

**A MEMORY EFFICIENT NOVEL
DEEP LEARNING ARCHITECTURE
ENABLING DIVERSE FEATURE EXTRACTION
ON WEARABLE MOTION SENSOR DATA**

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

By
Enes Koşar
September 2022

A Memory Efficient Novel
Deep Learning Architecture
Enabling Diverse Feature Extraction
on Wearable Motion Sensor Data

By Enes Koşar
September 2022

We certify that we have read this thesis and that in our opinion it is fully adequate,
in scope and in quality, as a thesis for the degree of Master of Science.

Billur Barshan Özaktaş (Advisor)

Ömer Morgül

Pınar Duygulu Şahin

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

A MEMORY EFFICIENT NOVEL
DEEP LEARNING ARCHITECTURE
ENABLING DIVERSE FEATURE EXTRACTION
ON WEARABLE MOTION SENSOR DATA

Enes Koşar

M.S. in Electrical and Electronics Engineering

Advisor: Billur Barshan Özaktas

September 2022

Extracting representative features to recognize human activities through the use of wearables is an area of on-going research. We propose a novel hybrid network architecture to recognize human activities through the use of wearable motion sensors and deep learning techniques. The long short-term memory (LSTM) and the 2D convolutional neural network (CNN) branches of the model that run in parallel receive the raw signals and their spectrograms, respectively. We compare the classification performance of the proposed network with five commonly used network architectures: 1D CNN, 2D CNN, LSTM, standard 1D CNN-LSTM, and an alternative 1D CNN-LSTM model. We tune the hyper-parameters of all six models using Bayesian optimization and test the models on two publicly available datasets. The proposed 2D CNN-LSTM architecture achieves the highest average accuracies of 95.66% and 92.95% on the two datasets, which are, respectively, 2.45% and 3.18% above those of the 2D CNN model that ranks the second. User identification is another problem that we have addressed in this thesis. Firstly, we use binary classifier models to detect activity signals that are useful for the user identity recognition task. Useful signals are transmitted to the next module and used by the proposed deep learning model for user identity recognition. Moreover, we investigate feature transfer between the human activity and user identity recognition tasks which enables shortening the training processes by 8.7 to 17 times without a significant degradation in classification accuracies. Finally,

we elaborate on reducing the model sizes of the proposed models for human activity and user identity recognition problems. By using transfer learning, pooling layers, and eight-bit weight quantization methods, we have reduced the model sizes by 17–116 times without a significant degradation in classification accuracies.



Keywords: Human Activity Recognition, User Identity Recognition, Wearable Sensors, Wearables, Motion Sensors, Deep Learning, Transfer Learning.

ÖZET

GİYİLEBİLİR HAREKET ALGILAYICILARI VERİSİ İLE KAPSAMLI ÖZNİTELİK ÇIKARMA SAĞLAYAN BELLEK VERİMLİ YENİLİKÇİ BİR DERİN ÖĞRENME MİMARİSİ

Enes Koşar

Elektrik ve Elektronik Mühendisliği, Yüksek Lisans

Tez Danışmanı: Billur Barshan Özaktaş

Eylül 2022

Giyilebilir algılayıcı verilerini analiz ederek insan hareketlerini algılamak için temsili öznitelik çıkarmak güncel bir araştırma alanıdır. Bu çalışmada insan aktivitelerini tanımak için giyilebilir hareket algılayıcıları ve derin öğrenme yöntemleri kullanılarak yenilikçi karma bir derin yapay sinir ağı mimarisi önerilmektedir. Sunulan yapıda zaman dizisi sinyalleri ve bu sinyallerin zaman-frekans resimleri (spektrogram) sırasıyla modelin paralel halde çalışan uzun-kısa vadeli bellek ve iki boyutlu evrişimsel sinir ağları kollarını beslemek için kullanılmıştır. Önerilen yapının sınıflandırma başarımı, bu alanda yaygın olarak kullanılmakta olan beş farklı mimari olan tek boyutlu evrişimsel yapay sinir ağları, iki boyutlu evrişimsel yapay sinir ağları, uzun-kısa vadeli bellek ağları, standart tek boyutlu karma ağlar ve alternatif tek boyutlu karma ağlar ile karşılaştırılmıştır. Söz konusu altı modelin üstdeğişken ayarları Bayesçi eniyileme kullanılarak yapılmış ve modellerin başarımları iki farklı veriseti kullanılarak elde edilmiştir. Önerilen karma derin yapay sinir ağı yapısı verisetleri üzerinde 95.66% ve 92.95% doğruluk oranlarıyla en iyi ikinci mimari olan iki boyutlu evrişimsel yapay sinir ağları yapısından 2.45% ve 3.18% daha yüksek başarımla elde etmiştir. Kullanıcı kimliği tanıma görevi, bu tezde çalışılmış olan bir diğer konudur. Bu görev için öncelikle hareket algılayıcılardan alınan sinyalleri, kullanıcı kimliği tanımak için elverişli ve elverişsiz olarak sınıflandıran bir

ikili model geliştirilmiştir. Kullanışlı olarak sınıflandırılan sinyaller, önerilen derin öğrenme yapısı kullanılarak cihaz kullanıcısının kimliğini saptamak için kullanılmıştır. Hareket tanıma ve kimlik tanıma görevleri arasında öğrenme aktarması yapılarak başarımlarında kayda değer bir kayıp yaşanmadan modellerin eğitilme süreleri yaklaşık olarak 8.7–17 kat azaltılmıştır. Son olarak, aktivite ve kullanıcı kimliği tanımak için kullanılan bu modellerin bellek gereksinimlerini düşürmek üzerine çalışılmıştır. Öğrenme aktarması, örnekleme katmanları ve sekiz-bit nicemleme yöntemleri uygulanarak modellerin bellek gereksinimleri yaklaşık olarak 17–116 kat azaltılmıştır.

Anahtar sözcükler: İnsan Hareketi Tanıma, Kullanıcı Kimliği Tanıma, Giyilebilir Algılayıcılar, Giyilebilir Teknolojiler, Hareket Algılayıcıları, Derin Öğrenme, Öğrenme Aktarması.

Acknowledgement

I would like to express my greatest gratitude to my supervisor Prof. Dr. Billur Barshan Özaktaş for her valuable support and guidance throughout the duration of my M.S. study. She was always encouraging, helpful, and accessible for me.

Furthermore, I would like to thank the members of my jury, Prof. Dr. Ömer Morgül and Prof. Dr. Pınar Duygulu Şahin for reviewing this thesis and sharing their valuable opinions, feedback, and criticism.

Türk Telekom and Information and Communication Technologies Authority (BTK) supported my master's degree through the 5G and Beyond Scholarship. Scientific and Technological Research Council of Turkey supported me with the the graduate studies scholarship (BİDEB-2210A) throughout my M.S. study. I owe particular thanks to these institutions for their support.

I would like to thank my friends, colleagues, and officemates for their supportive companionship and for the beautiful times we spent together.

Finally and most importantly, I want to express my sincere appreciation to my beloved family whose existence is always invaluable for me.

Contents

1	Introduction	1
1.1	Problem Definition	1
1.2	Related Work	4
1.2.1	Deep Learning for Human Activity Recognition	4
1.2.2	Transfer Learning for Human Activity Recognition	9
1.2.3	Computationally Efficient DL Models for Human Activity Recognition on Resource Limited Devices	13
1.3	Contributions and the Outline of the Thesis	16
2	Enabling Diverse Feature Extraction: A Novel 2D CNN-LSTM Architecture	19
2.1	The Proposed Methodology	19
2.1.1	Datasets	21
2.1.2	Preprocessing	22
2.1.3	Implementation and Resources	25

2.1.4	Layer Selection	25
2.1.5	Hyper-Parameter Selection	31
2.1.6	Training, Validation, Testing Methods, and Performance Analysis	32
2.2	Experimental Results and Discussion	34
2.2.1	Hyper-Parameter Optimization Results	34
2.2.2	Performance Comparison of the Proposed Model with the Five Existing Models	36
2.2.3	Loss versus Epoch, Accuracy versus Epoch Graphs and Confusion Matrices of the Proposed Model	40
2.2.4	Comparison of Computational Cost and Complexity	44
2.2.5	Comparison of Our Results with Existing Studies	46
3	Adopting Extracted Features for Different Problem Domains via Transfer Learning	50
3.1	Transfer of Features from the Activity Domain to the User Identity Domain	51
3.1.1	Selecting a Useful Group of Activities for User Identity Recognition	51
3.1.2	Selecting the Features to be Transferred	53
3.1.3	Hyper-parameter Selection	56
3.1.4	Comparison of the Full and Transfer Models	58

3.1.5	Binary Classifier for Useful Activity Selection	63
3.1.6	Comparison with Recent Studies	64
3.2	Transfer Features from the User Identity Domain to the Activity Domain	65
3.2.1	Selecting the Features to be Transferred	65
3.2.2	Hyper-parameter Selection	68
3.2.3	Comparison of the Full and Transfer Models	70
4	Reducing the Model Sizes of the Proposed Models for Memory Limited Devices	74
4.1	Investigating the Factors Increasing the Size of the Proposed Models	77
4.2	Methods to Reduce the Size of the Proposed Model	79
4.3	Experimental Results	83
5	Summary, Conclusions, and Future Work	87
A	Hyper-parameters	100

List of Figures

1.1	Human activity recognition through the use of wearables.	2
1.2	Sample raw sensor signals acquired from the (a) UCI HAR Dataset and the (b) DSA Dataset.	3
1.3	Single network models. (a) The LSTM model and the (b) 1D CNN model processing the raw input signals, and the (c) 2D CNN model processing the spectrogram of raw input signals.	6
1.4	The standard 1D CNN-LSTM network architecture.	8
2.1	The proposed 2D CNN-LSTM hybrid architecture.	20
2.2	Sample spectrograms obtained from the UCI HAR Dataset for the activities (a) walking on a flat surface, (b) walking upstairs, (c) walking downstairs, (d) sitting, (e) standing, and (f) laying. . . .	23
2.3	Sample spectrograms obtained from the DSA Dataset for the activities (a) lying on the right side, (b) ascending stairs, (c) walking on a treadmill at a speed of 4 km/h in flat position, (d) rowing, (e) jumping, and (f) playing basketball.	24
2.4	Layers of the 1D CNN, 2D CNN, LSTM, and the standard 1D CNN-LSTM models used in processing the UCI HAR Dataset. . .	27

2.5	Layers of the alternative 1D CNN-LSTM and the proposed 2D CNN-LSTM models used in processing the UCI HAR Dataset. . .	28
2.6	Layers of the 1D CNN, 2D CNN, LSTM, and the standard 1D CNN-LSTM models used in processing the DSA Dataset.	29
2.7	Layers of the alternative 1D CNN-LSTM and the proposed 2D CNN-LSTM models used in processing the DSA Dataset.	30
2.8	Loss versus epoch graph of the proposed model based on the test set of the UCI HAR Dataset.	41
2.9	Accuracy versus epoch graph of the proposed model based on the test set of the UCI HAR Dataset.	41
2.10	Confusion matrix of the proposed model based on the test set of the UCI HAR Dataset.	42
2.11	Loss versus epoch graph of the proposed model based on testing with the second subject's data of the DSA Dataset.	42
2.12	Accuracy versus epoch graph of the proposed model based on testing with the second subject's data of the DSA Dataset.	43
2.13	Confusion matrix of the proposed model based on testing with the second subject's data of the DSA Dataset.	43
3.1	Transfer learning from the activity domain to the user identity domain for the UCI HAR Dataset.	54
3.2	Transfer learning from the activity domain to the user identity domain for the DSA Dataset.	55
3.3	User identity domain convergence with respect to epoch numbers for the UCI HAR Dataset.	61

3.4	User identity domain convergence with respect to time for the UCI HAR Dataset.	61
3.5	User identity domain convergence with respect to epoch numbers for the DSA Dataset.	62
3.6	User identity domain convergence with respect to time for the DSA Dataset.	62
3.7	User identity recognition scheme with wearable motion sensor data.	63
3.8	Transfer learning from the user identity domain to the activity domain for the UCI HAR Dataset.	67
3.9	Transfer learning from the user identity domain to the activity domain for the DSA Dataset.	68
3.10	Activity domain convergence with respect to epoch numbers in processing the UCI HAR Dataset.	71
3.11	Activity domain convergence with respect to time in processing the UCI HAR Dataset.	71
3.12	Activity domain convergence with respect to epoch numbers in processing the DSA Dataset.	72
3.13	Activity domain convergence with respect to time in processing the DSA Dataset.	72
4.1	Transfer models for identity recognition on UCI HAR (left) and DSA (right) Datasets.	75
4.2	Binary models for useful activity detection on UCI HAR (left) and DSA (right) Datasets.	75

4.3	Transfer models for activity recognition on UCI HAR (left) and DSA (right) Datasets.	76
4.4	Max pooling operation.	80
4.5	Convolution operation with a filter size of 1×1	81
4.6	Quantization operation.	82
4.7	Eight-bit quantization.	83

List of Tables

1.1	Existing standard 1D CNN-LSTM hybrid model architectures. . .	7
2.1	Optimized hyper-parameters of the proposed 2D CNN-LSTM in processing the UCI HAR Dataset.	34
2.2	Optimized hyper-parameters of the proposed 2D CNN-LSTM in processing the DSA Dataset.	35
2.3	Total hyper-parameter tuning times for the UCI HAR Dataset. . .	35
2.4	Total hyper-parameter tuning times for the DSA Dataset.	36
2.5	Comparison of the implemented single and hybrid models in terms of their subnetworks, merging method, and performance metrics. We provide the average values of accuracy, f_1 score, recall, and precision metrics plus/minus one standard deviation for processing the UCI HAR Dataset.	38
2.6	Comparison of the implemented single and hybrid models in terms of their subnetworks, merging method, and accuracy performance metric. We provide the average value of accuracy plus/minus one standard deviation for processing the DSA Dataset.	39

2.7	Complexities of the models implemented to process the UCI HAR Dataset.	45
2.8	Complexities of the models implemented to process the DSA Dataset.	45
2.9	Comparison of accuracy performance metrics of the proposed model and the related work for processing the UCI HAR Dataset.	48
2.10	Comparison of accuracy performance metrics of the proposed model and the related work for processing the DSA Dataset. . . .	48
3.1	User identity recognition accuracies for each type of activity in the UCI HAR Dataset.	52
3.2	User identity recognition accuracies for each type of activity in the DSA Dataset.	52
3.3	User identity recognition accuracies by features extracted from each layer for the UCI HAR Dataset.	55
3.4	User identity recognition accuracies by features extracted from each layer for the DSA Dataset.	55
3.5	User identity recognition accuracies by features extracted from each layer for the DSA Dataset.	56
3.6	User identity recognition accuracies as the number of neurons changes for the UCI HAR Dataset.	57
3.7	User identity recognition accuracies as the number of neurons changes for the DSA Dataset.	57
3.8	User identity recognition accuracies as the dropout probability changes for the UCI HAR Dataset.	58

