THE ROLE OF REPRESENTATIONS IN HUMAN ACTIVITY RECOGNITION

A Dissertation
Presented to
The Academic Faculty

By

Harish Kashyap Haresamudram

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

May 2019

THE ROLE OF REPRESENTATIONS IN HUMAN ACTIVITY RECOGNITION

Approved by:

Dr. Thomas Ploetz
College of Computing
Georgia Institute of Technology

Dr. David V. Anderson
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Irfan Essa
College of Computing
Georgia Institute of Technology

Dr. Patricio Antonio Vela
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: April 26, 2019

The greatest challenge to any thinker is stating the problem in a way that will allow a solution

Bertrand Russell

For my family, who have helped me in all things great and small.

ACKNOWLEDGEMENTS

As I have worked on the Master‘s thesis, I have received a great deal of support and assistance. First, I would like to thank my advisor Prof. Thomas Plötz. Without Thomas, I would have never started working on machine learning for wearable devices. His structured approach brought focus to my research and allowed me to pursue lines of questioning in a disciplined way. I would also like to thank other members of the Computational Behavior Analysis (CBA) Lab for their helpful discussions and invaluable feedback. Next I would like to thank Dr. David Anderson, my co-advisor for his guidance and our many chats discussing works of fantasy fiction.

A big thank you to Varun Agrawal for his help with PyTorch, and anything implementation related. I would like to thank Andreas Geist for our philosophical discussions, technical debates and indispensable feedback. Many thanks to Apoorva Beedu, for the technical discussions and endless support, without which it would not have been possible to complete the dissertation.

Finally, I would like to thank my parents and my sister for encouraging me in my pursuits, and inspiring me to follow my dreams. It would not have been possible for me to pursue a Master‘s degree without your support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

We investigate the role of representations in sensor based human activity recognition (HAR). In particular, we develop convolutional and recurrent autoencoder architectures for feature learning and compare their performance to a distribution-based representation as well as a supervised deep learning representation based on the DeepConvLSTM architecture. This is motivated by the promises deep learning methods offer – they learn end-to-end, eliminate the necessity for hand crafting features and generalize well across tasks and datasets. The choice of studying unsupervised learning methods is motivated by the fact that they afford the possibility of learning meaningful representations without the need for labeled data. Such representations allow for leveraging large, unlabeled datasets for performing feature and transfer learning.

The study is performed on five datasets which are diverse in terms of the number of subjects, activities, and settings. The analysis is performed from a wearables standpoint, considering factors such as memory footprint, the effect of dimensionality, and computation time. We find that the convolutional and recurrent autoencoder based representations outperform the distribution-based representation on all datasets. Additionally, we conclude that autoencoder based representations offer comparable performance to supervised DeepConvLSTM based representation.

On the larger datasets with multiple sensors such as Opportunity and PAMAP2, the convolutional and recurrent autoencoder based representations are observed to be highly effective. Resource-constrained scenarios justify the utilization of the distribution-based representation, which has low computational costs and memory requirements. Finally, when the number of sensors is low, we observe that the vanilla autoencoder based representations produce good performance.

CHAPTER 1
INTRODUCTION

Human activity recognition (HAR) – which involves the automated inference of what people do and when – constitutes a central aspect of wearable computing. Since its inception, a multitude of sensing modalities have been explored and are used to capture human activities. Essentially, HAR utilizes body-worn sensors to record movement data, which is then analyzed through a combination of signal processing and machine learning techniques. The goal, is to recognize – i. e. segment and classify – either finite sets of activities of interest, or to study them in an open-ended manner.

In [1], the Activity Recognition Chain (ARC) is defined as a "sequence of signal processing, pattern recognition and machine learning techniques that implements a specific activity recognition system behavior". Traditionally, HAR using wearables has been based on (variants of) the ARC, which describes a processing pipeline from sensory data to the classification of portions (segments) of subsequent readings into activities of interest (or the null class).

In recent years, end-to-end learning approaches have been adopted for HAR using wearables, primarily due to their promise of integrated learning – which effectively eliminates manual crafting and tuning of suitable data representations. This advantage, coupled with astonishing classification capabilities and transfer learning has made end-to-end learning models the de facto approach studied in the last few years. Although end-to-end learning models have outperformed the conventional ARC based approaches, the performance boost comes with the price of requiring substantial computational resources (even though considerable progress has been made in the efficient deployment of deep learning models on

resource-constrained devices [2, 3, 4, 5]), and considerable training datasets of **annotated** sample data .

Unlike other domains where a clear understanding exists on how sensor data is best represented (e.g., speech – mel frequency cepstral coefficients (MFCC) or log-mel spectrograms, vision – image edges), no clear consensus exists regarding the *gold standard* feature representation for HAR using wearables. In general, the representations for HAR take the following forms:

- Statistical features - these include a host of metrics such as the mean and standard deviation. However, statistical features do not carry any domain related knowledge but rather aim for generic representations; often requiring tweaking across tasks and domains [6].
- Representations focusing on specific aspects of the underlying signals, such as spectral methods and Fourier analysis - these methods apply signal processing techniques to wearable data. However, they face generalization issues [6, 7].
- Distribution-based approaches - completely abstract away from the application domain and instead focus on compact signal representation, thereby minimizing reconstruction loss; have universal applicability [8].
- Learned features - some approaches utilize the raw sensor data itself as input. These approaches include deep learning based methods (both supervised and unsupervised), and dimensionality reduction techniques like principal component analysis (PCA) [6, 9, 10].

In light of the advent of end-to-end deep learning approaches, the central question this work seeks to answer is – *What role do the representations play in human activity recognition?* In order to answer this, we study representations on a series of datasets which are

diverse in terms of the number of subjects, actions performed, goals, settings, and number of samples.

We develop convolutional and recurrent autoencoder models for application as feature learners in HAR. These unsupervised representations are contrasted against a distribution-based representation as well as a supervised representation based on the DeepConvLSTM architecture [11]. We evaluate the representations from a wearables standpoint and consider factors like memory footprint, computation time, and number of trainable parameters. From our analysis, we observe the efficacy of convolutional and recurrent autoencoder based representations for the larger datasets with multiple sensors. In resource-constrained scenarios, the distribution-based representation provides a more advantageous option due to the low computational requirements.

Our contributions are the following:

- We study the task of human activity recognition from a representation learning perspective.
- The experiments are performed on diverse datasets and feature types.
- To our knowledge, this is the first work that utilizes convolutional and recurrent autoencoders for representation learning in human activity recognition.

CHAPTER 2
RELATED WORK

Activity recognition is a time-series problem, in which the task is generally to segment out and classify contiguous portions of sensory data. Predominantly, segmentation is achieved by the use of the sliding window technique, which involves a shifting window of fixed length moved over time, for the extraction of 'frames' of interest. These frames usually have some degree of overlap (generally around 50%), but are considered to be independent.

The keys to performing successful HAR (as they are for any pattern recognition task), are: (i) appropriate feature representations of the sensor data; (ii) suitable classifiers. While the study of classifiers has intensified over the last few years, little systematic research has been performed towards the design of suitable feature representations; and lesser still is understood about the role of the representations in HAR. This lack of systematic research has been identified as one of the major shortcomings of the current systems in activity recognition [12].

The investigation and understanding of the sensor data's properties is key to find a representation that directly captures its core characteristics. In HAR, there is no all-encompassing model that affords the expert-driven design of a universal feature representation [13]. However, recent developments in machine learning, especially deep learning have the potential of overcoming this shortcoming by automatically *learning* relevant feature representations for sensor data.

In [14], the authors explore the design of representations that would allow the possibility of robust recognition in HAR. However, no feature extraction technique has been

established, that tackles this problem by providing a well-motivated representation for human movement for tasks where prior (or expert) knowledge is unavailable [13].

The authors in [14] distinguish two domains for the preprocessing step in AR, namely the time domain and the frequency domain. In the time domain, the most widely used feature extraction scheme computes statistical metrics directly from the raw sensor data, for every frame in the dataset. These metrics include the mean, standard deviation, entropy, and correlation coefficient. In the frequency domain, the feature extraction usually involves Fourier analysis on the frames. An experimental evaluation of the statistical and frequency domain features was performed in [7], who conclude that the Fourier coefficients represent the sensor data better than its statistical features.

