space from unlabeled data is achieved at the cost of an additional computation overhead. As discussed in Section 3.1.3, the computational complexity was determined as $\mathcal{O}(i*j*k+exp(i+j+k))$ for the pre-training phase. However, during the inference stage, the computational complexity reduces to $\mathcal{O}(i*j*k)$ due to the absence of the weight updation step. Due to the greedy nature of the pre-training phase, the choice of the stopping criteria is critical in making the algorithm feasible. We defined a threshold of 0.05 for the reconstruction mean-squared loss for each RBM as the stopping criteria. Another aspect that allowed the pre-training phase feasible was the usage of GPU during training. The average time taken to train the DBN for a batch of unlabeled data on a GPU was 0.003 ms. During the inference on the CPU, the execution time to process a batch (batch size $-$ 32) of data was 0.065 ms.

### 8.2. Scaling STAR-Lite for new dataset/population

We believe that *STAR-Lite* can readily be scaled to a new dataset characterized by a population of a different age group, underlying health conditions, and so on. To adapt to a new dataset, labeled data from the new dataset will first be passed into the pre-trained (on an existing dataset) to learn the representations of the new dataset with respect to the label space of the existing dataset. As these representations inherently optimize inter-class and intra-class variations, exposing these representations to the supervised learning phase would provide us with an enhanced classification performance.

## 9. Conclusion

In this paper, we have demonstrated that the challenging problem of AR for older adults can be boosted using self-taught learning, especially leveraging abundant unlabeled data to aid the downstream classification task. We have presented self-taught learning-based activity recognition that has achieved approx. 36% increase in informedness and markedness; and outperforms the state-of-the-art research. We have also demonstrated that the proposed method, *STAR-Lite* is scalable to other populations as well by evaluating on a publicly available dataset obtained for a different population. We further explored various model compression techniques, performed a feasibility study of its adaptability in a real-life scenario, and obtained a 62% reduction in model size and a 28% reduction in average execution time. Besides, we have deployed our data collection system, *SenseBox*, in the real world (in a retirement community) and collected 25 participants' data envisioned to study the relationship between activities and underlying functional and behavioral health. We believe *STAR-Lite* could be scaled to other data sources, such as sports analytics, fitness applications where capturing labeled sensory data is challenging and acquiring abundant unlabeled data is plausible.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Nirmalya Roy reports financial support was provided by National Science Foundation. Nirmalya Roy reports financial support was provided by Alzheimer's Association.

## Data availability

The data that has been used is confidential.

## Acknowledgment

This research was supported by NSF CAREER Award #1750936 and Alzheimer's Association Grant #AARG-17-533039.