3.9	User identity recognition accuracies as the dropout probability changes for the DSA Dataset.	58
3.10	Full versus transferred network from the activity domain to the user identity domain for the UCI HAR Dataset.	59
3.11	Full versus transferred network from the activity domain to the user identity domain for the DSA Dataset.	59
3.12	Training times of each layer in processing the UCI HAR Dataset.	60
3.13	Training times of each layer in processing the DSA Dataset. . . .	60
3.14	Binary classifier results for the UCI HAR Dataset.	64
3.15	Binary classifier results for the DSA Dataset.	64
3.16	Results in the literature for the UCI HAR Dataset.	65
3.17	Activity recognition accuracies by features extracted from each layer for the UCI HAR Dataset.	66
3.18	Activity recognition accuracies by features extracted from each layer for the DSA Dataset.	66
3.19	Activity recognition accuracies by features extracted from each layer for the DSA Dataset.	67
3.20	Activity recognition accuracy as the number of neurons is varied for the UCI HAR Dataset.	69
3.21	Activity recognition accuracy as number of neurons is varied for the DSA Dataset.	69
3.22	Activity recognition accuracy as dropout probability is varied for the UCI HAR Dataset.	69

3.23	Activity recognition accuracies as dropout probability is varied for the DSA Dataset.	69
3.24	Full versus transfer network from the user identity domain to the activity domain for the UCI HAR Dataset.	70
3.25	Full versus transferred network from the user identity domain to the activity domain for the DSA Dataset.	70
4.1	Number of parameters of the proposed full model layers for the user identity recognition task in processing the UCI HAR Dataset.	78
4.2	Number of parameters of the proposed transfer model layers for the user identity recognition task in processing the UCI HAR Dataset.	78
4.3	Number of parameters of the proposed transfer model with pooling layers for the user identity recognition task in processing the UCI HAR Dataset.	81
4.4	User identity recognition accuracies, number of parameters, and the model size for the UCI HAR Dataset.	84
4.5	User identity recognition accuracies, number of parameters, and the model size for the DSA Dataset.	84
4.6	Binary classifier accuracies, number of parameters, and the model size for the UCI HAR Dataset.	85
4.7	Binary classifier accuracies, number of parameters, and the model size for the DSA Dataset.	85
4.8	Activity recognition accuracies, number of parameters, and the model size for the UCI HAR Dataset.	86

4.9	Activity recognition accuracies, number of parameters, and the model size for the DSA Dataset.	86
A.I	Optimized hyper-parameters of the 1D CNN in processing the UCI HAR Dataset.	100
A.II	Optimized hyper-parameters of the 2D CNN in processing the UCI HAR Dataset.	101
A.III	Optimized hyper-parameters of the LSTM in processing the UCI HAR Dataset.	101
A.IV	Optimized hyper-parameters of the standard 1D CNN-LSTM in processing the UCI HAR Dataset.	102
A.V	Optimized hyper-parameters of the alternative 1D CNN-LSTM in processing the UCI HAR Dataset.	102
A.VI	Optimized hyper-parameters of the 1D CNN in processing the DSA Dataset.	103
A.VII	Optimized hyper-parameters of the 2D CNN in processing the DSA Dataset.	103
A.VIII	Optimized hyper-parameters of the LSTM in processing the DSA Dataset.	103
A.IX	Optimized hyper-parameters of the standard 1D CNN-LSTM in processing the DSA Dataset.	104
A.X	Optimized hyper-parameters of the alternative 1D CNN-LSTM in processing the DSA Dataset.	104

Chapter 1

Introduction

1.1 Problem Definition

Through the pervasiveness of a communicating network of interconnected devices and computing intelligence, wearables have become one of the key elements of the Internet of Things (IoT) ecosystem. Continuous streaming of easily accessible signals, acquired from sensors embedded in wearable devices, provide vast amounts of data that carry valuable information about the user state and well being. Proper processing of these data allows developing innovative solutions to challenging problems [1].

The aim of human activity recognition (HAR) is to detect and monitor activities automatically through the use of sensory input [2, 3]. It is pertinent to areas such as ambient intelligence, context-aware systems, assistive technologies, healthcare, biomechanics, sports science, and ergonomics [4, 5, 6, 7]. Smart environment based and wearable sensor based solutions to HAR exist [8]. Besides the high installation cost and confining the user to a bounded area, the common use of videos in the former approach entails privacy issues. A more favorable and less costly approach is based on wearable sensor technology which allows direct

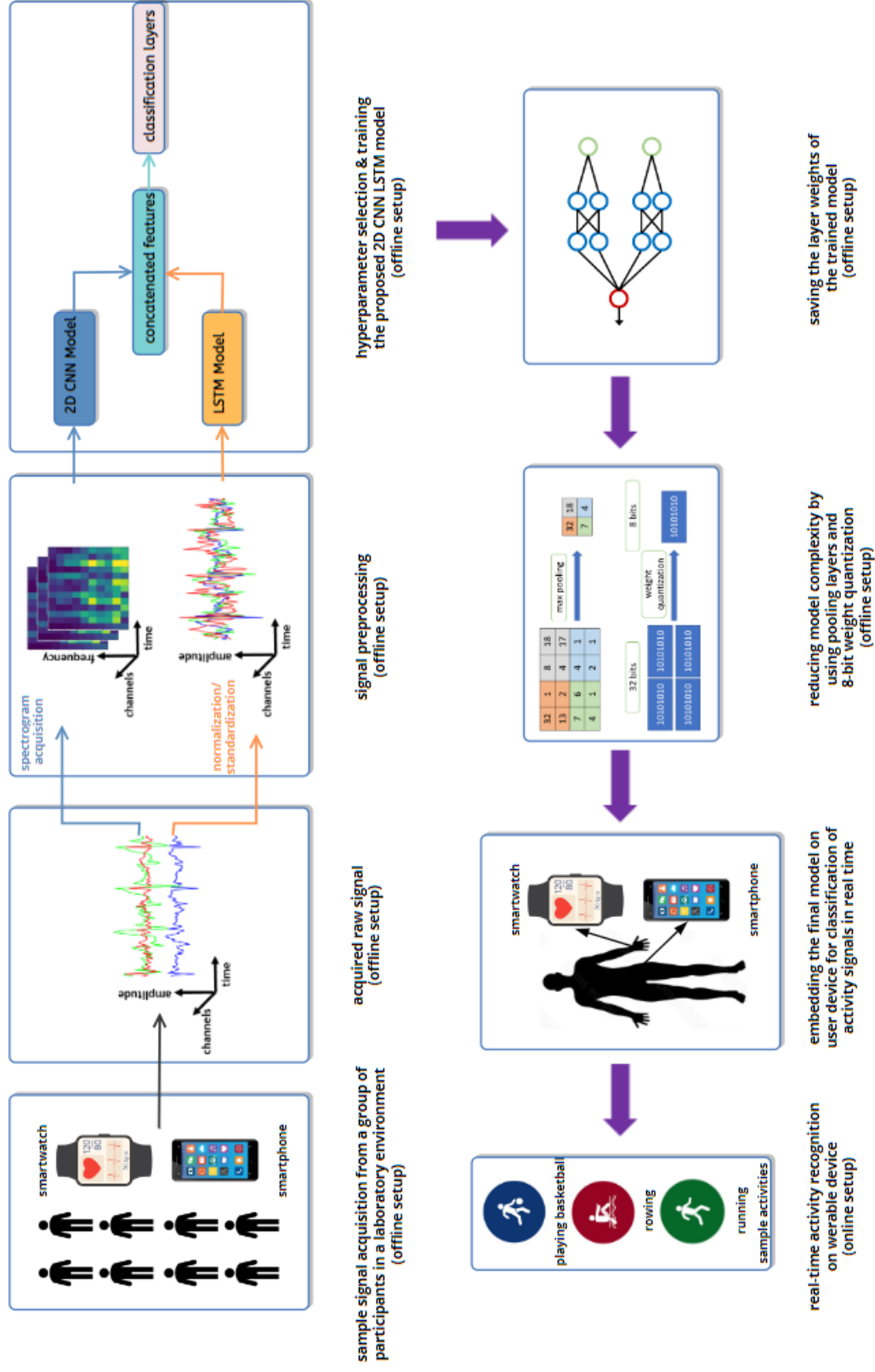
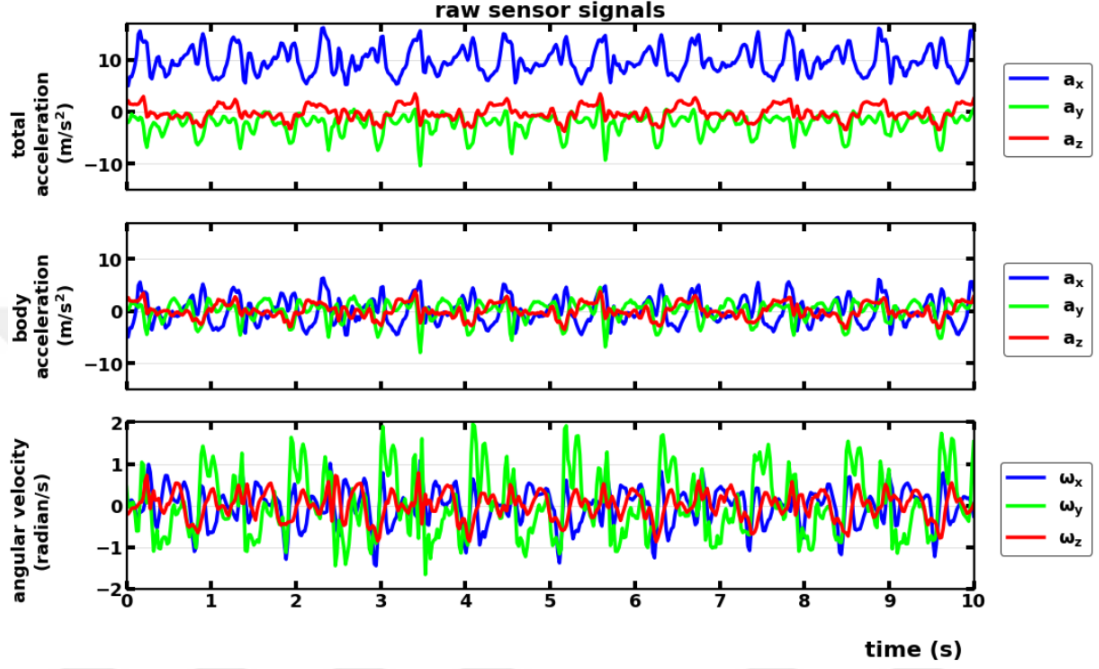
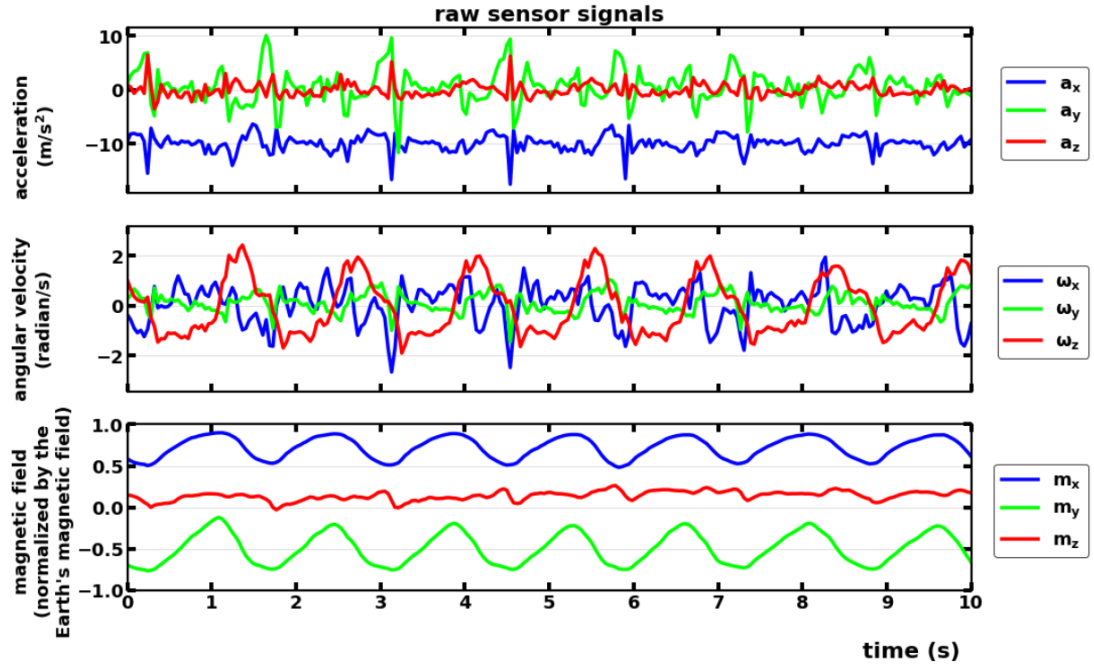


Figure 1.1: Human activity recognition through the use of wearables.



(a)



(b)

Figure 1.2: Sample raw sensor signals acquired from the (a) UCI HAR Dataset and the (b) DSA Dataset.

recording of time-series signals in 3D without any occlusion effects or correspondence problem (Fig. 1.1). Time-series signals are recorded from multiple sensor axes which can possibly be partitioned into shorter time segments. Accelerometers, gyroscopes, and magnetometers are the commonly used wearable motion sensor types. Fig. 1.2 illustrates sample raw signal recordings from the two datasets that we have employed in this thesis.

Wearable sensor signals usually contain personal information about the device users such as recordings of daily and sport activities, heart rates, and sleeping patterns. In order to prevent third party access to personal user information and to be able to keep recordings only when the device is on its real user, wearable devices are required to recognize user identities. User identity recognition aims to detect identities of device users automatically by using sensor signals and find applications in security and surveillance systems, personalized technologies, and healthcare systems.

One of the challenges of activity recognition, which involves a classification task, is the extraction of features [9]. In this thesis, we investigate the best deep learning (DL) architecture for extracting highly representative features for the accurate classification of human activities through the use of wearable motion sensors. Afterwards, we use these features for the user identity recognition problem and investigate feature transfer between the activity and user identity recognition tasks. Finally, several methods for reducing the model sizes are implemented in order to deploy proposed models on memory limited wearable devices.

1.2 Related Work

1.2.1 Deep Learning for Human Activity Recognition

Recent studies have favored DL models over machine learning (ML) techniques [10, 11, 12] since the former has the advantage of providing automated

feature extraction and superior classification performance, despite the heavy computational requirements. Specifically, single CNN, single LSTM, and hybrid models that combine CNN and LSTM in different ways have been investigated for HAR.

Since LSTM models provide favorable results on time-series data [Fig. 1.3(a)], they are a suitable choice for HAR based on wearable sensors [13, 14, 15, 16]. CNNs, which are particularly successful models for processing image data, have also been used for HAR [16, 17, 18, 19] where the acquired signals are typically time-series data. When this is the case, a commonly preferred approach is to use 1D CNNs and perform the convolution only along the time axis [Fig. 1.3(b)]. However, other methods can be also employed such as extracting the spectrogram of the raw signal and feeding it as input to a 1D or 2D CNN [Fig. 1.3(c)]. Although 2D CNNs with spectrogram inputs have been used in processing micro-Doppler radar signals for various purposes [20, 21, 22, 23], they are not sufficiently well investigated for wearable sensor signals. To our knowledge, there is a small number of studies where spectrograms of wearable sensor signals are provided as input to a 1D CNN [24, 25, 26] or to a 2D CNN [27, 28, 29, 30].