Another approach to feature extraction has been the use of time-delay embeddings for activity and gait recognition [15]. This approach has shown good results in the analysis of repetitive activities. However, the features obtained from this technique are not appropriate for non-periodic activities. An alternate approach has been the use of discrete domain features to compute distance measures on the string representations of the sensor data [16]. However, such discretization removes the detailed information present in sensor data; which is required for robust performance of activity recognition models [13]. Apart from these, classic dimensionality reduction techniques such as PCA have been used in HAR. In [6], the performance of representations based on statistical metrics, Fourier coefficients and PCA are compared on four datasets. The performance of the Fourier coefficients, and PCA are similar on most of the datasets studied while the statistical features perform worse.

Distribution based approaches comprise the state-of-the-art for feature extraction in HAR. Introduced in [8], they are based on the empirical cumulative distribution function (ECDF) of the data within the frame. These representations have shown excellent perfor-

mance on a variety of datasets. In [8], the performance of these features is evaluated on six datasets and they outperform representations based on statistical metrics, Fourier coefficients, and PCA. Additionally, they incur little computational costs for their extraction. Further analysis is performed on the distribution based representations in [17], where the focus is on identifying the optimum window lengths for various activities across different datasets, while [18] adds structural characteristics to these representations.

In contrast to heuristic feature design (or handcrafting of features), feature learning optimizes an objective function to learn the features, in lieu of requiring domain expertise. Over the last few years, machine learning, especially deep learning, has shown great promise towards extracting valuable feature representations from large amounts of data. This includes both supervised models, which require annotations, and unsupervised models that do not require annotations.

There has been a lot of interest towards developing supervised, end-to-end, deep learning models for HAR. In [19], CNNs are developed for multichannel time series data. Meanwhile, the authors in [20] design a convolutional neural network (CNN) which outperforms other representations based on distributions, PCA, statistical metrics, and fully connected neural networks. The evaluation is performed on three datasets, and further experimentation is performed to identify the optimum architecture. To utilize the time-series nature of the sensor data, ensembles of recurrent neural networks (RNN) have been used in [21] for performing HAR. The application of RNNs for HAR is extended in [22], where continuous attention mechanisms over the sensory channels as well as time are developed to improve the performance. DeepConvLSTM [11] uses a combination of convolutional and recurrent layers. Improvements to the DeepConvLSTM have been proposed in [23], where an attention mechanism is added on the long short-term memory (LSTM) network to determine the 'important' time steps. While the supervised models provide excellent performance, they

come with the cost of requiring large amounts of *annotated data* to perform learning. This is both time and cost intensive, as it requires domain expert knowledge.

On the unsupervised learning front, deep networks have the capability to learn useful representations without utilizing class labels [24]. These methods assume that the characteristics of training data can be discovered by learning 'how' to generate data, and that a subset of the characteristics are then suitable to differentiate between classes [25]. Deep networks have also been applied to tasks in other domains, such as learning unsupervised representations for video classification [26] and audio scene classification [27], and obtaining vector representations for words from a corpus [28].

The focus of unsupervised deep learning methods in HAR has been towards developing Restricted Boltzmann Machines (RBM) and stacked autoencoders for learning representations. In [6], Restricted Boltzmann Machines were compared to PCA for feature learning, while [29] utilize a sparse coding framework to learn features on unlabeled data. State assessment of Parkinson's Disease in naturalistic environments is performed using RBMs in [9]. Both [30] and [10] utilize stacked, vanilla autoencoders to learn features.

While vanilla autoencoders have been evaluated for their feature learning, little work exists utilizing the more powerful variants of autoencoders which involve convolutional and recurrent layers. These layers can take advantage of the inherent time-series nature of sensor data to learn richer representations. This leads us to study their effectiveness as unsupervised feature learners for HAR.

CHAPTER 3

DATASETS

In this chapter, we detail the datasets used in this study. Since the goal is to understand the *role* of the representations in HAR, we choose a wide variety of datasets. Five datasets are studied, where the number of subjects ranges from 1 to 14, and the activities vary from opening/closing doors to walking/jumping to detecting freezing of gait in patients with Parkinson's Disease (PD). Such diversity aims to capture the scope of applicability of feature representations for different activities and tasks. A summary of the datasets is detailed in Table 3.1.

Opportunity

Opportunity is a benchmark dataset for human activity recognition using bodyworn sensors which contains annotated recordings from four participants [31]. The data has been collected for every-day life domestic activities especially focusing on kitchen routine. Inertial Measurement Units (IMU) attached to 12 on-body positions, sampled at 30Hz were used to collect the data. The annotations are provided for 18 mid-level activities such as Open Door / Close Door. Each participant performed five different runs of kitchen activities.

For the training and evaluation, we utilize the same protocol as [32]. Accelerometer recordings from the upper limbs, back and complete IMU data from both feet were used, resulting in a 77 dimensional dataset. The data is normalized to have zero mean and unit variance. As per the protocol, the second run from participant 1 is used as the validation set, while runs 4 and 5 from participants 2 and 3 are used as the test set. The rest of the data is used for training (approximately 700k samples). We utilize the sliding window approach to obtain frames of 1 second, with 50% second overlap.

Skoda

The Skoda dataset has been recorded in a manufacturing scenario and covers the problem of recognizing the activities of assembly-line workers in a car production environment [33]. A worker wore a plethora of sensors while undertaking manual quality checks for the correct assembly of parts in newly constructed cars. These checks involve 10 manipulative gestures of interest, such as checking the boot, opening/closing the bonnet, boot and doors, and turning the steering wheel. The accelerometer data is downsampled to 33Hz and normalized to have zero mean and unit variance. The data is recorded using 10 accelerometers on the right arm of the worker, resulting in 60 dimensional data. The training set consists of the first 80% of each class, followed by the validation and test sets taking up a remaining 10% each. The sliding window approach is utilized to obtain frames of 1 second and 50% overlap.

PAMAP2

PAMAP2 is a dataset recorded under scripted, and hence rather constrained conditions where the participants were instructed to carry out a total of 12 activities of daily living such as domestic activities, and various sportive exercises (nordic walking, running, etc) [34]. Full IMU data along with temperature and heart rate were recorded using devices attached to the chest, hand and ankle. Over 10 hours of data were collected, with the resulting dataset having 52 dimensions.

We downsample the data to 33Hz and normalize it to have zero mean and unit variance. Replicating the protocol from [32], we used runs 1 and 2 from participant 5 for validation and runs 1 and 2 from participant 6 for testing. The remaining data is used for training. As for the previous datasets, the sliding window approach is utilized to obtain frames of 1 second and 50% overlap.

USC-HAD

The USC-HAD dataset [35] was collected on the MotionNode sensing platform and consists of data from 14 subjects. The sensors include three-axis accelerometers and gyroscopes resulting in six channels. Particular emphasis was placed to ensure divergence in gender, age, height and weight of subjects. The activities correspond to the most basic and common human activities in daily lives. Twelve activities were recorded and they include various walking motions, jumping, sitting, etc. Seven male and seven female subjects participated in the data collection, with age between $21 - 49$ years (30.1 ± 7.2 years), height between $160 - 185$cm (170 ± 6.8) and weight between $43 - 80$kg (64.6 ± 12.1kg).

Dataset	Sensors	Partic -ipants	Monitored activities
Opportunity	Accele -rometer	4	Null, open door 1, open door 2, close door 1, close door 2, open fridge, close fridge, open dishwasher, close dishwasher, open drawer 1, close drawer 1, open drawer 2, close drawer 2, open drawer 3, close drawer 3, clean table, drink from cup, toggle switch
Skoda	Accele -rometer	1	Null, write on pad, open hood, close hood, check gaps, open left front door, close left front door, close both left door, check trunk gaps
PAMAP2	Accele -rometer	9	Lying, sitting, standing, walking, running, cycling, nordic walking, ascending stairs, descending stairs, vacuum cleaning, ironing, rope jumping
USC-HAD	Accele -rometer, Gyroscope, Magne -tometer	14	Walk forward, walk left, walk right, walk upstairs, walk downstairs, run forward, jump, sit on a chair, stand, sleep, elevator up, elevator down
Daphnet FoG	Accele- rometer	8	no movement, trembling in place, shuffling forward

Table 3.1: Summary of the datasets used in this study

Daphnet Freezing of Gait Dataset

The Daphnet Freezing of Gait Dataset is a publicly available dataset which contains data from eight subjects with Parkinson's Disease (PD), who experience regular Freezing of Gait (FoG) in daily life[36]. FoG is a common gait impairment defined as a "brief, episodic absence or marked reduction of forward progression of the feet despite the intention to walk" [37].