## References


[1] D. Cook, K.D. Feuz, N.C. Krishnan, Transfer learning for activity recognition: A survey, Knowl. Inf. Syst. 36 (3) (2013) 537–556.
[2] M. Alam, N. Roy, A. Misra, J. Taylor, CACE: Exploiting behavioral interactions for improved activity recognition in multi-inhabitant smart homes, in: ICDCS, IEEE, 2016, pp. 539–548.
[3] M. Alam, N. Roy, S. Holmes, A. Gangopadhyay, E. Galik, Automated functional and behavioral health assessment of older adults with dementia, in: CHASE, IEEE, 2016, pp. 140–149.
[4] M. Khan, R. Kukkapalli, P. Waradpande, S. Kulandaivel, N. Banerjee, N. Roy, R. Robucci, RAM: Radar-based activity monitor, in: INFOCOM 2016-the 35th Annual IEEE International Conference on Computer Communications, IEEE, IEEE, 2016, pp. 1–9.
[5] M. Khan, H. Hossain, N. Roy, Infrastructure-less occupancy detection and semantic localization in smart environments, in: 12th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, 2015, pp. 51–60.
[6] N. Pathak, M. Khan, N. Roy, Acoustic based appliance state identifications for fine-grained energy analytics, in: Pervasive Computing and Communications (PerCom), 2015 IEEE International Conference on, IEEE, 2015, pp. 63–70.
[7] J. Ma, H. Wang, D. Zhang, Y. Wang, Y. Wang, A survey on wi-fi based contactless activity recognition, in: 2016 Intl IEEE Conferences (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld), IEEE, 2016.
[8] J. Taylor, H.S. Hossain, M. Alam, M. Khan, N. Roy, E. Galik, A. Gangopadhyay, Sensebox: A low-cost smart home system, in: Pervasive Computing and Communications Workshops (PerCom Workshops), 2017 IEEE International Conference on, IEEE, 2017.
[9] D.J. Cook, A.S. Crandall, B.L. Thomas, N.C. Krishnan, CASAS: A smart home in a box, Computer 46 (7) (2012) 62–69.
[10] P.N. Dawadi, D.J. Cook, M. Schmitter-Edgecombe, Automated cognitive health assessment using smart home monitoring of complex tasks, IEEE Trans. Syst. Man Cybern.: Syst. 43 (6) (2013) 1302–1313.
[11] A.A. Aramendi, A. Weakley, A.A. Goenaga, M. Schmitter-Edgecombe, D.J. Cook, Automatic assessment of functional health decline in older adults based on smart home data, J. Biomed. Inform. 81 (2018) 119–130.
[12] S. Ramasamy Ramamurthy, N. Roy, Recent trends in machine learning for human activity recognition — A survey, Wiley Interdiscip. Rev.: Data Min. Knowl. Discov. (2018) e1254.
[13] R. Hadsell, S. Chopra, Y. LeCun, Dimensionality reduction by learning an invariant mapping, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR'06, 2, IEEE, 2006, pp. 1735–1742.
[14] K.Q. Weinberger, J. Blitzer, L.K. Saul, Distance metric learning for large margin nearest neighbor classification, in: Advances in Neural Information Processing Systems, 2006, pp. 1473–1480.
[15] W. Chen, X. Chen, J. Zhang, K. Huang, Beyond triplet loss: a deep quadruplet network for person re-identification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.
[16] K. Sohn, Improved deep metric learning with multi-class n-pair loss objective, in: Advances in Neural Information Processing Systems, 2016.
[17] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, 2020, arXiv preprint arXiv:2002.05709.
[18] S.R. Ramamurthy, I. Ghosh, A. Gangopadhyay, E. Galik, N. Roy, STAR: A scalable self-taught learning framework for older adults' activity recognition, in: 2021 IEEE International Conference on Smart Computing, SMARTCOMP, IEEE, 2021, pp. 121–128.
[19] A. Saeed, T. Ozcelebi, J. Lukkien, Multi-task self-supervised learning for human activity detection, Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 3 (2) (2019) https://doi.org/10.1145/3328932.
[20] Y. Zhao, R. Yang, G. Chevalier, X. Xu, Z. Zhang, Deep residual bidir-LSTM for human activity recognition using wearable sensors, Math. Probl. Eng. 2018 (2018).
[21] J. Wang, Y. Chen, S. Hao, X. Peng, L. Hu, Deep learning for sensor-based activity recognition: A survey, Pattern Recognit. Lett. 119 (2019) 3–11.
[22] F. Ordóñez, D. Roggen, Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition, Sensors 16 (1) (2016) 115.