Besides the single CNN and LSTM models, hybrid structures aiming to combine the strong aspects of both models exist. Hybrid models usually involve a cascaded (series) connection such as CNN-LSTM and LSTM-CNN where the output of the first model is fed as input to the second one in the sequence. Fig. 1.4 illustrates the standard 1D CNN-LSTM model that we have also implemented in this study for comparison with our proposed model. Some standard 1D CNN-LSTM hybrid model architectures used in previous studies are provided in Table 1.1. Mekruksavanich and Jitpattanakul [10] propose a 1D CNN-LSTM network structure using a four-layer 1D CNN and a single-layer LSTM and show that this structure has superior performance compared to a single LSTM model. However, that study does not compare the performance of the 1D CNN-LSTM hybrid model with CNN models. Studies reported in [31] and [32] also compare the 1D CNN-LSTM with the LSTM and confirm that using this hybrid model improves the activity classification accuracy. Reference [33] uses four 1D CNN and two LSTM layers. The use of two LSTMs in sequence distinguishes it from

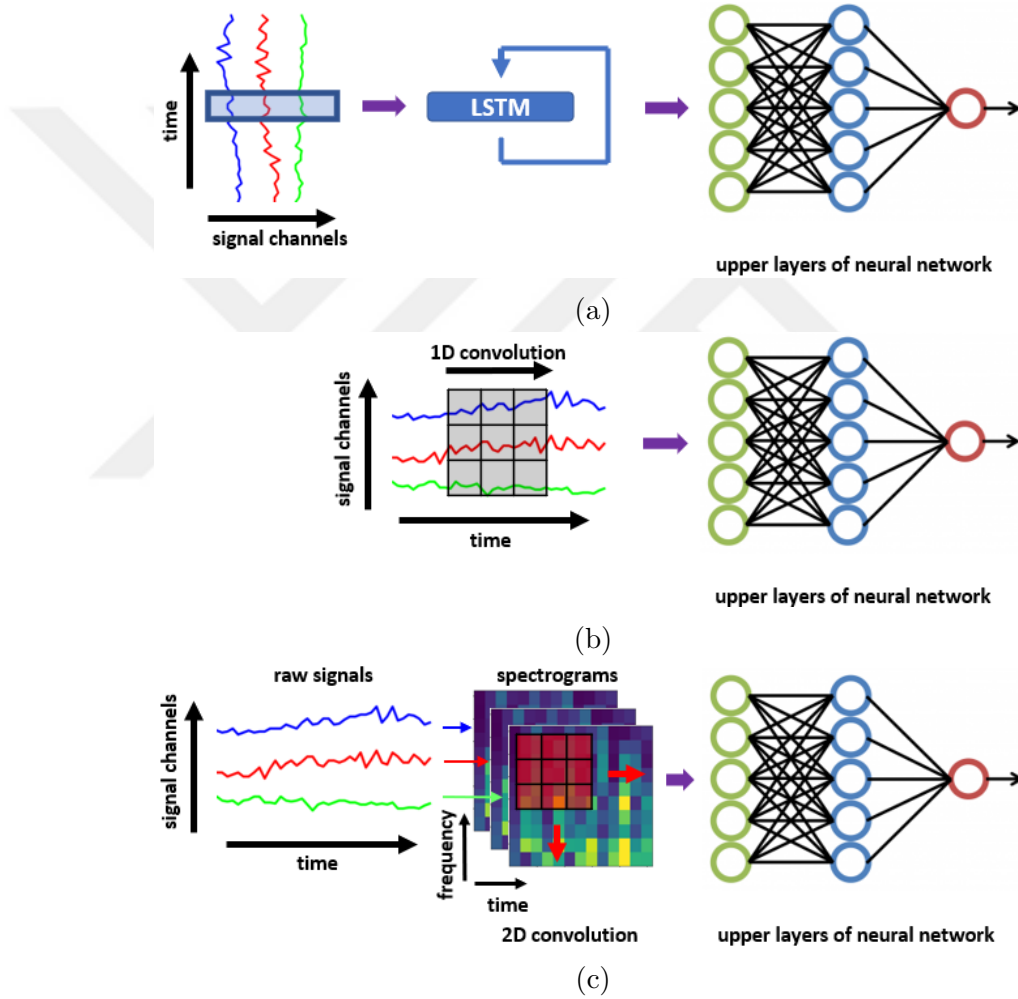


Figure 1.3: Single network models. (a) The LSTM model and the (b) 1D CNN model processing the raw input signals, and the (c) 2D CNN model processing the spectrogram of raw input signals.

other studies which usually use only a single LSTM layer. It compares the standard 1D CNN-LSTM (named as DeepConvLSTM in [33]) proposed there with baseline 1D CNN, but not with a single LSTM model. Reference [34] compares the 1D CNN-LSTM with both single CNN and single LSTM, verifying that this hybrid model classifies the activities more accurately compared to these two single models. The study reported in [35] uses three 1D CNN layers and a single recurrent neural network (RNN) layer. Different variations of RNNs are implemented in that study which are LSTM, Gated Recurrent Unit (GRU), and bi-directional LSTM (BiLSTM). GRUs are a gating mechanism in RNNs [36] similar to LSTM with a forget gate but has fewer parameters than LSTM since they lack an output gate. BiLSTM is a sequence processing model comprising two LSTMs, one taking the input in the forward and the other in the backward direction. The study compares the results with single 1D CNN and LSTM models. It is shown that while the hybrid model with LSTM layer achieves better accuracy compared to a single 1D CNN and LSTM, hybrid models with GRU and BiLSTM do not improve the single model performances.

Table 1.1: Existing standard 1D CNN-LSTM hybrid model architectures.

Reference [10]	Reference [31]	Reference [32]	Reference [33]	Reference [34]	Reference [35]
1D CNN	1D CNN	1D CNN	1D CNN	1D CNN	1D CNN
1D CNN	pooling	1D CNN	1D CNN	1D CNN	pooling
1D CNN	flatten	dropout	1D CNN	dropout	1D CNN
1D CNN	LSTM	pooling	1D CNN	pooling	pooling
dropout	softmax	flatten	LSTM	LSTM	1D CNN
pooling		LSTM	LSTM	dropout	pooling
flatten		dropout	softmax	softmax	LSTM
LSTM		dense			dense
dropout		softmax			batch
dense					softmax
softmax					

As examples of some non-standard models (not included in Table 1.1), [25] proposes a hybrid architecture composed of GRU and CNN networks. The proposed architecture outperforms four ML algorithms and three variants of the proposed method. Reference [37] uses 1D CNN and LSTM separately to extract

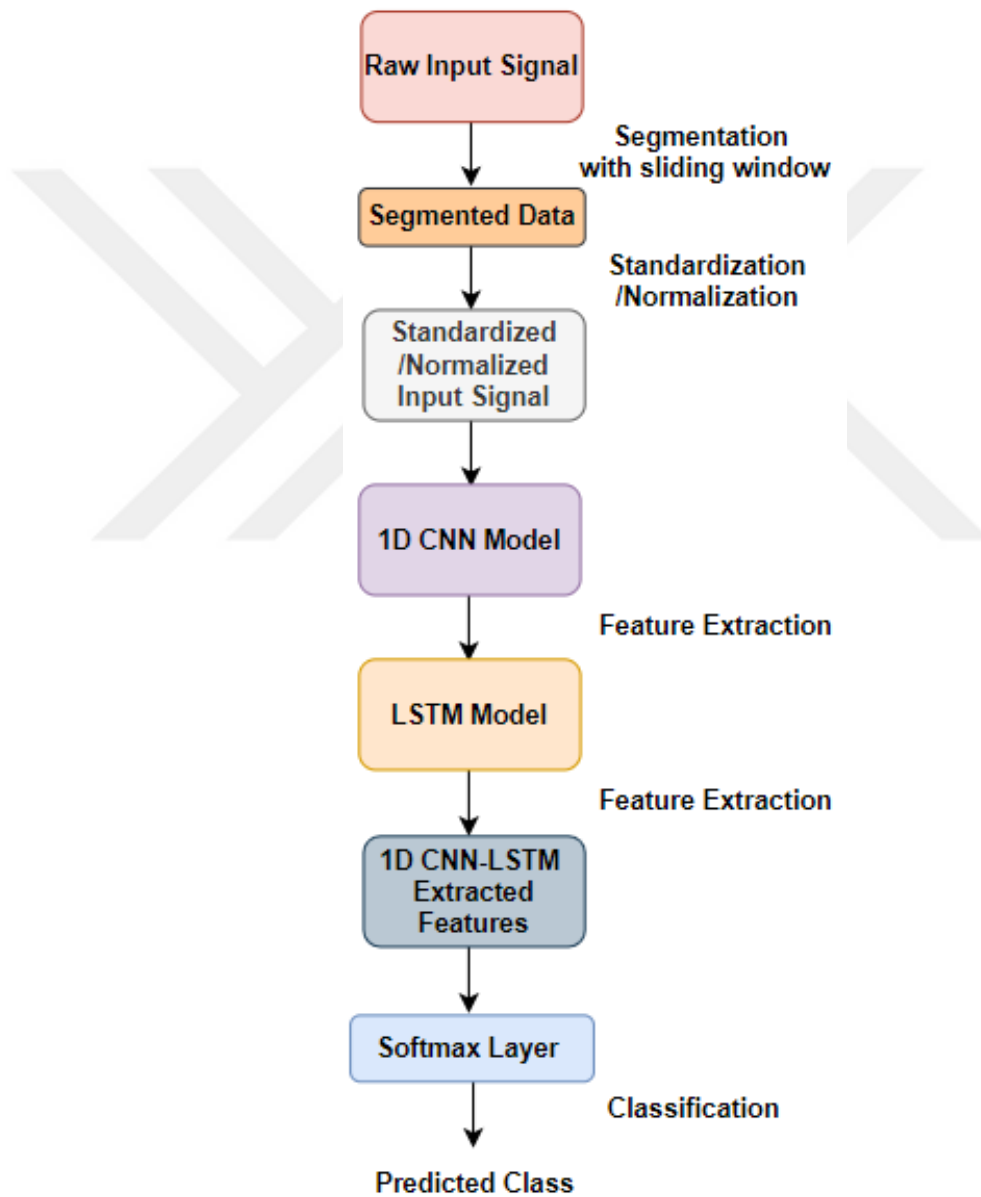


Figure 1.4: The standard 1D CNN-LSTM network architecture.

features from raw data which are then concatenated. This study shows that using two different models enables better feature extraction compared to employing two identical models (both being LSTMs or both being 1D CNNs) for the same purpose.

Reference [38] combines CNN-extracted features with hand-crafted features to improve the activity recognition performance. In [39], the authors employ a LSTM layer following a CNN layer to recognize complex activities while they use a single CNN model for simple activities. Reference [40] uses majority voting over three different DL models to classify the activities. Most of the previous studies have compared hybrid models with either single CNN or single LSTM models (but not both) despite that a comparison with both would have been more comprehensive in terms of verifying the superiority of hybrid models. Furthermore, investigating different ways of combining CNN and LSTM networks could enhance the performance even further. However, this issue is not considered at all in the above-mentioned studies.

1.2.2 Transfer Learning for Human Activity Recognition

Training a DL model is usually a computationally intensive and time-consuming process. Moreover, it requires acquiring large amount of labeled data which is a challenging task. By considering the pervasiveness of DL applications and the above-mentioned problems, it is not practical to train a separate model from scratch for every new DL task. Transfer learning (TL) aims to address these challenges by utilizing previously learned knowledge in some domain to train a model in a similar domain [41]. These domains are called the *source* and the *target* domains, respectively. Therefore, we can say that TL enables to facilitate a learning task in the target domain by using the knowledge from the source domain.

Intuitively, this idea is similar to that if a person has learned playing tennis previously in his life then it will be easier for him to learn playing squash compared to someone who has never played tennis or squash before. Because, tennis and

squash have similar rules and players perform similar activities while playing both games. Therefore, someone who has experience in tennis can transfer his game knowledge from tennis to squash and this knowledge transfer can help him to learn playing squash faster compared to someone who has no tennis or squash experience.

Formal definition of TL can be given as follows [42]:

Definition 1. (*Transfer Learning*). *Let a domain to be represented by $D = \{\chi, P(X)\}$ where χ denotes feature space, $P(X)$ denotes edge probability distribution and $X = \{x_1, \dots, x_n\} \in \chi$. Let a DL task to be represented by $T = \{y, f(x)\}$ where y represents the label space and $f(x)$ represents the label prediction function. Then, for a given learning task T_t based on D_t , TL aims to improve the performance of the predictive function $f_T(\cdot)$ for learning task T_t on the target domain by using the knowledge from D_s and T_s on the source domain.*

There are several different approaches for the categorization of TL methods. These approaches are basically grouped under problem/solution-based, data-based, and model-based interpretations. From the model-based perspective, there are several sub-categories [43]:

Parameter Control Strategy: Initial layers of deep neural networks (DNNs) act as feature extraction layers while the last few layers handle the classification process. Feature extraction is done in a hierarchical structure which means that while the earlier layers extract primitive features, the last few layers extract more complicated and domain specific features. For example, initial layers of image-based models extract features such as edges and corners while the last few layers extract more complicated features such as particular components of objects. When the tasks and problem types are sufficiently similar, different DL models can utilize the same features and feature extraction layers. Parameter control strategy depends on this idea and proposes that for similar tasks, most of the layers of a pretrained artificial neural network on the source domain can be frozen and only the last few layers can be tuned for the task in the target domain. The TL method used in this thesis falls in this category.

Model Ensemble Strategy: For a given DL task, there might be several datasets and models which can be used as source domains. However, combining them directly may not be an efficient method since the distributions of each model might be different. Model ensemble strategy aims to transfer the knowledge from each weak classifier in a way that the classifiers strengthen each other and the ensemble classifier would be more accurate on the target domain.

Deep Learning Techniques: DL models are used for various tasks and applications. One of those tasks is obtaining feature representation of the input samples. Feature representations should be carrying characteristic information about the input samples to be more generalizable. Traditional DL approach commonly uses autoencoders as feature extractors. While the encoder part produces a feature representation, the decoder part aims to reconstruct the original input from the obtained feature representation. Therefore, the objective function contains a term which measures the difference between the input and the output of the autoencoders. For TL applications, two autoencoders are used for the source and the target domains that share the same parameters. The objective function has an additional term which is a difference measure between the feature representations. This term enables to minimize the distribution difference between the source domain and the target domain. Therefore, obtained feature extractor model can produce feature representations which can be used in both domains. As distinguished from the traditional DL approach, adversarial DL approach adapts generative-adversarial network (GAN) based methods. GAN framework has two components which are the generator and the discriminator. While the generator part aims to generate real input like instances, the discriminator part aims to discriminate generated instances from real ones. TL uses generators as feature extractors. The aim is to extract similar features from the source and the target domain instances. On the other hand, the discriminator part aims to differentiate the source domain instances from the target domain instances. As a result, this architecture enables similar feature extraction for both the source and the target domains.