The data was recorded using three 3D accelerometers attached to the shank, the thigh and the lower back of the subjects. The collection was performed in the lab with an emphasis on generating many freeze events. The sessions were $20 - 30$ minutes each, and consisted of three walking tasks: straight line walking, walking with numerous turns and a more realistic activity of daily living (ADL), where users went into different rooms while performing tasks such as fetching coffee, opening doors, etc. More than eight hours of data were recorded in which professional physiotherapists identified 237 FoG events in a post hoc video analysis . The duration of the events varies between 0.5s and 40.5s (7.3 ± 6.7s) [36].

CHAPTER 4

METHODOLOGY

Pipeline

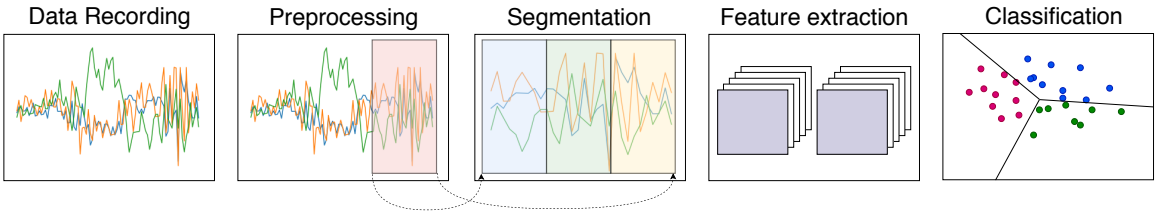


Figure 4.1: The Activity Recognition Chain (ARC).

In [1], the Action Recognition Chain is defined as a "sequence of signal processing, pattern recognition, and machine learning techniques that implements a specific activity recognition system behavior". The sequence is shown in Figure 4.1, and comprises of five steps.

- Data recording - measurements of one or more sensors are captured for a period of time.
- Preprocessing - the impact of sensor noise, artifacts and the environment are reduced using data transformations.
- Segmentation - after preprocessing, the signal is split into short segments (called frames), that are likely to contain activities or movements of interest.
- Feature extraction - the raw sensor data is used to extract useful representations so that activities of interest may be differentiated with more ease.
- Classification - a probability is assigned to each activity class of interest, thereby producing a hypothesis for each extracted segment.

The pipeline defines a series of discrete steps, where each step has a clearly defined goal. However, substantial manual optimizations are necessary for each part of the pipeline – with known issues such as poor generalization, thereby causing the need for specialized domain knowledge which hinders widespread adoption. On the other hand, this process can be heavily optimized with respect to computational resources (such as memory and processing power). In this manner, the pipeline can be deployed with ease *on* wearables themselves, without the need for offloading the computation to external servers over the internet.

In this study, we focus on the fourth step of the pipeline - Feature Extraction. A number of previous works exist - which utilize statistical features, data-driven representations like PCA or distribution-based representations. Papers such as [6] and [30, 10] study deep-learning based RBMs and stacked autoencoders. However, little analysis has been done towards utilizing more powerful variants of autoencoders - ones which have convolutional and recurrent layers. While there exist many supervised models such as DeepConvLSTM (which utilizes convolutional and recurrent layers to capture the temporal aspect of data), there has been no exploration to utilize these layers towards learning unsupervised feature representations. The lack of work on unsupervised learning in HAR, along with their ability to learn from unlabeled data sets motivates us to study their effectiveness as feature learners for HAR.

We study three types of autoencoders – vanilla, convolutional and recurrent. In [30] and [10], vanilla (or stacked) autoencoders have already been studied. In order to capture the temporal aspect of the sensor data, we study recurrent autoencoders. On the other hand, the spatio-temporal aspect of data is exploited using convolutional autoencoders.

The performance of these autoencoder-based unsupervised feature representations is

contrasted against a distribution-based representation, and a representation based on the widely used supervised classifier – DeepConvLSTM [11]. The rest of the chapter details the representations studied – distribution-based representation, autoencoder-based unsupervised representations, and DeepConvLSTM-based supervised representation – along with the common backend classifier and the core metric used in this study.

Distribution-based representations

Since the sensor data in HAR is time-series, the samples are correlated to their neighbors. Distribution-based representations take advantage of these correlations by computing the empirical cumulative distribution (ECDF) of the data in each frame. At the heart of ECDF lies the idea to extract a fixed set of real-valued coefficients that best represents the underlying distribution for each degree of freedom within a frame (i. e. each sensing axis of the accelerometer data) [8].

The ECDF representation f_i for a degree of freedom of analysis frame i is obtained by first estimating the ECDF P_c^i, given by

$$P_c^i = P(X \leq x) \tag{4.1}$$

The distribution is quantified by selecting d equally spaced and monotonically increasing points $C = p_1 ... p_d$ between 0 and 1. For each of those points, the value x_k is estimated, for which $P_c^i(x) = p_k$

$$C = p_i \in \mathbb{R}^d_{[0,1]}, p_i < p_{i+1} \tag{4.2}$$

$$f_i = x, \exists j : P_c^i(x) = p_j \tag{4.3}$$

where cubic interpolation is used as necessary to obtain each x. The new representation for each analysis frame i then corresponds to the concatenated ECDF representations of the individual degrees of freedom. In effect, this process provides an estimate for the *quantile function* for each of the selected points in C. The d dimensional representation f_i fully covers the spatial position of a distribution, as well as its overall shape. The only parameter that can be tuned is the number of points at which the inverse of P_c is interpolated. This parameter controls the granularity with which the shape of P_c is captured in the resulting representation [8].

Autoencoder-based unsupervised representations

An autoencoder is an unsupervised neural network that is trained to reconstruct the input after being passed through a series of layers. Internally, an autoencoder has a hidden layer h, that performs linear as well as non-linear function operations on the layers input data to obtain a 'latent representation' of the data. We can view the network as consisting of two parts – an encoder function $h = f(x)$ which transforms the input data x, and a decoder that produces the reconstruction $r = g(h)$. The network is restricted, e. g. h has smaller dimensions than x, thereby creating an *intentional bottleneck*. Thus, an autoencoder is forced to prioritize some aspects of the input data that need to be copied and thereby learns useful representations of the data [38].

The learning process is described as minimizing the loss function -

$$L(x, g(f(x))) \tag{4.4}$$

where L is a loss function that penalizes $g(f(x))$ for being dissimilar from x. Typically, mean squared error (MSE) is used as the loss function and h is used as the latent representation (or the bottleneck feature) for tasks such as classification and clustering.

An autoencoder is said to be 'undercomplete' when its hidden dimension is less than the input dimension. Learning an undercomplete representations forces the network to capture the most salient features of the training data. If the decoder is linear and the loss function is MSE, then the autoencoder learns to span the same space as the input datas principal component analysis (PCA). Thus, the autoencoder trained to perform the copying (or reconstruction) task has learned the principal subspace of the training data as a side effect. The utilization of non-linear activations in the network results in more powerful generalizations of PCA [38].