[23] N.Y. Hammerla, S. Halloran, T. Plötz, Deep, convolutional, and recurrent models for human activity recognition using wearables, 2016, arXiv preprint arXiv:1604.08880.
[24] L. Zhang, X. Wu, D. Luo, Recognizing human activities from raw accelerometer data using deep neural networks, in: 2015 IEEE 14th International Conference on Machine Learning and Applications, ICMLA, IEEE, 2015, pp. 865–870.
[25] S. Bhattacharya, P. Nurmi, N. Hammerla, T. Plötz, Using unlabeled data in a sparse-coding framework for human activity recognition, Pervasive Mob. Comput. 15 (2014) 242–262.
[26] K.D. Feuz, D.J. Cook, Collegial activity learning between heterogeneous sensors, Knowl. Inf. Syst. 53 (2) (2017) 337–364.
[27] R. Fallahzadeh, H. Ghasemzadeh, Personalization without user interruption: boosting activity recognition in new subjects using unlabeled data, in: Proceedings of the 8th International Conference on Cyber-Physical Systems, ACM, 2017, pp. 293–302.
[28] M. Khan, N. Roy, Untran: Recognizing unseen activities with unlabeled data using transfer learning, in: Internet-of-Things Design and Implementation (IoTDI), 2018 IEEE/ACM Third International Conference on, IEEE, 2018, pp. 37–47.
[29] R. Raina, A. Battle, H. Lee, B. Packer, A.Y. Ng, Self-taught learning: Transfer learning from unlabeled data, in: Proceedings of the 24th International Conference on Machine Learning, ICML '07, ACM, New York, NY, USA, 2007.
[30] R. Grosse, R. Raina, H. Kwong, A.Y. Ng, Shift-invariance sparse coding for audio classification, 2012, arXiv preprint arXiv:1206.5241.
[31] H. Lee, P. Pham, Y. Largman, A.Y. Ng, Unsupervised feature learning for audio classification using convolutional deep belief networks, in: Advances in Neural Information Processing Systems, 2009.
[32] P. He, P. Jia, S. Qiao, S. Duan, Self-taught learning based on sparse autoencoder for E-nose in wound infection detection, Sensors 17 (10) (2017) 2279.
[33] J. Gan, L. Li, Y. Zhai, Y. Liu, Deep self-taught learning for facial beauty prediction, Neurocomputing 144 (2014) 295–303.
[34] O. Amft, Self-taught learning for activity spotting in on-body motion sensor data, in: 2011 15th Annual International Symposium on Wearable Computers, IEEE, 2011, pp. 83–86.
[35] Y. Cheng, D. Wang, P. Zhou, T. Zhang, A survey of model compression and acceleration for deep neural networks, 2017, arXiv preprint arXiv:1710.09282.
[36] Y. Gong, L. Liu, M. Yang, L. Bourdev, Compressing deep convolutional networks using vector quantization, 2014, arXiv preprint arXiv:1412.6115.
[37] S.A. Tailor, J. Fernandez-Marques, N.D. Lane, Degree-quant: Quantization-aware training for graph neural networks, 2020, arXiv preprint arXiv:2008.05000.
[38] I. Lazarevich, A. Kozlov, N. Malinin, Post-training deep neural network pruning via layer-wise calibration, 2021, arXiv preprint arXiv:2104.15023.
[39] B. Li, B. Wu, J. Su, G. Wang, Eagleeye: Fast sub-net evaluation for efficient neural network pruning, in: European Conference on Computer Vision, Springer, 2020, pp. 639–654.
[40] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (2006) 1527–1554.
[41] D.M. Powers, Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation, 2020, arXiv preprint arXiv:2010.16061.

[42] H. Hossain, M. Al Haiz Khan, N. Roy, Deactive: Scaling activity recognition with active deep learning, in: Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, ACM, 2018.
[43] D.P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, M. Welling, Improved variational inference with inverse autoregressive flow, Adv. Neural Inf. Process. Syst. 29 (2016) 4743–4751.
[44] D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, A. Ferscha, et al., Collecting complex activity datasets in highly rich networked sensor environments, in: Networked Sensing Systems (INSS), 2010 Seventh International Conference on, IEEE, 2010, pp. 233–240.
[45] R. Chavarriaga, H. Sagha, A. Calatroni, S.T. Digumarti, G. Tröster, J.d.R. Millán, D. Roggen, The opportunity challenge: A benchmark database for on-body sensor-based activity recognition, Pattern Recognit. Lett. 34 (15) (2013) 2033–2042.