Above mentioned TL methods have been successfully used in different application areas such as medical imaging, biological sequence analysis, recommender-systems, and WiFi localization tasks [43]. HAR is also one of the areas which has utilized TL methods to overcome several challenges in this area. Proposed solutions for the TL related challenges in HAR are mainly categorized under four groups: cross-body part, cross-sensor, cross-user, and cross-activity TL.

Wearable sensor based HAR uses sensor signals of wearable devices such as smart watches and smart phones. Users can carry these devices on different body parts at different positions. However, locations of the sensors can affect the signal patterns significantly which may confuse the activity classifier model. Therefore, applicability of such models in real life requires that a DL model which has been trained on a particular body part and location can transfer this knowledge to the other body parts and locations which would enable accurate recognition of activities even when the sensors are carried on different body parts and locations. Reference [44] compares cascade learning to end-to-end learning in various experiments for TL across sensor locations on different body parts. Reference [45] proposes stratified TL which transforms the source and the target domains into the same feature space. Therefore, data distributions of the source and the target domains become similar and enable knowledge transfer for activity labeling.

Collecting and labeling data is a challenging and time consuming task. It is not always possible to find sufficient amount of labeled data in a particular domain. Therefore, it is valuable to transfer knowledge from a domain which has large amount of data to a domain which has insufficient amount of data. Collecting data from different sensor types for HAR has various difficulties. Hence, transferring knowledge across different sensor types has been studied. Reference [46] uses bi-directional generative adversarial networks to extract a common feature representation for different sensor types. Reference [47] aligns the feature spaces of different sensors which enables knowledge transfer with common feature representations. Autoencoders are used as stochastic feature extractors. Common feature representation is obtained by minimizing the Kullback–Leibler divergence between the outputs of two encoders which are trained on the source and target

domains, respectively.

Some differences such as age, gender, height, weight between the users affect the way they perform activities. Therefore, DL models trained on a group of people may not perform well enough on new users. This problem is studied to be solved by successful cross-subject knowledge transfer. Reference [48] proposes to use a GAN model to generate artificial data which have similar feature space distribution with new users' data. The DL model trained on this artificial data performs improved performance when it is tested on new users. Reference [44] states that the initial layers of DL models extract primitive features which are common among different users. Therefore, initial layers of the model from the source domain are copied as initial layers of the model on the target domain and only the last few layers are trained. Since the number of layers to be trained is reduced, this approach enables training on devices with low computational capabilities.

Different human activities may have some similar patterns which enables knowledge transfer across different activities. Reference [49] finds a similarity function between different activities and use this similarity function to produce pseudo training data. Labeled source domain activities are utilized as pseudo training data for training a model in the target domain for different set of activities. The study reported in [44] transfers initial layer weights and trains only the last few layers for new activities as it does for cross-user TL. Reference [50] shows that transferring pretrained features from the source domain to the target domain can reduce the training time significantly.

1.2.3 Computationally Efficient DL Models for Human Activity Recognition on Resource Limited Devices

Developments in the sensor, communication, and ubiquitous computing technologies enabled use of vast amount of IoT sensors and devices in many areas such as ambient intelligence, healthcare, transportation, and industrial IoT [51]. As a

result, massive data which carry valuable information are acquired. Since most IoT devices have limited computational capacities, required computations for the processing of raw data are usually not employable on IoT devices. Offloading the computations to the cloud servers has the disadvantages of increased network traffic and latency. On the other hand, 5G and beyond communication systems aim to enable real-time communication and computation systems for IoT devices [52]. Therefore, enabling technologies for deploying required computations on edge devices rather than cloud servers are needed. For this purpose, we need to develop computationally efficient DL models for resource limited IoT devices such as wearables.

Studies for developing computationally efficient DL models for HAR can be categorized mainly under two groups: developing shallow networks and developing networks with less precise (quantized) model weights. Shallow networks aim to minimize the number of layers and the number of neurons of a network while keeping the model accuracy high. The challenge of this method is that the learning capacity of a network is proportional to the neuron and layer numbers. Therefore, it may not be always possible to learn some tasks in some datasets with small number of layers and neurons. Weight quantization method aims to store model parameters by using a lower number of bits in computer memory. Therefore, 1,2,4,8, or 16-bit quantized representations are used for storing model weights instead of the standard 32-bit floating point number representation.

Wearable motion sensor based HAR studies mostly focus on developing shallow network structures for enabling the models on resource limited devices. Resource constraints are considered in terms of memory space, computation time, and battery life. Reference [53] proposes a lightweight CNN model by using low dimensional filters and deploys local loss function to train the network. The work in [54] uses an LSTM model with only two hidden layers and 30 neurons making it deployable on memory limited edge devices. Reference [55] proposes a shallow CNN structure which considers cross-channel communication. Each channel can communicate with other channels by a graph neural network and this enables extracting more discriminative features and eliminating the redundant features. Therefore, CNN model can extract the features with a more shallow structure,

requiring less memory space.

The work presented in [56] focuses on reducing the computational time of a CNN model by removing the pooling layers. This study adds strides to the CNN layers for downsampling instead of pooling layers and presents that this method reduces the computational time noticeably. Reference [24] proposes combining hand-crafted features with DL extracted features. Since the hand-crafted features provide a baseline, a shallow DL model which extracts shallow features enable effective activity recognition performance. Since the model does not have deep layers, it achieves low computational complexity. Reference [57] proposes that the use of spectrogram signals rather than raw signals helps reducing the model complexity. This study uses a 1D CNN model which shares the same filter weights for three axes of a sensor in order to reduce the computational complexity. Reference [58] proposes extracting power spectral density feature vectors by calculating magnitude squared of short-time Fourier transformation (STFT) outcomes. Extracted feature vectors are given as input to an LSTM model. Computational complexity of LSTM models is proportional to the number of time steps in the input signal. Since the STFT operation reduces the number of time steps in the input signal, the proposed method enables lower complexity compared to the raw signal-based LSTM models.

Reference [59] proposes training a bidirectional LSTM (BLSTM) model by using binary values as model weights instead of 32-bit floating point numbers. Computational time of the binary model is approximately four times less than the full model despite that the required memory space for a single parameter value is reduced by 16 times. The reason might be that the binarized model might have needed higher number of parameters in order to catch the performance of the full precision model. The work presented in [60] proposes a novel two-bit quantization method for a pretrained 1D CNN model. This study achieves similar activity recognition performance with the full precision model while occupying approximately 11 times less memory space and requiring nine times less inference time compared to the full precision model.

1.3 Contributions and the Outline of the Thesis

This thesis makes contributions in three respects: (i) enhancing the classification accuracy of daily and sport activities, (ii) successful feature transfer between the activity and user identity recognition domains, and (iii) enabling proposed DL models on memory limited devices.

In the present chapter, we summarize recent studies in the above-mentioned areas. In Chapter 2, we propose a novel 2D CNN-LSTM architecture to enable enhanced accuracy on the recognition of daily and sports activities by using wearable motion sensors. The proposed model achieves improved accuracy by extracting diverse and more representative features. In the preprocessing step, we obtain spectrogram signals, which have not been well investigated for wearable sensor based HAR, and use them as the second input besides the raw time series signals. In order to make a fair performance comparison, we implement five more commonly used DL models which are three single (1D CNN, 2D CNN, LSTM) and two hybrid (standard 1D CNN-LSTM and an alternative 1D CNN-LSTM) models. We optimize the hyper-parameters of all six models and test them on two publicly available datasets which are the UCI HAR and the DSA Datasets. We measure the model performances by using four different performance metrics which are the accuracy, f_1 score, precision, and recall. Also, we measure the model complexities by using the number of parameters of the models, training times, and memory spaces occupied by the models. The proposed 2D CNN-LSTM architecture achieves average accuracies of 95.66% and 92.95% on the UCI HAR and DSA Datasets, respectively. These accuracies are 2.45% and 3.18% greater than the second best model which is the 2D CNN model. While achieving this performance enhancement, we keep the complexity of the proposed model comparable with the other models.

User identity recognition is a similar task to HAR where wearable motion sensor signals are used to recognize the user of the wearable device. In Chapter 3, we investigate if the features extracted by the proposed 2D CNN-LSTM model can be used for the user identity recognition problem and vice versa. While some

activities provide useful signals for the user identity recognition task, some others may not be as useful. Therefore, we initially select a group of activities for accurate user identity recognition and train a simple binary classifier model to classify sensor signals as useful or not useful for user identity recognition. Proposed model transmits the useful signals to the next module which is used to recognize the user of the device. In this module, we use the feature extractor layers of proposed model which we have trained for the activity recognition problem. Therefore, we only train the classifier part of this model for the user identity recognition task. We show that eliminating the training phase of the feature extractor layers of the proposed model shortens the training process approximately by 17 times and 9 times for the UCI HAR and DSA Datasets, respectively, without a significant accuracy degradation compared to the full models where all of the layers have been trained from scratch for the user identity recognition problem. In the second part of this chapter, we train all layers of the proposed model for the user identity recognition task. Then, we freeze the feature extractor layers and transfer them to the activity recognition model. We train only the last few layers of the model for the activity recognition task. We show that the training process is shortened by approximately 15.6 times and 8.7 times for the UCI HAR and DSA Datasets, respectively, with quite a small accuracy loss compared to the full models.

Proposed 2D CNN-LSTM architecture achieves enhanced accuracy on activity and user identity recognition tasks. Moreover, it enables successful feature transfer between the two domains and decreases the training times dramatically. Chapter 4 investigates how to deploy this model on a memory limited device. Initially, we inspect the main factors increasing the size of the proposed model. Afterwards, we add pooling layers to the proposed models and quantize the obtained model weights to eight bits by using the TensorFlow library. We apply these methods on the six proposed models for activity recognition, user identity recognition, and binary classifier models on the UCI HAR and the DSA Datasets. As a result, we reduce the sizes of the proposed models by 17 to 116 times without a significant degradation in the classification accuracies.

In summary, we propose a novel 2D CNN-LSTM architecture for the human activity and user identity recognition problems. Proposed architecture achieves

improved accuracy on these tasks. We reduce the size of the model by using pooling layers and eight-bit weight quantization methods. Hence, we make the proposed model ready to be deployed on resource limited devices.



Chapter 2

Enabling Diverse Feature Extraction: A Novel 2D CNN-LSTM Architecture

2.1 The Proposed Methodology

We propose a hybrid model where a 2D CNN and a LSTM network are combined in parallel, as illustrated in Fig. 2.1. Since LSTMs are favored for time-series data and CNNs are more suitable for processing image data, we feed the LSTM branch with the raw time-series sensor readings and the CNN branch with the spectrogram of the raw sensor recordings. While LSTM focuses on the time-dependent patterns in the data, CNN considers the frequency patterns as well so that each model can extract features representing different characteristics of the signal, conveying complementary information. By combining such features, we can obtain better feature representation of the signal, resulting in improved activity classification performance. Proposed model is different from the standard 1D CNN-LSTM which is illustrated in Fig. 1.4 in terms of having two different DL models in parallel branches where each one receives the input in a different form as appropriate for the model characteristics.

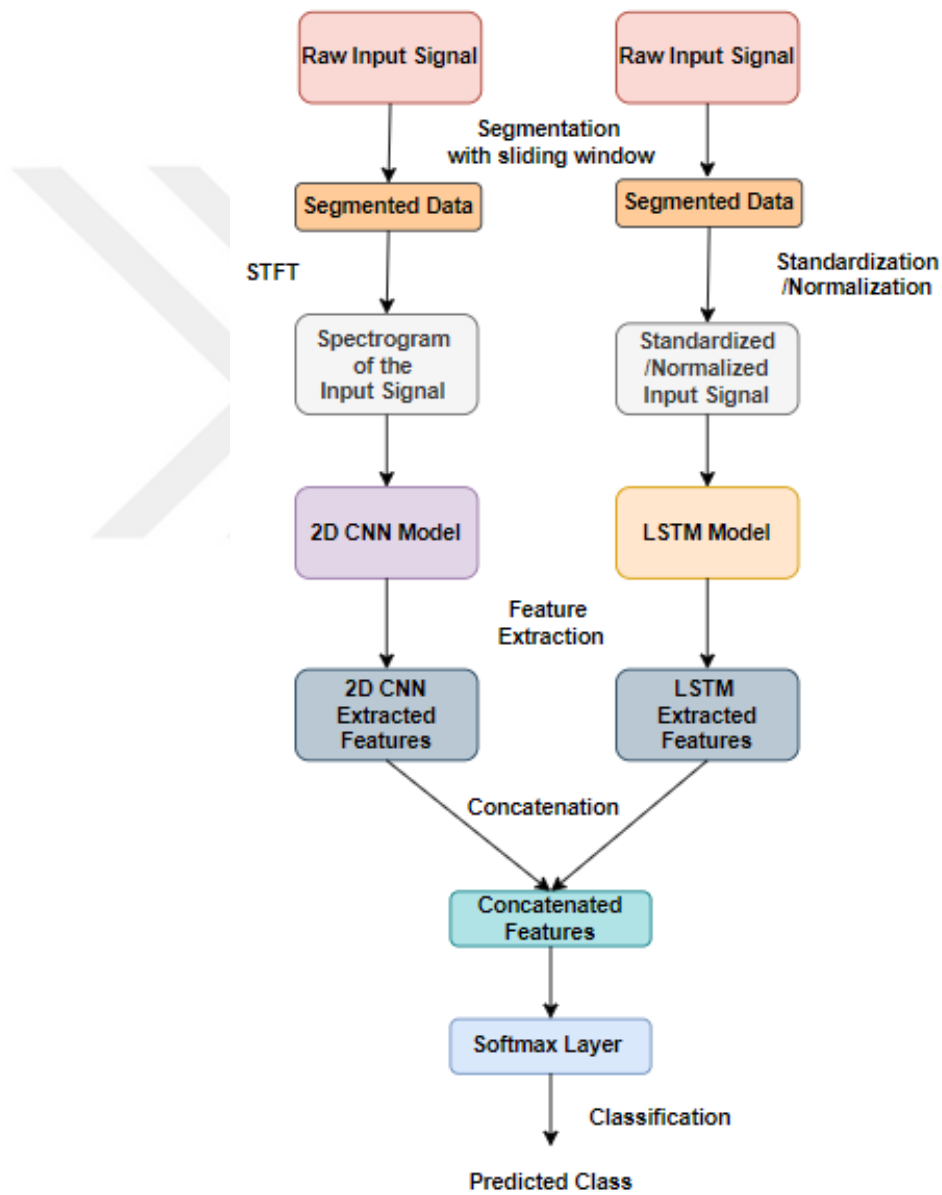


Figure 2.1: The proposed 2D CNN-LSTM hybrid architecture.

2.1.1 Datasets

In this subsection, we briefly describe the two publicly available datasets that we have employed in this study.