Vanilla autoencoders

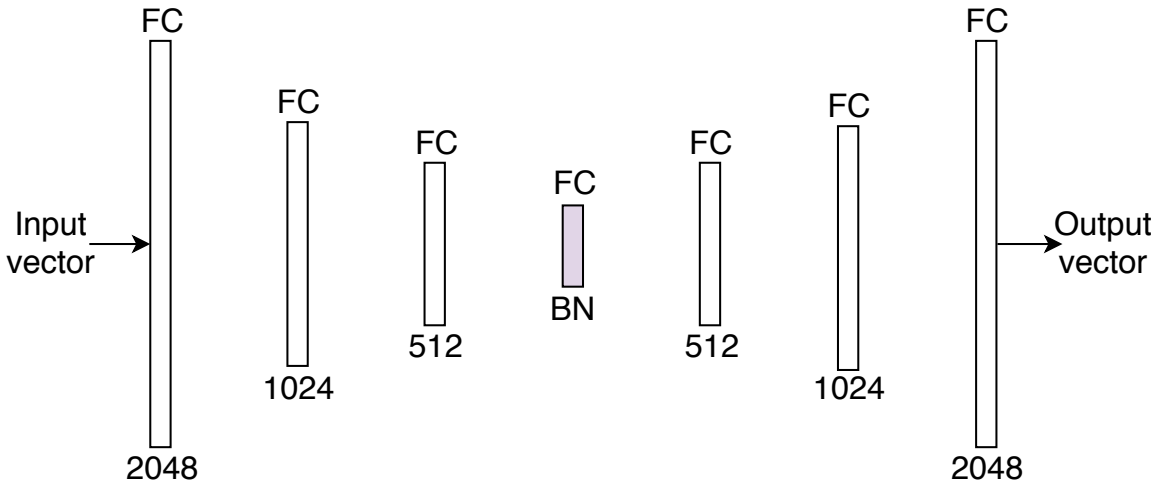


Figure 4.2: An overview of the vanilla autoencoder architecture used in this study

In vanilla autoencoders, both the encoder and decoder consist of multi-layer perceptrons (MLP). The architecture for the vanilla autoencoder is shown in Figure 4.2. Excluding the bottleneck layer, the encoder consists of three layers, which have $2048, 1024$ and 512 units respectively. The decoder is a mirror of the encoder. One frame of data (or features) is vectorized and passed as input to the model.

Convolutional autoencoders

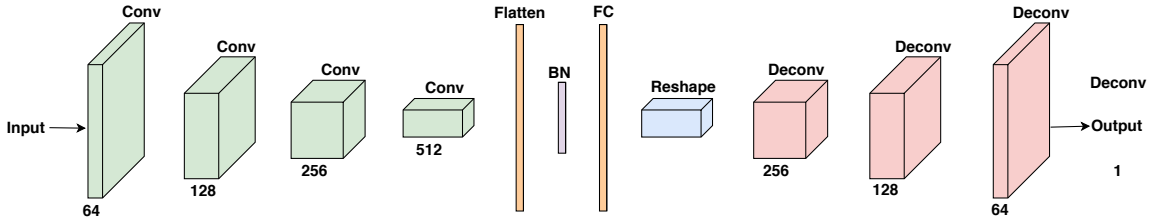


Figure 4.3: An overview of the convolutional autoencoder architecture used in this study

Convolutional autoencoders utilize convolutional layers in lieu of the fully connected layers. The architecture of the convolutional autoencoder is shown in Figure 4.3. The input is one second of data (or frame), and the autoencoder considers it a single channel image. The encoder contains four convolution blocks, leading to the bottleneck layer. Each of these blocks contains two 3×3 convolution layers with the same number of filters. Batch normalization is performed after each layer, and the convolution layers are followed by 2×2 max-pooling. The output of the last convolution block in the encoder is flattened into a vector, and is then connected to the bottleneck layer (from which the latent representations are taken).

The decoder begins with the a fully-connected (FC) layer containing the same number of units as the flattened output. It is reshaped to the same size as the fourth convolution block. Each deconvolution layer consists of an upsampling layer, appropriate padding and a 3×3 convolution layer to match the sizes to the corresponding convolution blocks. The choice of utilizing this combination instead of transposed convolutions is motivated by [39], which provides an illuminating illustration of the undesirable checkerboard artifacts that transposed convolution layers exhibit. The last deconvolution block results in the reconstructed output. Throughout, ReLU is the activation function used and the hyperbolic tangent function is used on the output.

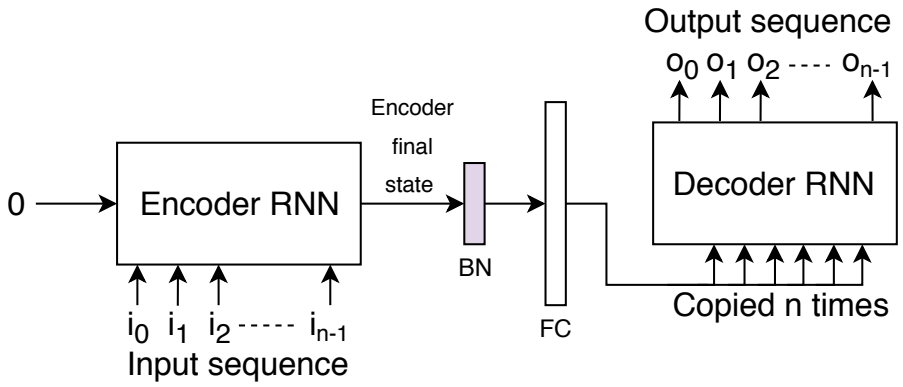


Figure 4.4: An overview of the recurrent autoencoder architecture used in this study

In a recurrent autoencoder, Vanilla RNNs or more powerful variants such as long short-term memory networks (LSTM) [40], gated recurrent units (GRU) [41] are used for the encoder and decoder. In order to formulate the design of the recurrent autoencoder, we turn to [26], who develop recurrent autoencoder models to learn representations for videos.

In our model (shown in Figure 4.4), both the encoder and decoder are initialized with zeros and the input sequence, which has n timesteps, is passed to the encoder. The final state of the encoder is passed through the bottleneck layer before being connected to another fully-connected (FC) layer. The resulting output is replicated (or copied) n times, and used as input to the decoder. The loss between the input and output (or reconstructed) sequences is used to update the model parameters.

DeepConvLSTM-based supervised representations

The DeepConvLSTM architecture was introduced in [11]. It consists of four convolutional layers with 64 filters, and a filter size of 5×1. The output of these convolutional layers is connected to a two-layer LSTM with 128 hidden units. The last hidden state of the LSTM is connected to the softmax output layer.

One of the analyses we perform is studying the effect of dimensionality of representations on the performance. In order accomplish this, we add another fully-connected layer after the LSTM whose dimension can be varied. To compute the DeepConvLSTM-based supervised representations, we perform a forward pass until the penultimate fully-connected layer.

Classifier

The common multi-layer perceptron (MLP) classifier has two layers, followed by the softmax output layer. These layers contain 2048 and 512 units respectively, and each layer is followed by batch normalization. The activation function used is ReLU. The model is shown in Figure 4.5.

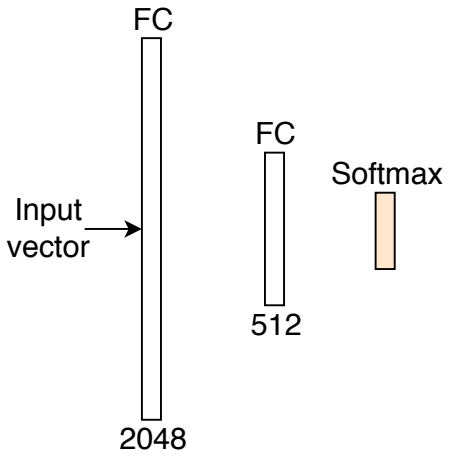


Figure 4.5: The architecture of the common backend classifier

Performance metrics

The mean f1-score is utilized as the core metric. The datasets used in this study, Opportunity in particular, are imbalanced and hence require performance metrics that are independent of the class distribution. Thus, we use mean f1-score, which is given by -

$$F_m = \frac{2}{|c|} \sum_c \frac{prec_c \times recall_c}{prec_c + recall_c} \tag{4.5}$$

where $prec_c$ and $recall_c$ are the precision and recall for each class.

CHAPTER 5
RESULTS AND DISCUSSION

This section examines the performance of the representations on the common backend classifier. As our focus is on understanding the role of these representations from a wearables standpoint, the considerations go beyond mere performance. Some other factors which affect the adoption of these representations for human activity recognition include time requirements for the computation of representations, the memory footprint and computational costs.

In order to assess the performance of the representation on these factors, we perform the following analyses –

- Performance of the representations vs the representation dimensions: we study how the performance of the representations is affected by the number of dimensions. This is an important factor for HAR as lower feature dimensions result in lower computational costs and computational time during classification.

- Time taken to compute the representations: the goal of HAR is to perform real-time recognition and lowering the latency in obtaining the representations results in better performance.

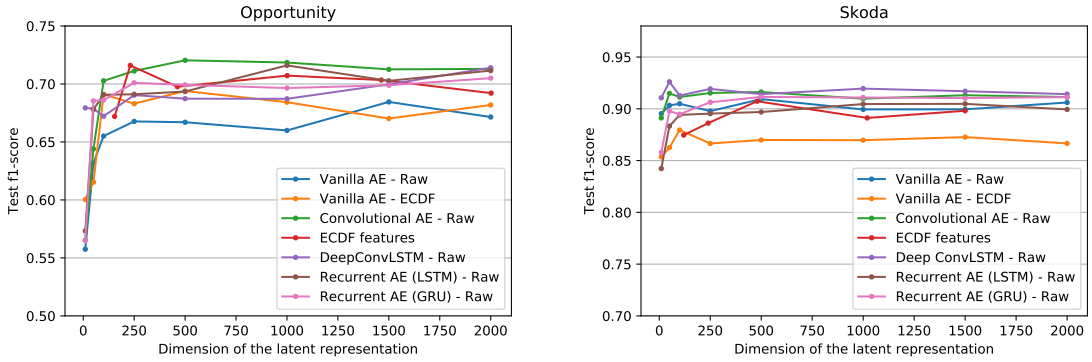
- Memory footprint: the onboard memory on wearables is limited, and hence, the memory required to store the models used to compute the representations is a vital factor.

- Number of trainable parameters for computing the representations: many representations in this study are obtained from deep learning based methods, in which the

number of trainable parameters offers insight into the effort required to compute the representations.

- Performance of the representations based on the amount of training data required: the size of the datasets in HAR is generally smaller than in other domains. Hence, representations which require less data to perform comparably are more suitable for HAR.

Performance of representations vs the representation dimensions

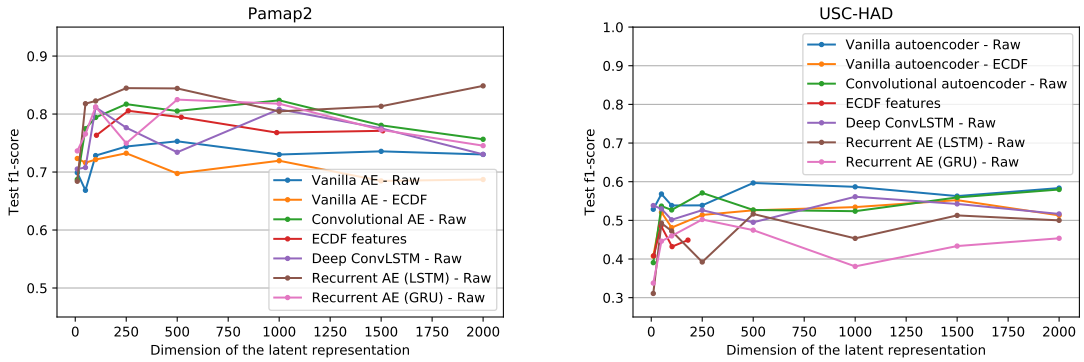


(a) Opportunity (b) Skoda

Figure 5.1: Performance of the models compared to the representation size on Opportunity and Skoda, respectively.

In Figure 5.1a, the performance of various models on the Opportunity dataset is illustrated. The Convolutional AutoEncoder (CAE) outperforms the other models at around 500 dimensions. DeepConvLSTM and Recurrent AE (LSTM) only match the performance of the CAE at 2000 dimensions. The distribution-based representation (ECDF) [1] obtains similar performance as the CAE at 250 dimensions. Thus, CAE and ECDF provide excellent

[1]Note: The number of dimensions of the distribution-based representation depends on the number of components (which is the number of points at which the inverse of P_i^c is computed, see Equation 4.1). For this study, we set the number of components so that the resulting feature dimensions match the dimensions of the latent representation for the other autoencoder models. However, the number of components is also limited by the length of the frame (which is 30 in our case). Thus, for the USC-HAD and Daphnet FoG datasets, the maximum number of dimensions possible is limited to around 250 (see Figures 5.2b and 5.3).



(a) PAMAP2 (b) USC-HAD

Figure 5.2: Performance of the models compared to the latent representation size on PAMAP2 and USC-HAD respectively.

performance at a fraction of the number of dimensions other representations require.

Next, we consider Skoda, which was recorded in a manufacturing scenario with a single worker performing tasks. Here, DeepConvLSTM provides the best performance (at 50 dimensions), with the CAE and Recurrent AE (GRU) providing similar performances. The worst performance is shown by the Vanilla AE on the distribution-based ECDF feature. In this case, both DeepConvLSTM or CAE at around 50 dimensions provide viable options for performing HAR with low dimension representations.

For PAMAP2, the performance of each representation is seen in Figure 5.2a. The Recurrent AE (LSTM) offers peak performance at 2000 dimensions, but also provides comparable performance at 250 dimensions (an eightfold reduction in the number of dimensions). The distribution-based representation performs best at 250 dimensions, but offers around 5% lower mean f1-score than the Recurrent AE (LSTM). Other representations require 1000 dimensions to perform well.

From Figure 5.2b, one can see that the Vanilla AE on raw data provides the best performance on USC-HAD. At higher dimensions, the DeepConvLSTM and Vanilla AE on

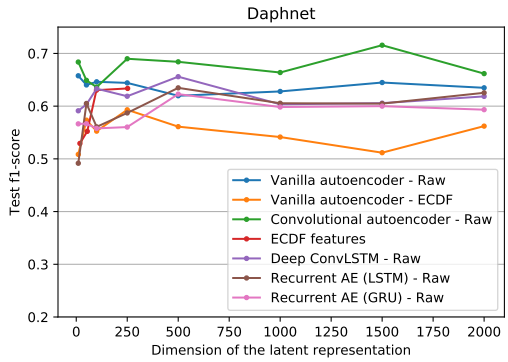


Figure 5.3: Performance of the models compared to the latent representation size on Daphnet FoG

ECDF features offer similar performance. The best test F1-score is obtained at 500 dimensions by the Vanilla AE on raw data. In contrast, the ECDF features perform considerably worse (around 8% at 50 dimensions). While the Convolutional AE matches the performance of the Vanilla AE on raw data, it happens at 2000 dimensions.

Daphnet FoG is a dataset which is used to study the freezing of gait using wearable sensors. From Figure 5.3, we can see that DeepConvLSTM and Vanilla AE on raw data have similar performances. At around 100 dimensions, the performance of ECDF features is similar to both DeepConvLSTM and Vanilla AE on raw data. In contrast, the Convolutional AE outperforms all other models even at 10 dimensions (0.684 mean f1-score). The best performance is achieved by the Convolutional AE at 1500 dimensions.

Time taken to compute the representations

Figure 5.4 compares the average time required to compute the representation for one frame of data. For this setup, the size of the representations is set to 500 dimensions. The time taken to compute the convolutional, recurrent, and DeepConvLSTM representations is comparably high (around 0.0004 seconds). In comparison, the Vanilla AE (both on the raw data as well as the ECDF features) take much lesser time to compute the representation.