1) UCI HAR Dataset: 30 participants between the ages 19 and 48 perform six activities while the accelerometer and gyroscope sensors of a smartphone (Samsung Galaxy S2), carried on their waist, records the data [61, 62, 63]. Recorded activities are walking on a flat surface, walking upstairs, walking downstairs, sitting, standing, and laying. The number of samples in each activity class are comparable but not equal, causing this dataset to be slightly imbalanced. The accelerometer captures tri-axial linear acceleration while the gyroscope detects tri-axial angular velocity (or rate) at a sampling frequency of $f_s = 50$ Hz. Both the total acceleration and the body component of acceleration are provided in this dataset. Body components are obtained by subtracting the gravitational component from the total acceleration. Signals are segmented by using an overlapping sliding window with a length of 128 readings/window. Each time segment contains data from nine channels in parallel: x, y, z axes of (i) accelerometer total components, (ii) accelerometer body components, and (iii) gyroscope angular velocity components. The acquired dataset has been randomly partitioned into two, where data from 70% of the subjects are used for training and data from the remaining 30% are employed for testing.

2) Daily and Sports Activities (DSA) Dataset: Eight subjects (four male, four female) between the ages 20 and 30 perform 19 different daily and sports activities [64, 65]. During the experiments, the subjects wear five sensor units placed on their chest, right/left arm, and right/left leg. Each sensor unit contains three tri-axial sensors: accelerometer, gyroscope, and magnetometer. These three sensor types capture signals in the x, y, z axes at a sampling frequency of $f_s = 25$ Hz. Hence, there are 45 signal channels (5 sensor units $\times$ 3 sensor types $\times$ 3 axes). The subjects perform each activity for five minutes after which the recorded signals are divided into non-overlapping 5-sec segments. The activity classes are balanced since there is an equal number of samples from each

activity type.

It is notable that the UCI HAR Dataset contains a small number of activities (six) performed by a large number of participants (30) whereas the DSA Dataset contains a large number of activities (19) performed by a smaller number of participants (eight). The UCI HAR Dataset contains 269 MB data while the DSA Dataset comprises 402 MB data.

2.1.2 Preprocessing

1) Preprocessing the Data for the 1D CNN and LSTM Branches of the DL Models: Since each sensor type provides output in different ranges with different units, the signals can be normalized or standardized. Reference [10] employs standardization for the UCI HAR Dataset and [66, 67, 68, 69] employ normalization for the DSA Dataset. For compatibility with these related studies, we standardized the signals in the UCI HAR Dataset and normalized those in the DSA Dataset, considering each signal channel separately.

We standardized the UCI HAR Dataset as follows:

$$x[n]_{\text{standardized}} = \frac{x[n] - \bar{x}[n]}{\sigma_x}$$

where $x[n]$ is the data sequence in one time segment of data, $\bar{x}[n]$ is the average value of $x[n]$, and σ_x is the standard deviation of $x[n]$. Standardized sequences have zero mean and unit variance.

For the DSA Dataset, we modified the normalization formula by subtracting the average value of the sequence from the sequence instead of its minimum value. Thus, we normalized the data using the following equation:

$$x[n]_{\text{normalized}} = \frac{x[n] - \bar{x}[n]}{x_{\max} - x_{\min}}$$

Here, $x_{\min}$ and $x_{\max}$ are the minimum and maximum values of the sequence $x[n]$, respectively.

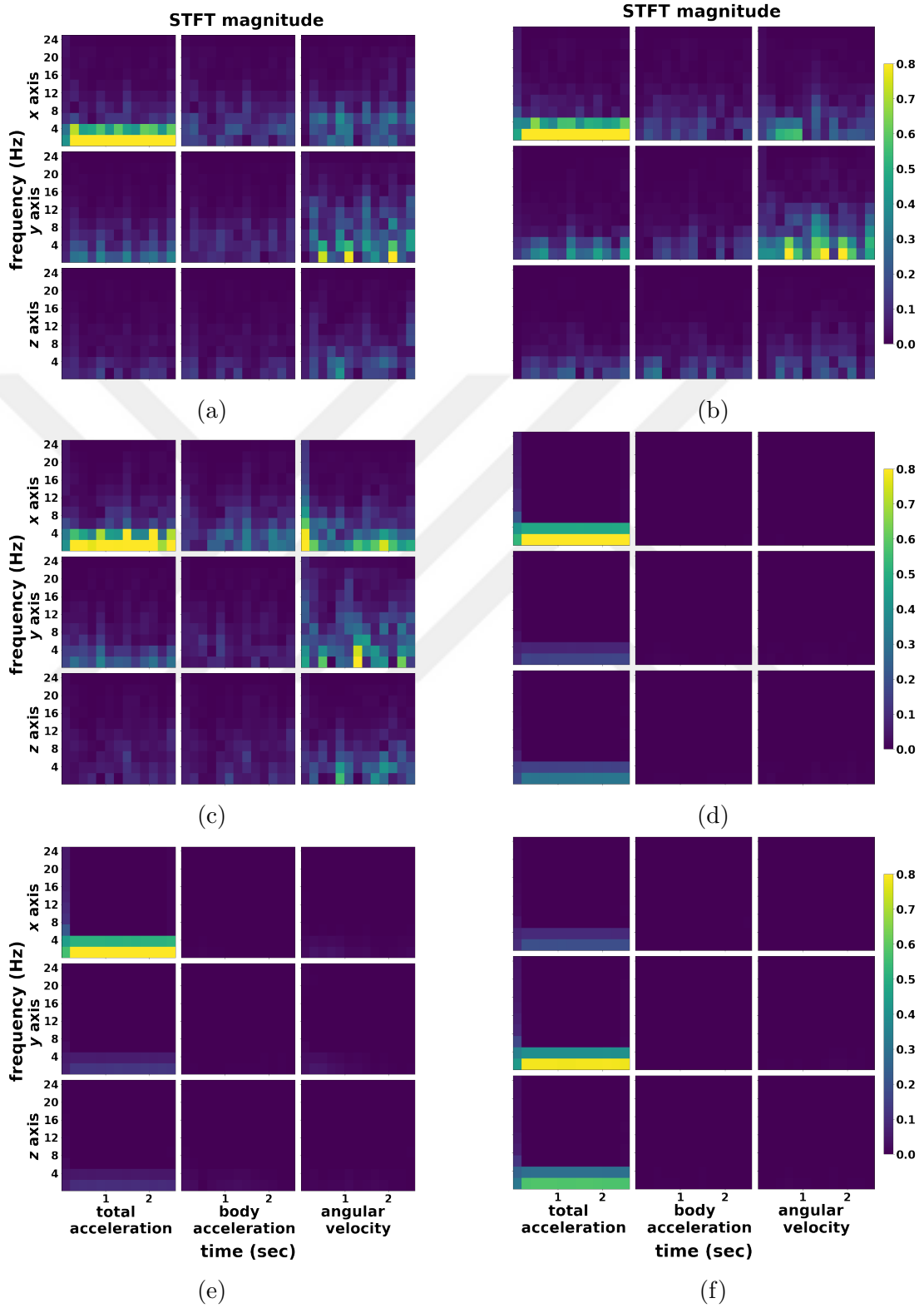


Figure 2.2: Sample spectrograms obtained from the UCI HAR Dataset for the activities (a) walking on a flat surface, (b) walking upstairs, (c) walking downstairs, (d) sitting, (e) standing, and (f) laying.

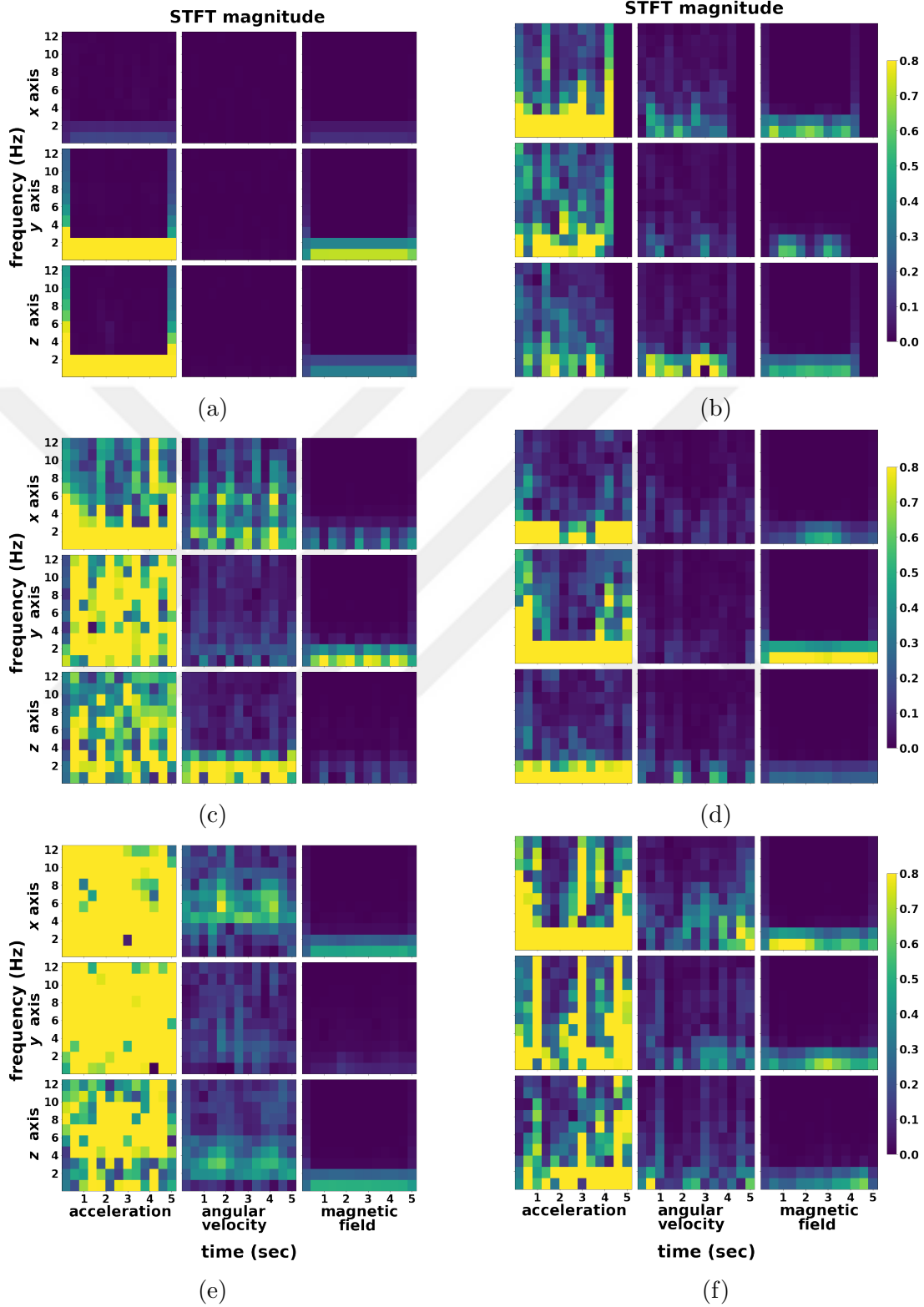


Figure 2.3: Sample spectrograms obtained from the DSA Dataset for the activities (a) lying on the right side, (b) ascending stairs, (c) walking on a treadmill at a speed of 4 km/h in flat position, (d) rowing, (e) jumping, and (f) playing basketball.

2) Preprocessing the Data for the 2D CNN Branches of the DL Models: We did not standardize/normalize the data for the 2D CNN because these operations remove the zero frequency (DC) component of the signal, which is not desirable. We directly convert the data in each time segment of each signal channel into a spectrogram by taking the magnitude of its STFT [70]. To do this, we first padded the beginning and the end of each time segment with zeroes to obtain a total of 140 samples per time segment for both datasets. Based on the 140 samples, we have obtained 14 subsegments of 20 samples each, with 50% overlap between consecutive subsegments. We took the Discrete Fourier Transform (DFT) of each subsegment to get 11 frequency components in the interval $[0, \frac{f_s}{2}]$ with frequency increments of $\frac{f_s}{20}$ where f_s is the sampling frequency. As a result, the spectrogram of each time segment has dimensionality 11×14 for both datasets. Thus, the time segments of the two datasets with dimensionalities 128×9 and 125×45 are respectively transformed to spectrograms with dimensionalities $11 \times 14 \times 9$ and $11 \times 14 \times 45$. Sample spectrograms obtained from the two datasets are provided in Figs. 2.2 and 2.3.

2.1.3 Implementation and Resources

We have implemented all six models using the Keras and Tensorflow libraries in Python. Since tuning the hyper-parameters and processing the datasets are computationally intensive, we used the GPUs of two different platforms to expedite the computations. These are the Amazon Web Services (AWS) platform with EC2 p2.xlarge GPU for tuning the hyper-parameters and the Google Colab platform that provides 12 GB NVIDIA Tesla K80 GPU for testing the final models.

2.1.4 Layer Selection

There are two ways to choose the layers while making a performance comparison between the different DL models. The first option is to optimize the number

of each layer type for each model. While this scenario appears to be optimal, it brings an uncertainty about whether the performance differences among the different models are caused by different layer types or different numbers of layers. The alternative is to fix the type and the number of layers for each model and only change the type of a single layer as appropriate to each model. In this scenario, performance differences are only the result of different layer types employed in each model. To be able to test our proposed model in both cases, we used the UCI HAR Dataset for the first scenario and the DSA Dataset for the second scenario.

In processing the UCI HAR Dataset, we optimized the number of layers in each type of network before selecting the other hyper-parameters. In these experiments, we observed the effect of changing the number of layers while keeping the other hyper-parameter values fixed. Model layers are displayed in Figs. 2.4 and 2.5. We obtained the optimum number of layers as four for 1D CNN, two for 2D CNN, three for LSTM, and three for the standard 1D CNN-LSTM. The optimal numbers of fully connected (dense) layers are one, two, one, and one, respectively. Since using a pooling layer did not improve the performance of 1D CNN and 2D CNN models, we did not use pooling layers in the networks developed in this study. Because the alternative 1D CNN-LSTM and the proposed 2D CNN-LSTM model are composed of single networks whose layers are already optimized as described above, we did not conduct a separate layer selection process for them. We used dropout layers as regularizers in between two layers which are placed before the fully connected layers in all six models.

In processing the DSA Dataset, we fixed the total number of layers in each model and only changed the type of the first layer. The first layers used in the respective models are 1D CNN, 2D CNN, LSTM, 1D CNN and LSTM in series, 1D CNN and LSTM in parallel, 2D CNN and LSTM in parallel. Model layers are illustrated in Figs. 2.6 and 2.7. Since the remaining layers of the networks are identical, differences in the results with this dataset are the consequence of including different types of first layers.