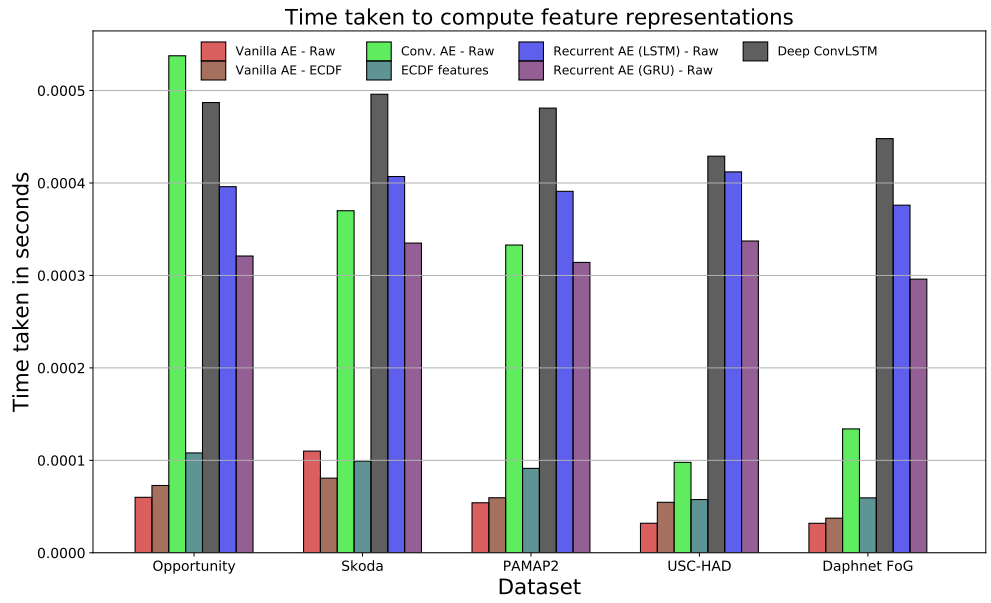


Figure 5.4: Comparison of the time taken to compute the representations on different datasets.

The ECDF features require around 0.0001 seconds to compute, which is slightly higher than the Vanilla AE. Thus, analyzing purely in terms of time required to compute the representations, we conclude that it is advantageous to utilize the distribution-based or Vanilla AE based representations.

Number of trainable parameters for computing the representations

In this section, we compare the number of trainable parameters required to compute representations. For the deep learning models, the parameters include both the encoder and decoder. The distribution-based representation does not have any trainable parameters and hence the bar plot shows zero (ECDF features are marked in dark green). As shown in Figure 5.5, the Vanilla AE possess a large number of trainable parameters (due to the presence of multiple fully connected layers). In comparison, the DeepConvLSTM based features have fewer trainable parameters (less than $0.25 * 10^7$ parameters).

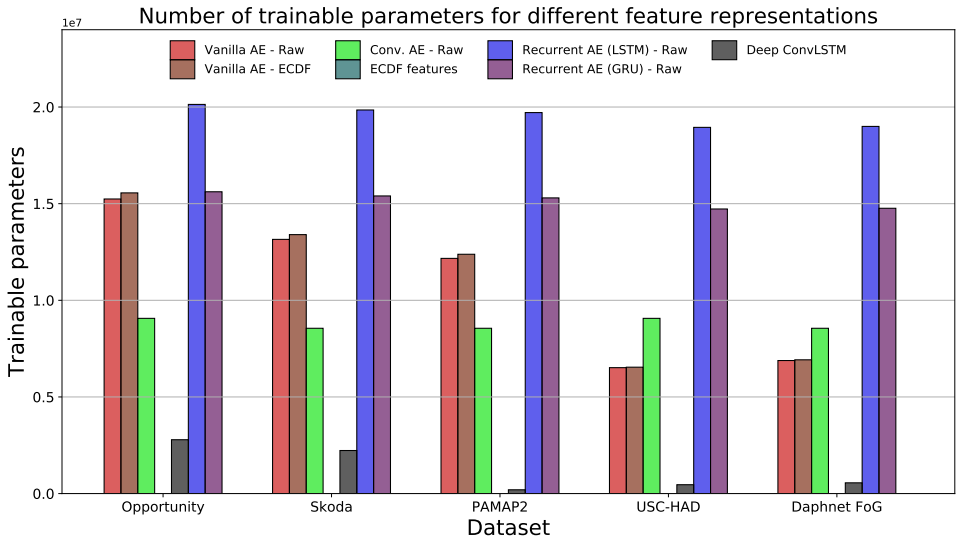


Figure 5.5: Comparison of the number of trainable parameters required to compute the representations on different datasets.

Models having larger number of trainable parameters are more prone to overfitting and poor generalization. Thus, models with fewer trainable parameters are preferred – such as DeepConvLSTM and Convolutional AE.

Memory footprint

Figure 5.6 details the amount of memory required onboard the wearable for the computation of representations. Since the distribution-based representation is computed directly, the bar plot shows a zero (or the gap). The Recurrent AE (GRU and LSTM) require the highest amount of memory (around 40 MB). In comparison, the Convolutional AE requires around 20MB. Among the learned features, the DeepConvLSTM based features require the least amount of memory. Thus, in terms of memory requirements, the distribution-based and DeepConvLSTM-based representations present the best option. Additionally, while the memory requirement for the Convolutional AE is higher, the potential improvements in performance render it a good alternative.

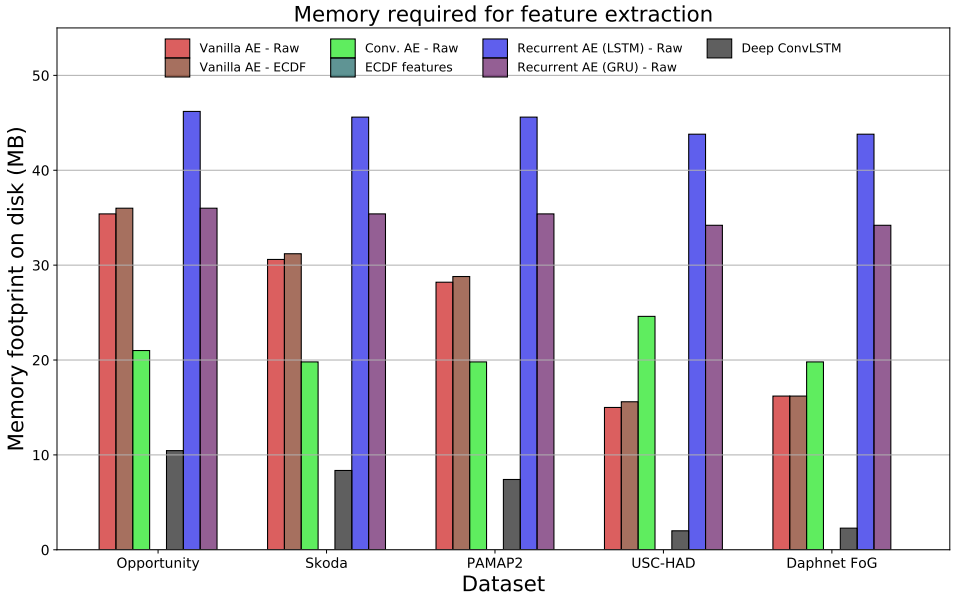
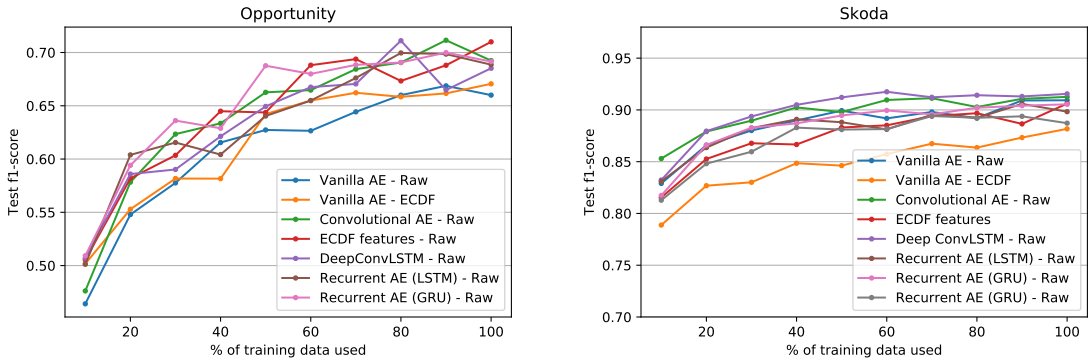


Figure 5.6: Comparison of the amount of memory required to store models for computing representations on different datasets.

Performance of the representations based on the amount of training data required

In this section, we analyze the performance of the features when the amount of training data is limited. Representations displaying good performance even on small datasets are critical for HAR as the datasets are generally smaller in size when compared to other domains such as vision. We set the size of the representations to 500 for this analysis since Figures 5.1a-5.3 suggest it to be a reasonable compromise between performance and representation dimensionality.