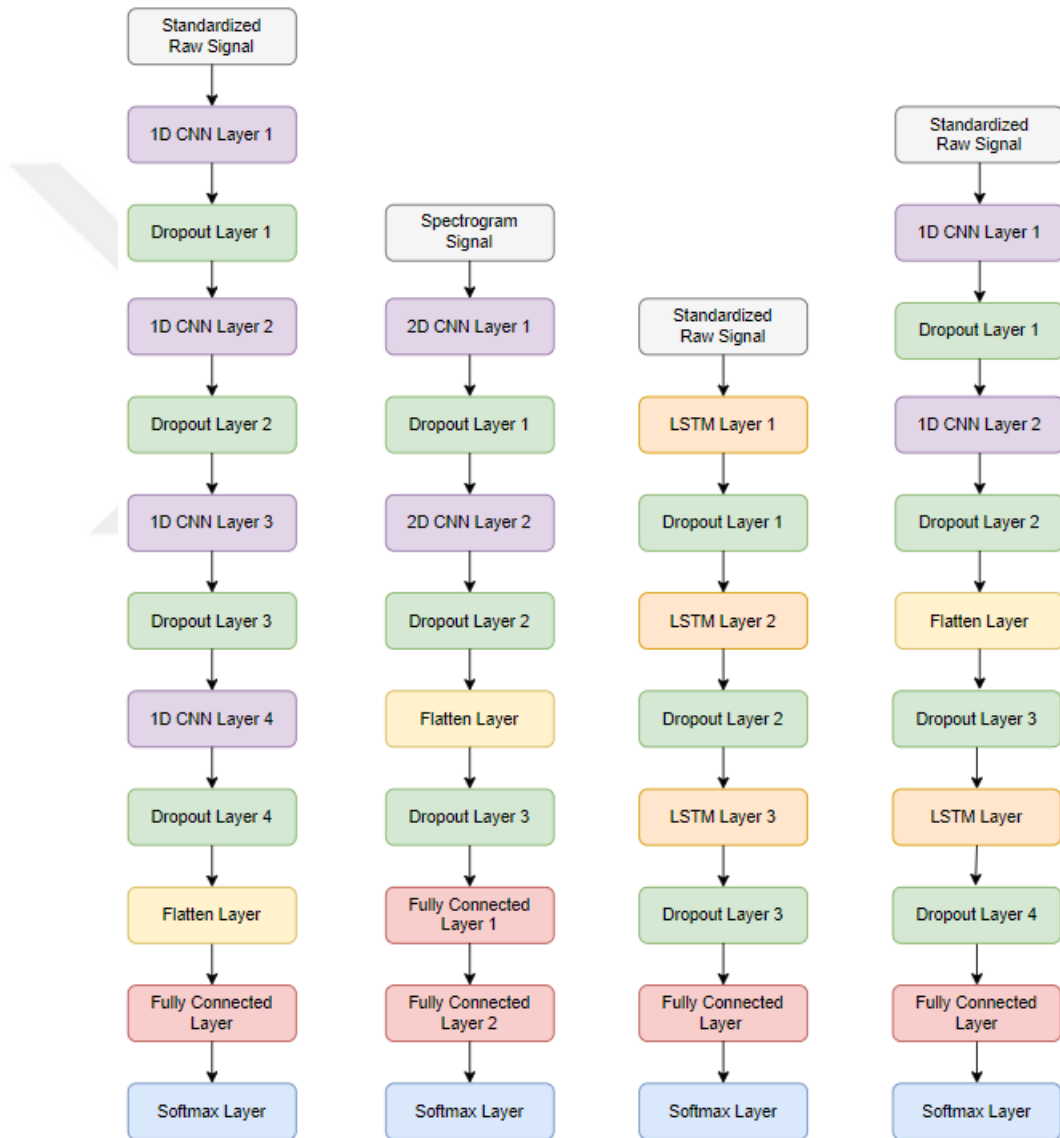


Figure 2.4: Layers of the 1D CNN, 2D CNN, LSTM, and the standard 1D CNN-LSTM models used in processing the UCI HAR Dataset.

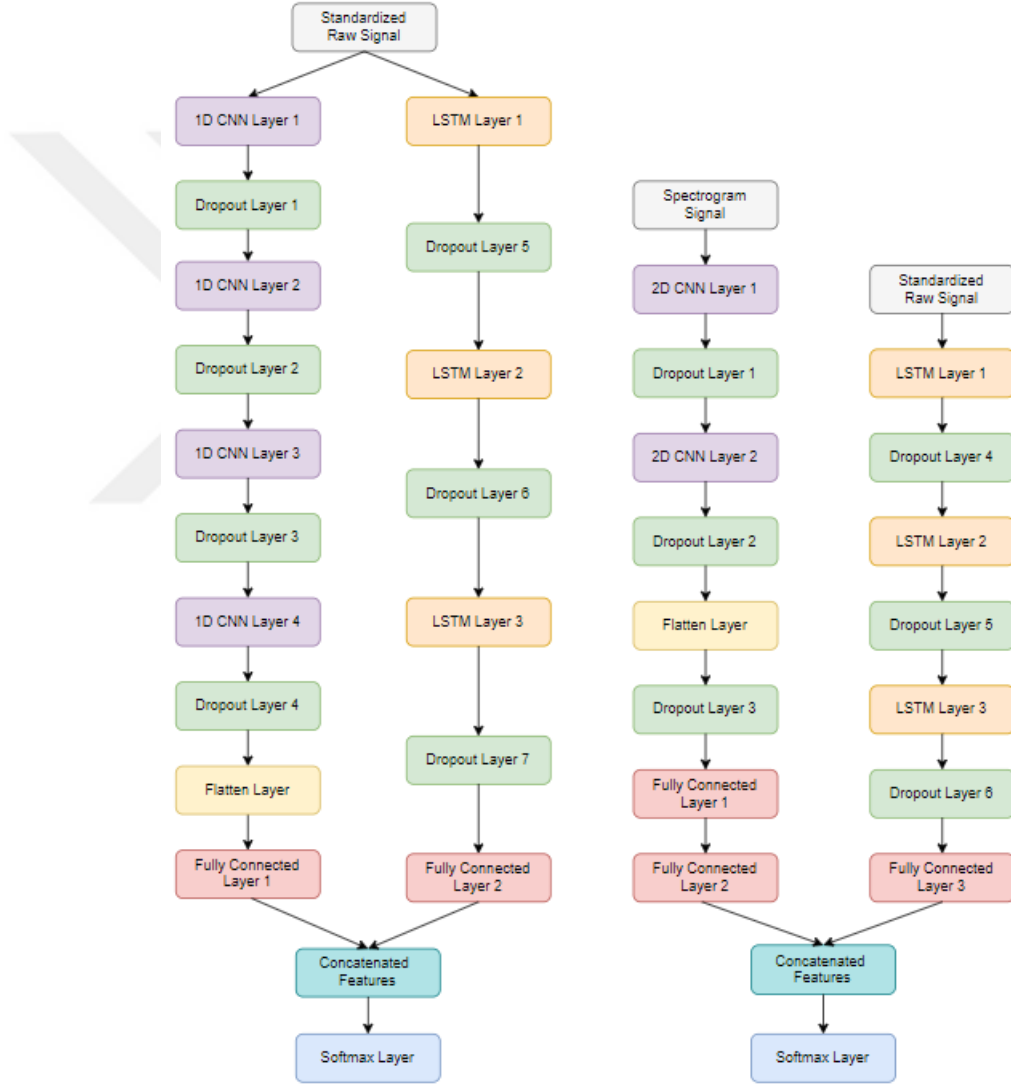


Figure 2.5: Layers of the alternative 1D CNN-LSTM and the proposed 2D CNN-LSTM models used in processing the UCI HAR Dataset.

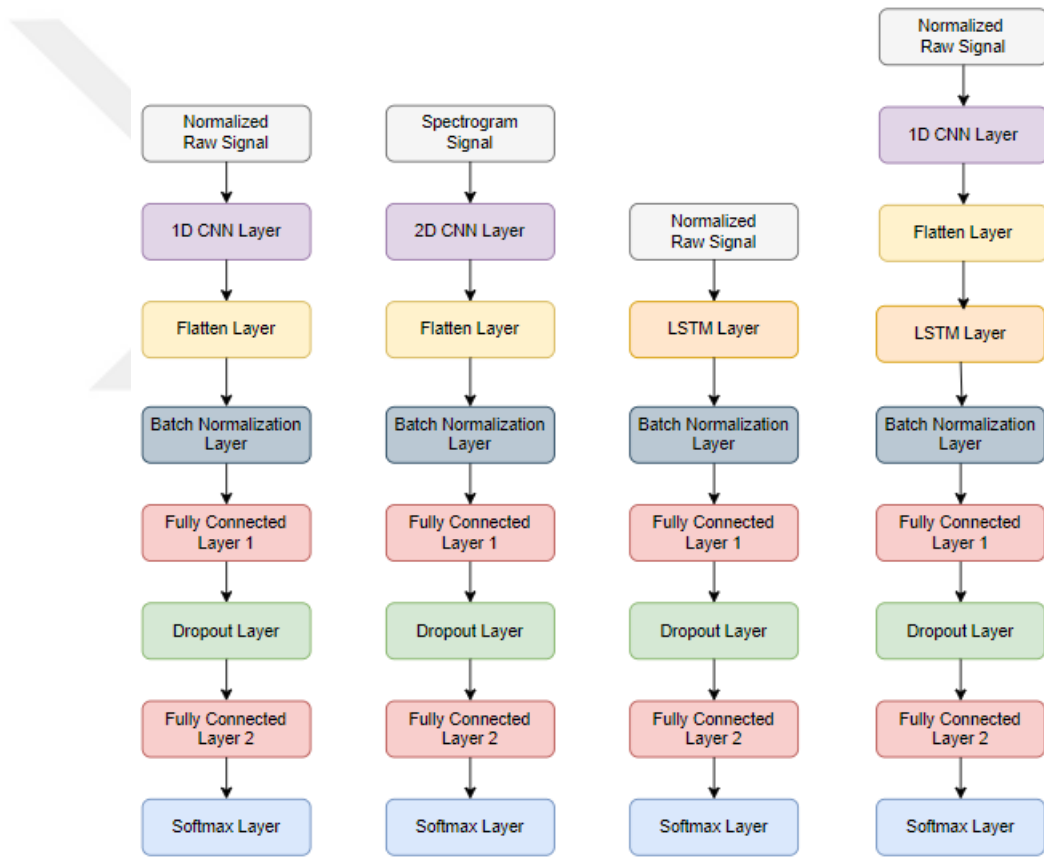


Figure 2.6: Layers of the 1D CNN, 2D CNN, LSTM, and the standard 1D CNN-LSTM models used in processing the DSA Dataset.

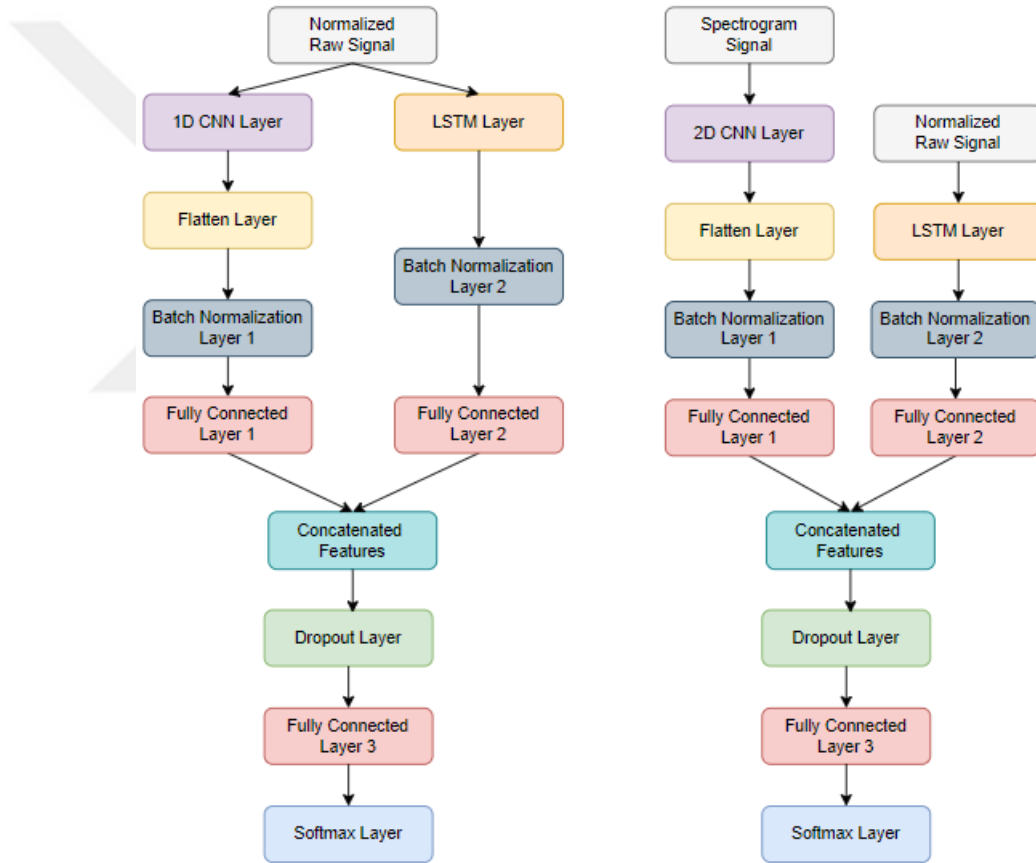


Figure 2.7: Layers of the alternative 1D CNN-LSTM and the proposed 2D CNN-LSTM models used in processing the DSA Dataset.

2.1.5 Hyper-Parameter Selection

After determining the number of layers in each type of network, we need to select the other hyper-parameters. Although hyper-parameter selection is an on-going research area, there are some common practices, one of which is grid searching. However, when the number of hyper-parameters is large, the time required for grid searching grows exponentially. Finding the optimal hyper-parameters through this computationally intensive process takes a long time.

Another approach is selecting the hyper-parameters sequentially or one at a time. In this method, the value of only one hyper-parameter is changed over an interval. After the best value is selected for that hyper-parameter, it is kept fixed. Then, the next hyper-parameter is tuned in a pre-determined interval. Since many hyper-parameter combinations are not considered, this method is not optimal but computationally less intensive.

The third method is Bayesian optimization [71] used in Reference [10] for HAR. This is the method that we have adopted for hyper-parameter selection. We have used Scikit-Optimize library in Python for the necessary implementation tools. We have employed EC2 p2.xlarge GPU instance in the AWS platform and the “gbdt_minimize” function from Scikit-Optimize library for hyper-parameter tuning using Bayesian optimization. Before the optimization process, the set of model hyper-parameters to be optimized and their initial values are provided as input to the function. During the optimization process, based on the model performance for previously selected hyper-parameter values, a new set of hyper-parameter values is automatically selected by the algorithm to improve the model performance. The total number of repetitions for this process is chosen as 50 for each model, taking into account the time and the computational resources. Thus, we have considered 50 hyper-parameter combinations for each model to determine the optimum set of hyper-parameter values that provide the best performance for that model.

There are some common practices on the choice of some of the hyper-parameter values for the network layers. Among these hyper-parameters are the optimizer,

loss function, and the activation function which we have selected directly, without tuning. We have employed “Adam” as the optimizer and “Categorical Cross Entropy” as the loss function for all models. We have used the sigmoid activation function for the fully connected layers and the ReLU activation function for the 1D CNN layers of the models developed for processing both datasets, as well as the 2D CNN layers used for processing the UCI HAR Dataset. However, excluding activation functions in the 2D CNN layers while processing the DSA Dataset resulted in better classification accuracy. Therefore, we did not use any activation function for those layers. Also, the last fully connected layers of all six networks developed for processing the DSA Dataset do not have any activation functions.