In Figure 5.7a, we analyze the percentage of training data required by the representations to obtain good performance on the Opportunity dataset. The distribution-based representation obtains the best performance at 100% (around 0.71), but also exhibits similar performance at 70% (around 0.694). Thus, the distribution-based representation requires 30% fewer training samples to produce similar performance. In case of the Recurrent AE



(a) Opportunity

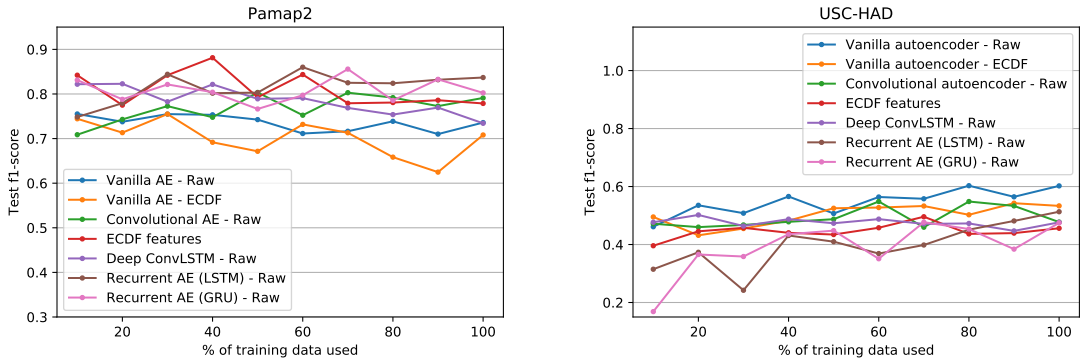
(b) Skoda

Figure 5.7: Performance of the representations based on the percentage of the training data required – for Opportunity and Skoda, respectively.

(GRU), the peak performance is obtained at 90% (0.7) but at 50%, it results in 0.6876. Thus, with 50% fewer training samples, the performance is reduced by ~ 0.11 mean F1-score.

Next, we study the relationship between performance and the amount of training data required, for Skoda. As seen in Figure 5.7b, most representations (apart from the Vanilla AE trained on ECDF features) result in similar performance. In the case of the Convolutional AE, the performance at 100% is 0.9126, and it shows 0.9096 at 60%. Thus, with 60% of the available data, approximately similar performance can be achieved. DeepConvLSTM displays the best performance at 60% (0.9175), which is slightly higher than the performance at 100% (0.9155). While the difference is not very high, it still shows that the performance of the DeepConvLSTM when trained on 40% fewer samples equals the performance when trained on the entire dataset.

The analysis for PAMAP2 is shown in Figure 5.8a. The Vanilla AE trained on ECDF features gives the worst performance. The Recurrent AE (LSTM) exhibits the best performance at 60% and the distribution-based representation achieves its best performance utilizing just 40% of the data.



(a) PAMAP2 (b) USC-HAD

Figure 5.8: Performance of the representations based on the percentage of the training data required – for PAMAP2 and USC-HAD, respectively.

Considering USC-HAD, Figure 5.8b depicts the performance of the representations based on the amount of training data used. The best performance utilizing the entire dataset is achieved by the Vanilla AE on raw data. It achieves similar performance at 80%. On this dataset, the more complex models perform worse. This could probably be explained by the fact that USC-HAD has only two sensors, resulting in a feature size of 6 dimensions.

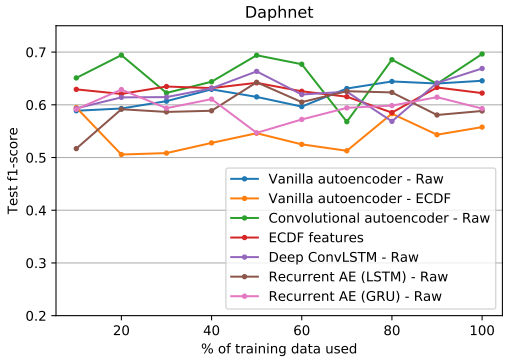


Figure 5.9: Performance of the representations based on the percentage of the training data required – Daphnet FoG

The Daphnet FoG dataset is studied for performing recognition on the freezing of gait. Considering the entire dataset, the Convolutional AE obtains the best performance (0.6964). But, it also achieves 0.6940 utilizing only 50% of the data. Similarly, the Deep-

ConvLSTM shows performance approximately equal to using the entire dataset using 50% of the data.

CHAPTER 6
CONCLUSION

Human activity recognition (HAR) involves the development of systems for the automated inference of peoples' activities. The feature representations used for the development of these systems are central to their performance. Previously, carefully designed features were utilized to represent the sensor data. However, the advent of deep learning has brought the promise of integrated learning – which eliminates the manual crafting and tuning of representations. In light of the developments in deep learning, we study the following question – 'What role do the representations play in HAR?'

From a wearables standpoint, the performance of a variety of representations is evaluated on a group of diverse datasets in order to establish their scope of applicability. The major contribution of this study lies in the application of convolutional and recurrent autoencoder based unsupervised feature learning methods for HAR. This allows us to potentially leverage large, unannotated datasets to derive rich representations.

We studied three categories of representations – distribution-based, deep learning based unsupervised representations (autoencoders), and deep learning based supervised representation (DeepConvLSTM). Further, we analyzed the computation time and computational costs of these representations using criteria relevant to wearable computing such as representation dimensionality, memory footprint, computation time, number of trainable parameters and amount of training data required.

The unsupervised deep learning representations outperform the distribution-based representation on all five datasets. The deep learning based approaches perform better in the

presence of abundant data. Therefore, on relatively larger datasets such as Opportunity and PAMAP2, the Convolutional and Recurrent AE outperform the other representations. However, this increase in performance comes at the cost of increased memory footprint, computational time and number of trainable parameters. On the other hand, the Vanilla AE based representations require lower computational time, but have a high memory footprint, as well as a large number of trainable parameters. However, the Vanilla AE based representations work well with datasets having few sensors such as the USC-HAD dataset.

The performance of the learned representations is contrasted against the distribution-based representation, which can be computed on-the-fly, and does not have any trainable parameters. Based on our experiments, we conclude that these properties – coupled with low computation time and comparable performance on multi-sensor datasets such as Opportunity – renders distribution-based representations a good option for usage in applications with constraints on memory and computation power.

We systematically demonstrate the effectiveness of Convolutional and Recurrent AE approaches for HAR. We believe that the increase in computational capabilities of wearable hardware in the years to come presents a great opportunity for the utilization of autoencoder based representations for HAR.

Limitations and Future Work

A challenge that requires further attention is the tradeoff inherent to deep learning models between being very flexible function approximators and being prone to overfitting. This tradeoff – which is caused by the large number of network parameters – is especially predominant in smaller HAR datasets. Due to the difficulty in creating large annotated HAR datasets, an interesting avenue of research is the study of data augmentation techniques. In computer vision, the datasets are augmented by adding image transformations, random

flips and crops. However, no such standard augmentation techniques exist in sensor based HAR. There is some preliminary work such as [42], which details jitter, crop and rotation techniques for Parkinson's Disease monitoring using CNNs. However, there is no work which systematically analyzes these augmentation techniques. Another approach to data augmentation would be the use of generative adversarial networks. Such data augmentation techniques could improve the performance by increasing variability in the training set.

As we have seen in Chapter 5, the autoencoder representations trained on the mean squared error (MSE) objective provide discriminative representations in an unsupervised manner. We hypothesize that adding another clustering objective to the MSE objective could improve the clustering ability of the representations. The clustering objective has been defined in Deep Embedding Clustering [43] where the authors improve the clustering performance on vision based datasets by utilizing a clustering objective along with the MSE. Improvements have been proposed in works such as [44, 45] which detail different training schedules and utilize convolutional layers for the encoding and decoding.

Appendices

APPENDIX A

DATA PROCESSING

The frames of data used in this study are 1 second long. While the original sampling rate varies with the dataset (30Hz for Opportunity and 100hz for USC-HAD for example), we downsample the data to 30Hz to maintain uniformity. Additionally, the data is normalized to have zero mean and unit variance.

The neural network architectures are implemented using the PyTorch framework [1] [46]. The learning rate is set to 0.001 and is reduced by a factor of 0.8 every 25 epochs. The Adam optimizer is used to update network parameters [47]. The autoencoders are trained for 150 epochs, while the common backend MLP classifier is trained for 300 epochs. The batch size is set to 100.

The distribution-based representation is computed using the code released on GitHub [2] by the authors of [8].

[1] https://pytorch.org/

[2] https://github.com/nhammerla/ecdfRepresentation

REFERENCES

[1] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Computing Surveys (CSUR)*, vol. 46, no. 3, p. 33, 2014.

[2] S. Bhattacharya and N. D. Lane, "Sparsification and separation of deep learning layers for constrained resource inference on wearables," in *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, ACM, 2016, pp. 176–189.

[3] M. Alizadeh and N. D. Lane, "Using pre-trained full-precision models to speed up training binary networks for mobile devices," in *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, ACM, 2018, pp. 528–528.

[4] P. Georgiev, N. D. Lane, C. Mascolo, and D. Chu, "Accelerating mobile audio sensing algorithms through on-chip gpu offloading," in *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*, ACM, 2017, pp. 306–318.

[5] N. D. Lane, S. Bhattacharya, A. Mathur, P. Georgiev, C. Forlivesi, and F. Kawsar, "Squeezing deep learning into mobile and embedded devices," *IEEE Pervasive Computing*, vol. 16, no. 3, pp. 82–88, 2017.

[6] T. Plötz, N. Y. Hammerla, and P. L. Olivier, "Feature learning for activity recognition in ubiquitous computing," in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[7] T. Huynh and B. Schiele, "Analyzing features for activity recognition," in *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: Innovative context-aware services: Usages and technologies*, ACM, 2005, pp. 159–163.

[8] N. Y. Hammerla, R. Kirkham, P. Andras, and T. Ploetz, "On preserving statistical characteristics of accelerometry data using their empirical cumulative distribution," in *Proceedings of the 2013 International Symposium on Wearable Computers*, ACM, 2013, pp. 65–68.

[9] N. Y. Hammerla, J. Fisher, P. Andras, L. Rochester, R. Walker, and T. Plötz, "Pd disease state assessment in naturalistic environments using deep learning," in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[10] B. Almaslukh, J. AlMuhtadi, and A. Artoli, "An effective deep autoencoder approach for online smartphone-based human activity recognition," *Int. J. Comput. Sci. Netw. Secur*, vol. 17, p. 160, 2017.

[11] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[12] P. Lukowicz, S. Intille, Ward, and e. . J. A., *Proc. Int. Workshop on How To Do Good Research In Activity Recognition: Experimental methodology, performance evaluation and reproducibility*,

[13] N. Y. Hammerla, "Activity recognition in naturalistic environments using body-worn sensors," 2015.

[14] D. Figo, P. C. Diniz, D. R. Ferreira, and J. M. Cardoso, "Preprocessing techniques for context recognition from accelerometer data," *Personal and Ubiquitous Computing*, vol. 14, no. 7, pp. 645–662, 2010.

[15] J. Frank, S. Mannor, and D. Precup, "Activity and gait recognition with time-delay embeddings," in *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[16] D. Minnen, T. Westeyn, T. Starner, J Ward, and P. Lukowicz, "Performance metrics and evaluation issues for continuous activity recognition," *Performance metrics for intelligent systems. NIST, Gaithersburg*, pp. 141–148, 2006.

[17] H. Li, G. D. Abowd, and T. Ploetz, "On specialized window lengths and detector based human activity recognition," in *Proceedings of the 2018 ACM International Symposium on Wearable Computers*, ACM, 2018, pp. 68–71.

[18] H. Kwon, G. D. Abowd, and T. Ploetz, "Adding structural characteristics to distribution-based accelerometer representations for activity recognition using wearables," in *Proceedings of the 2018 ACM International Symposium on Wearable Computers*, ACM, 2018, pp. 72–75.

[19] J. Yang, M. N. Nguyen, P. P. San, X. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition.," in *Ijcai*, vol. 15, 2015, pp. 3995–4001.

[20] M. Zeng, L. T. Nguyen, B. Yu, O. J. Mengshoel, J. Zhu, P. Wu, and J. Zhang, "Convolutional neural networks for human activity recognition using mobile sensors," in *6th International Conference on Mobile Computing, Applications and Services*, IEEE, 2014, pp. 197–205.

[21] Y. Guan and T. Plötz, "Ensembles of deep lstm learners for activity recognition using wearables," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 2, p. 11, 2017.

[22] M. Zeng, H. Gao, T. Yu, O. J. Mengshoel, H. Langseth, I. Lane, and X. Liu, "Understanding and improving recurrent networks for human activity recognition by continuous attention," in *Proceedings of the 2018 ACM International Symposium on Wearable Computers*, ACM, 2018, pp. 56–63.

[23] V. S. Murahari and T. Ploetz, "On attention models for human activity recognition," *ArXiv preprint arXiv:1805.07648*, 2018.

[24] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[25] G. E. Hinton, "To recognize shapes, first learn to generate images," *Progress in brain research*, vol. 165, pp. 535–547, 2007.

[26] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *International conference on machine learning*, 2015, pp. 843–852.

[27] S. Amiriparian, M. Freitag, N. Cummins, and B. Schuller, "Sequence to sequence autoencoders for unsupervised representation learning from audio," in *Proc. of the DCASE 2017 Workshop*, 2017.

[28] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[29] S. Bhattacharya, P. Nurmi, N. Hammerla, and T. Plötz, "Using unlabeled data in a sparse-coding framework for human activity recognition," *Pervasive and Mobile Computing*, vol. 15, pp. 242–262, 2014.

[30] J. Wang, X. Zhang, Q. Gao, H. Yue, and H. Wang, "Device-free wireless localization and activity recognition: A deep learning approach," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 7, pp. 6258–6267, 2017.

[31] R. Chavarriaga, H. Sagha, A. Calatroni, S. T. Digumarti, G. Tröster, J. d. R. Millán, and D. Roggen, "The opportunity challenge: A benchmark database for on-body sensor-based activity recognition," *Pattern Recognition Letters*, vol. 34, no. 15, pp. 2033–2042, 2013.

[32] N. Y. Hammerla, S. Halloran, and T. Plötz, "Deep, convolutional, and recurrent models for human activity recognition using wearables," *ArXiv preprint arXiv:1604.08880*, 2016.

[33] T. Stiefmeier, D. Roggen, G. Ogris, P. Lukowicz, and G. Tröster, "Wearable activity tracking in car manufacturing," *IEEE Pervasive Computing*, no. 2, pp. 42–50, 2008.

[34] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *2012 16th International Symposium on Wearable Computers*, IEEE, 2012, pp. 108–109.

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

[36] M. Bachlin, M. Plotnik, D. Roggen, I. Maidan, J. M. Hausdorff, N. Giladi, and G. Troster, "Wearable assistant for parkinsons disease patients with the freezing of gait symptom," *IEEE Transactions on Information Technology in Biomedicine*, vol. 14, no. 2, pp. 436–446, 2010.

[37] J. G. Nutt, B. R. Bloem, N. Giladi, M. Hallett, F. B. Horak, and A. Nieuwboer, "Freezing of gait: Moving forward on a mysterious clinical phenomenon," *The Lancet Neurology*, vol. 10, no. 8, pp. 734–744, 2011.

[38] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, `http://www.deeplearningbook.org`.

[39] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016.

[40] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[41] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Gated feedback recurrent neural networks," in *International Conference on Machine Learning*, 2015, pp. 2067–2075.

[42] T. T. Um, F. M. J. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulić, "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks," *ArXiv preprint arXiv:1706.00527*, 2017.

[43] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*, 2016, pp. 478–487.

[44] X. Guo, L. Gao, X. Liu, and J. Yin, "Improved deep embedded clustering with local structure preservation.," in *IJCAI*, 2017, pp. 1753–1759.

[45] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *International Conference on Neural Information Processing*, Springer, 2017, pp. 373–382.

[46] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *ArXiv preprint arXiv:1412.6980*, 2014.