2.1.6 Training, Validation, Testing Methods, and Performance Analysis

Ideally, a large dataset should be divided into three parts or three separate datasets should be used for the training and testing process of a DL model: training, validation, and test datasets. Training and validation sets should be used in tuning the hyper-parameters to select their optimal values. Then, the final model should be trained on the union of the training and validation sets and the final performance analysis should be done on the test set. Also, each participant’s data should be included in only one of the three sets (training, validation, test sets) to eliminate any bias during the performance analysis. If there is any overlap in the data used during training, validation, and testing, this would not be a fair evaluation since the models would be tested on data similar to what they were trained on. Furthermore, in real-world applications, it is likely for a DL model to encounter new (unseen) data, different than those used for training.

The UCI HAR Dataset is originally partitioned into training and test sets as follows: Of the 30 subjects, data from subjects 2, 4, 9, 10, 12, 13, 18, 20, and 24 are assigned as test data whereas data from the remaining subjects are used for training. Since a validation set is not provided, we partitioned the data acquired

from the 21 training subjects into two. We used data from subjects 1, 3, 5, 6, 7, 8, 11, 14, 15, 16, 17, 19, 21, 22, 23, 25 for training and those from subjects 26, 27, 28, 29, and 30 for validation. While selecting the hyper-parameters, we trained the model on the training set and validated the results for the given set of hyper-parameters on the validation set. After this step, we determine the hyper-parameter values performing the best on the validation set as the optimum ones which are then used in building the final model. We train the final model on the union of the training and validation sets and evaluate its performance on the test set.

Accuracy is one of the commonly used performance measures. However, when the activity classes are not balanced, f_1 score, recall, and precision measures are more appropriate for evaluating the performance. In this study, we used all four performance metrics when processing the UCI HAR Dataset, which is somewhat imbalanced, and employed the accuracy metric for the DSA Dataset which is balanced in that it contains an equal number of samples from each activity class.

When the amount of data is limited, cross-validation techniques can be employed. For processing the DSA Dataset in this study, we employ leave-one-subject-out cross validation (L1SO) which uses all the data except one subject’s data for training and tests the trained model with the data from the left-out subject. The process is repeated for all subjects and the average of the performance metric is calculated over the subjects. L1SO method is preferred to the randomly partitioned multi-fold cross-validation technique where each subject’s data are distributed to both the training and the test set randomly, resulting in a bias against the test set. Consequently, the performance measure does not reflect the actual performance of the model on real data. We used L1SO cross validation for both hyper-parameter selection and testing the final models when processing the DSA Dataset because of its smaller number of subjects.

Since DL models are stochastic, single execution of an experiment can provide misleading results. Therefore, repeating each experiment multiple times (10 times in this study) and reporting the average and standard deviation values would be more appropriate. This was done for L1SO cross validation used with the DSA

Dataset as well where the whole cross-validation process was repeated 10 times.

2.2 Experimental Results and Discussion

2.2.1 Hyper-Parameter Optimization Results

Tuned hyper-parameter values for the proposed 2D CNN-LSTM hybrid model are provided in Tables 2.1 and 2.2 for the two datasets. Those for the other five models that we have implemented in this comparative study are provided in Tables A.I–A.X of the Appendix.

Table 2.1: Optimized hyper-parameters of the proposed 2D CNN-LSTM in processing the UCI HAR Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer 1 filter number	64	[4, 256]
CNN layer 2 filter number	64	[4, 256]
CNN layer 1 filter width	3	[3, 7]
CNN layer 2 filter width	3	[3, 7]
LSTM layer 1 neuron number	64	[8, 512]
LSTM layer 2 neuron number	64	[8, 512]
LSTM layer 3 neuron number	64	[8, 512]
fully connected layer 1 neuron number	64	[8, 512]
fully connected layer 2 neuron number	64	[8, 512]
fully connected layer 3 neuron number	64	[8, 512]
dropout layer 1 probability	0.5	[0.1, 0.9]
dropout layer 2 probability	0.5	[0.1, 0.9]
dropout layer 3 probability	0.5	[0.1, 0.9]
dropout layer 4 probability	0.5	[0.1, 0.9]
dropout layer 5 probability	0.5	[0.1, 0.9]
dropout layer 6 probability	0.5	[0.1, 0.9]
learning rate	0.001	$[10^{-6}, 0.1]$

Table 2.2: Optimized hyper-parameters of the proposed 2D CNN-LSTM in processing the DSA Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer filter number	103	[4, 256]
CNN layer filter width	5	[3, 7]
LSTM layer neuron number	297	[8, 512]
fully connected layer 1 neuron number	197	[8, 512]
fully connected layer 2 neuron number	144	[8, 512]
fully connected layer 3 neuron number	399	[8, 512]
dropout layer probability	0.76497736	[0.1, 0.9]
learning rate	$4.16143569 \times 10^{-5}$	$[10^{-6}, 0.1]$

Total times spent for tuning the hyper-parameters of the models on both datasets are displayed in Tables 2.3 and 2.4. Tuning the hyper-parameters of all six models is a computationally intensive process that took about 29.79 hours for the UCI HAR Dataset and 25.53 hours for the DSA Dataset, totaling to 55.32 hours. Note that hyper-parameter tuning needs to be done only once and the tuned hyper-parameters can be used in (near) real-time applications without tuning at each run of the DL algorithms.

Table 2.3: Total hyper-parameter tuning times for the UCI HAR Dataset.

Model Name	Epoch Number	Number of	Total Time (sec)
		Hyper-Parameter Combinations	
1D CNN	50	50	4,438
2D CNN	50	50	3,659
LSTM	50	50	30,911
standard 1D CNN-LSTM	50	50	3,740
alternative 1D CNN-LSTM	50	50	32,563
proposed 2D CNN-LSTM	50	50	31,926

Table 2.4: Total hyper-parameter tuning times for the DSA Dataset.

Model Name	Epoch Number	Number of	Total Time (sec)
		Hyper-Parameter Combinations	
1D CNN	8×20	50	7,464
2D CNN	8×20	50	4,141
LSTM	8×20	50	11,324
standard 1D CNN-LSTM	8×20	50	10,621
alternative 1D CNN-LSTM	8×20	50	41,724
proposed 2D CNN-LSTM	8×20	50	16,649

2.2.2 Performance Comparison of the Proposed Model with the Five Existing Models

The results of our experiments on the two datasets are tabulated in Tables 2.5 and 2.6. Since the imbalance of the UCI HAR Dataset is not too significant, the four performance metrics for each model are comparable for this dataset. The accuracy performance metric values based on the UCI HAR Dataset (Table 2.5) are consistently higher than those based on the DSA Dataset (Table 2.6) by 2.71–11.31%. This is mainly because there are only six activities to recognize in the UCI HAR Dataset whereas there are 19 in the DSA Dataset. The standard deviations in the two tables are comparable.

The results indicate that the proposed 2D CNN-LSTM hybrid model achieves better accuracy, f_1 , recall, and precision scores compared to the five existing models, namely, 1D CNN, 2D CNN, LSTM, standard 1D CNN-LSTM, and the alternative 1D CNN-LSTM. The proposed hybrid network achieves an average accuracy of 95.66% on the UCI HAR Dataset and 92.95% on the DSA Dataset. These figures are, respectively, 2.45% and 3.18% above the accuracy of the single 2D CNN model that ranks the second in all performance metrics. In fact, the proposed 2D CNN-LSTM hybrid model outperforms both of its individual components (the single 2D CNN and the single LSTM models) considerably for both datasets. For the UCI HAR Dataset, the four performance metrics improve by 1.98–2.52% and by 4.89–5.63% compared to the single 2D CNN and the single

LSTM models, respectively. On the other hand, for the DSA Dataset, the accuracy metric improves by 3.18% with respect to the single 2D CNN and by 14.23% compared to the single LSTM.

When we compare the results of the standard 1D CNN-LSTM model with its individual components (the single 1D CNN and LSTM), we do not observe a similar performance boost. This is mainly because of the inferior performance of the standard 1D CNN-LSTM network compared to the proposed model. The four performance metrics of the standard 1D CNN-LSTM model are below those of the proposed 2D CNN-LSTM hybrid network by 4.04–4.45% for the UCI HAR Dataset whereas its accuracy metric is lower than that of the proposed 2D CNN-LSTM model by 8.67% when the DSA Dataset is processed. The different performances of the two hybrid networks stem from the differences in their network architecture as well as the type of input that they receive and process. The standard 1D CNN-LSTM architecture uses the 1D CNN and LSTM networks consecutively to extract features from only one kind of input signal (the raw signals) while the proposed model employs the 2D CNN and LSTM in separate branches in parallel, each receiving a different form of input (spectrogram and raw signals, respectively).

To determine whether the superiority of the proposed 2D CNN-LSTM network model arises from the use of two different types of input or from its parallel architecture, we also compare it with an alternative 1D CNN-LSTM network [37], which has a parallel structure similar to that of the proposed model. In both the proposed model and the alternative 1D CNN-LSTM network, the parallel running CNN and LSTM subnetworks extract their own features which are then merged and classified with a softmax layer. The main difference is that while the proposed model takes input signals in two different forms, the alternative 1D CNN-LSTM hybrid model receives the raw signals as input for both the 1D CNN and LSTM branches of the network. The results in Tables 2.5 and 2.6 indicate that the performance metrics of the alternative 1D CNN-LSTM model are consistently lower than those of the proposed 2D CNN-LSTM model by 4.47–5.26% for the UCI HAR Dataset and by 8.35% for the DSA Dataset. The performance of the alternative 1D CNN-LSTM is comparable to the performances of its two

Table 2.5: Comparison of the implemented single and hybrid models in terms of their subnetworks, merging method, and performance metrics. We provide the average values of accuracy, f_1 score, recall, and precision metrics plus/minus one standard deviation for processing the UCI HAR Dataset.

	Model Name	Subnetworks	Merging Method	Input Types	Accuracy $\pm$ std (%)	f_1 Score $\pm$ std (%)	Recall $\pm$ std (%)	Precision $\pm$ std (%)
Single Models	1D CNN	N/A	N/A	raw input	90.93 $\pm$ 0.50	91.23 $\pm$ 0.49	91.17 $\pm$ 0.48	91.47 $\pm$ 0.48
	2D CNN	N/A	N/A	spectrogram	93.21 $\pm$ 1.60	93.18 $\pm$ 1.65	93.15 $\pm$ 1.64	93.67 $\pm$ 1.41
	LSTM	N/A	N/A	raw input	90.03 $\pm$ 0.68	90.41 $\pm$ 0.69	90.30 $\pm$ 0.69	90.76 $\pm$ 0.66
	standard	1D CNN	consecutive	raw input	91.21 $\pm$ 0.57	91.43 $\pm$ 0.58	91.39 $\pm$ 0.60	91.61 $\pm$ 0.52
Hybrid Models	1D CNN-LSTM	LSTM						
	alternative	1D CNN	parallel	raw input + raw input	90.40 $\pm$ 0.79	90.77 $\pm$ 0.73	90.66 $\pm$ 0.78	91.18 $\pm$ 0.60
	1D CNN-LSTM	LSTM						
	proposed	2D CNN	parallel	spectrogram + raw input	95.66 $\pm$ 0.63	95.62 $\pm$ 0.65	95.67 $\pm$ 0.63	95.65 $\pm$ 0.63
	2D CNN-LSTM	LSTM						

Table 2.6: Comparison of the implemented single and hybrid models in terms of their subnetworks, merging method, and accuracy performance metric. We provide the average value of accuracy plus/minus one standard deviation for processing the DSA Dataset.

	Model Name	Subnetworks	Merging Method	Input Types	Accuracy $\pm$ std (%)
Single Models	1D CNN	N/A	N/A	raw input	85.75 $\pm$ 0.75
	2D CNN	N/A	N/A	spectrogram	89.77 $\pm$ 0.97
	LSTM	N/A	N/A	raw input	78.72 $\pm$ 1.43
Hybrid Models	standard	1D CNN	consecutive	raw input	84.28 $\pm$ 1.76
	1D CNN-LSTM	LSTM			
	alternative	1D CNN	parallel	raw input + raw input	84.60 $\pm$ 0.94
	1D CNN-LSTM	LSTM			
	proposed	2D CNN	parallel	spectrogram + raw input	92.95 $\pm$ 0.47
	2D CNN-LSTM	LSTM			

individual components. More specifically, the four performance metrics of the alternative 1D CNN-LSTM in Table 2.5 consistently lie in between the individual performance metrics of the single 1D CNN and single LSTM models. This is also the case in Table 2.6, although the accuracy figure is much closer to that of 1D CNN. Since the alternative 1D CNN-LSTM model performance metrics are always lower than those of the 1D CNN, and consistently with larger standard deviation values, between these two models, it would be preferable to use the 1D CNN model which requires less training time.

The performance comparison between the proposed 2D CNN-LSTM and the alternative 1D CNN-LSTM models reveals that using inputs of different nature (raw signals and spectrogram) enables the proposed model to extract features that enhance and complement each other whereas using the same form of input in both branches of the network causes the alternative 1D CNN-LSTM model to extract possibly overlapping or redundant features. Hence, the proposed model succeeds in extracting features with better representation of activities, resulting in improved activity recognition performance.

The results indicate that the use of spectrograms has a particularly positive effect on DL model performance. We note that the single 2D CNN, also using the spectrogram of the raw signals as input, exhibits the second best performance on both datasets. The use of two inherently different subnetwork models (2D CNN and LSTM) that run in parallel, also contributes to the superiority of the proposed model.

2.2.3 Loss versus Epoch, Accuracy versus Epoch Graphs and Confusion Matrices of the Proposed Model

Figs. 2.8–2.10 show the loss versus epoch plot, accuracy versus epoch plot, and the confusion matrix for the proposed 2D CNN-LSTM hybrid model processing the UCI HAR Dataset. The corresponding results for the DSA Dataset are provided in Figs. 2.11–2.13.

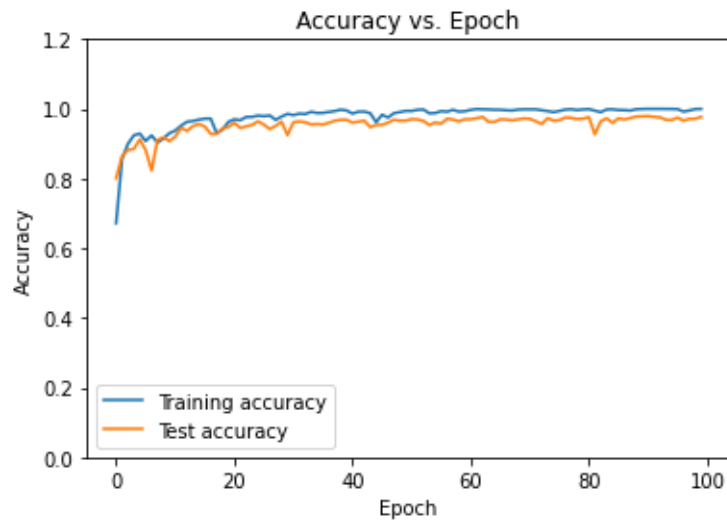


Figure 2.8: Loss versus epoch graph of the proposed model based on the test set of the UCI HAR Dataset.

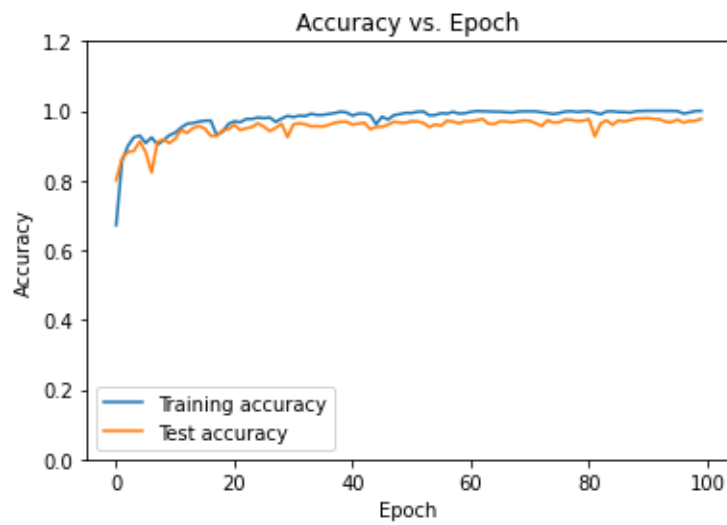


Figure 2.9: Accuracy versus epoch graph of the proposed model based on the test set of the UCI HAR Dataset.

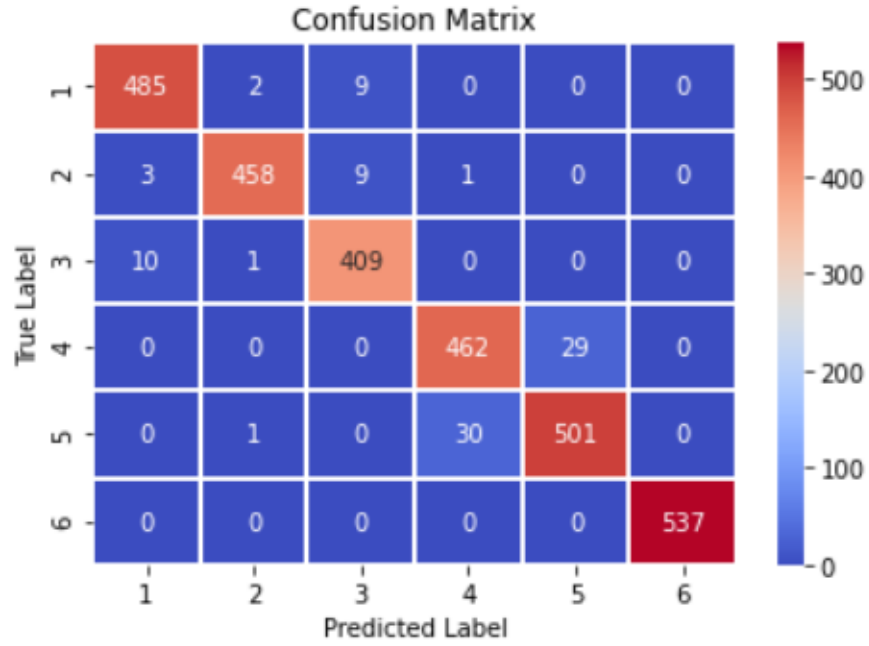


Figure 2.10: Confusion matrix of the proposed model based on the test set of the UCI HAR Dataset.

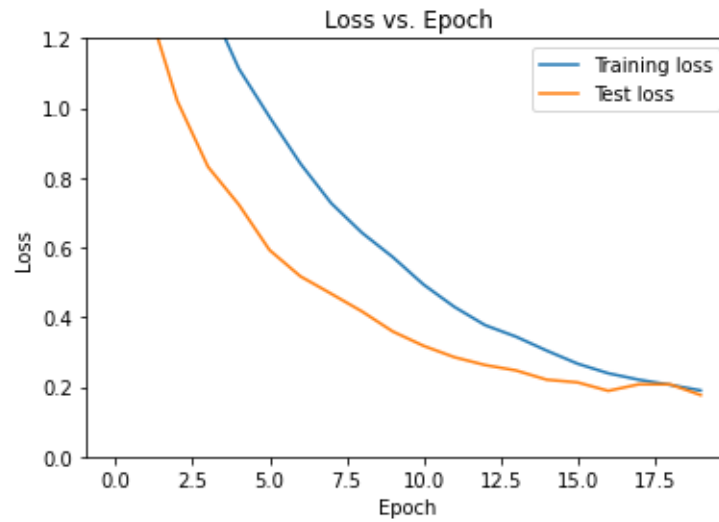


Figure 2.11: Loss versus epoch graph of the proposed model based on testing with the second subject's data of the DSA Dataset.

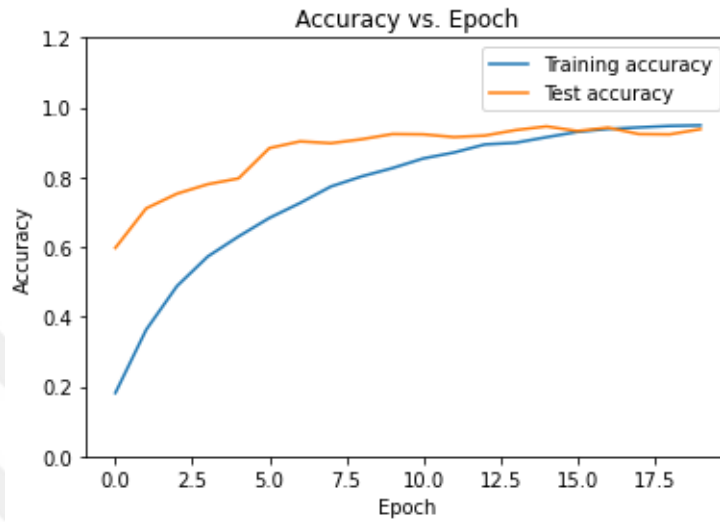


Figure 2.12: Accuracy versus epoch graph of the proposed model based on testing with the second subject's data of the DSA Dataset.

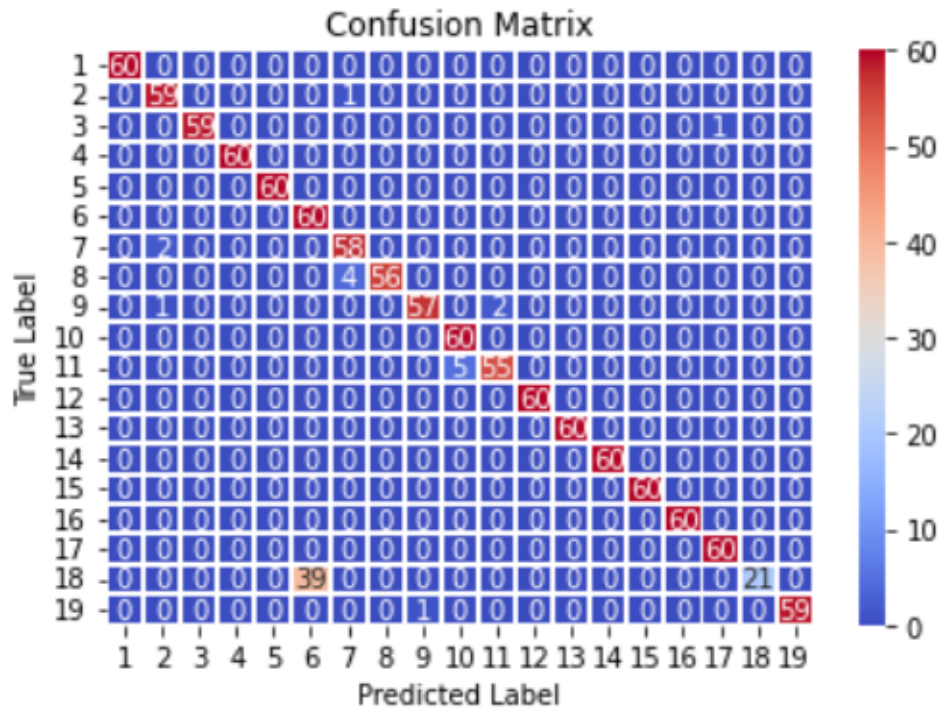


Figure 2.13: Confusion matrix of the proposed model based on testing with the second subject's data of the DSA Dataset.

Fig. 2.8 indicates that the proposed model for the UCI HAR Dataset works properly since the loss decreases gradually during both training and testing. In Fig. 2.9, we observe that there is no overfitting issue since there is a small gap between the training and test accuracies. This is achieved by using dropout layers for regularization.

Fig. 2.11 illustrates that the loss decreases quite smoothly which is an indicator of a good learning rate for the proposed model used in processing the DSA Dataset. On the other hand, this model has a very large dropout probability of 0.76 which means that the model does not use most of the neurons during the training phase while it uses all of them for testing. Therefore, we observe in Fig. 2.12 that the test accuracy is above the training accuracy until the last few epochs. Then, training and test accuracies converge approximately to the same value. This is a good indicator for the generalizability of the model. The large dropout probability enables different neurons to extract features independently and allows the development of a more robust and generalizable model.

The confusion matrix of the UCI HAR Dataset in Fig. 2.10 indicates that the model has similar recognition accuracies for all six activities (Recall that this dataset is somewhat imbalanced with comparable but unequal number of samples from each activity type.). On the other hand, the confusion matrix for the DSA Dataset in Fig. 2.13 shows that the activity 18 (jumping) has a lower recognition rate compared to the other activities. It is incorrectly classified as activity 6 (descending stairs) in 39 out of 60 instances.

2.2.4 Comparison of Computational Cost and Complexity

Given that the typical wearable device has limited computational resources, memory and computational requirements of DL models should be considered carefully for wearable device applications. In Tables 2.7 and 2.8, we display the total number of parameters, model size, and the training times of the six models that we have implemented and compared in this study. Among the models implemented to process the UCI HAR Dataset, the proposed model is the second smallest one

after 2D CNN in terms of the the total number of parameters and the model size. Although it is slower to train compared to single 1D CNN and 2D CNN models, among the three hybrid models, it is the second fastest one, following the standard 1D CNN-LSTM which is the fastest. The size and the training time of the proposed model in processing the DSA Dataset is not as favorable as the proposed model developed for the UCI HAR Dataset. However, it has still similar complexity measures compared to the other five models. When we consider the performance improvement the proposed model brings, this difference in complexity can be tolerated. In case it may not be tolerable for on-device computing in a particular application, model compression or computation off-loading methods can be employed.

Table 2.7: Complexities of the models implemented to process the UCI HAR Dataset.

Model Name	Total Number of Parameters	Model Size (MB)	Training Time (sec)
1D CNN	5,402,340	61.88	353
2D CNN	302,976	3.51	75
LSTM	2,498,327	28.65	2,055
standard 1D CNN-LSTM	1,630,916	18.72	242
alternative 1D CNN-LSTM	3,733,560	42.84	1,390
proposed 2D CNN-LSTM	767,110	8.88	1,236

Table 2.8: Complexities of the models implemented to process the DSA Dataset.

Model Name	Total Number of Parameters	Model Size (MB)	Training Time (sec)
1D CNN	5,404,163	61.73	25
2D CNN	227,293	2.60	14
LSTM	202,929	2.37	79
standard 1D CNN-LSTM	754,419	8.69	28
alternative 1D CNN-LSTM	1,239,742	14.12	88
proposed 2D CNN-LSTM	1,794,638	20.50	118

2.2.5 Comparison of Our Results with Existing Studies

We review the results of five recent studies [11, 12, 16, 31, 32], that use the UCI HAR Dataset (Table 2.9) and four recent works [66, 67, 68, 69] our research group has conducted that process the DSA Dataset (Table 2.10). However, since there are some differences among these studies in terms of the implementation and the performance evaluation details, the results of these studies are not totally comparable with those of the proposed model.

Mutegeki and Han [31] and Deep and Zheng [32] show that the standard 1D CNN-LSTM model improves the performance of the single LSTM model. Our results also confirm this: When the standard 1D CNN-LSTM model is used with the UCI HAR Dataset, there is 0.85–1.18% improvement in the performance metrics whereas with the DSA Dataset, the accuracy improves by 5.56% compared to the LSTM. However, these two studies do not make a performance comparison with the 1D CNN. Ordóñez and Roggen [33] show that the standard 1D CNN-LSTM improves the performance of 1D CNN. Consistent with this, our results in Table 2.5 indicate that there is improvement by 0.14–0.28% in the performance metrics for the UCI HAR Dataset. However, in Table 2.6, we observe that the accuracy metric for the standard 1D CNN-LSTM is lower than that of 1D CNN by 1.47%, with larger standard deviation. This could be due to some differences in the implemented models: The standard 1D CNN-LSTM that we developed for processing the UCI HAR Dataset contains three layers (two layers of 1D CNN followed by a LSTM layer) whereas the 1D CNN comprises four layers. When we make the comparison based on the DSA Dataset, we simply add a LSTM layer after the 1D CNN which is the only difference between the 1D CNN and the 1D CNN-LSTM models for this dataset. Hence, in the UCI HAR Dataset, we compare the standard 1D CNN-LSTM with a deeper 1D CNN with four layers, whereas in the DSA Dataset, we compare the sequential 1D CNN and LSTM layers that form the standard 1D CNN-LSTM with a single 1D CNN layer. On the other hand, Ordóñez and Roggen [33] replace the fully connected layers of the 1D CNN with LSTM layers. In other words, [33] compares the effect of using LSTM or fully connected layers after four 1D CNN layers whereas this paper

compares the effect of using LSTM or 1D CNN layers after two 1D CNN layers in processing the UCI HAR Dataset. For the DSA Dataset, we compare the effect of including or not including an LSTM layer after a 1D CNN layer.

Some other differences in the implementation for the studies that use the UCI HAR Dataset are as follows: Reference [11] uses a window size of 256 while the other studies, including ours, create the segmented data by using a 128-length sliding window. The number of test samples provided in [12] is not the same as in the other studies which indicates that the way the participants' data are split into training and test sets is different than in the other studies. Moreover, in some earlier studies, it is not clear which set is used for hyper-parameter tuning. Some studies may allocate a separate validation set for hyper-parameter tuning while others may not. In this case, the former group would be disadvantaged compared to the latter. Lastly, since DL models are stochastic in nature, we typically get a different result at each repetition of an experiment. For a fair comparison, providing the average and the standard deviation values of the performance metrics would be more appropriate.

In Table 2.10, we compare the accuracy result obtained using the DSA Dataset with the results of our previous studies that use the same dataset but implement ML algorithms instead of DL models. In the table, we have chosen to present the results of the most relevant ML classifier used which is artificial neural networks (ANNs). Among the seven state-of-the-art ML classifiers implemented in [67], ANNs result in the highest accuracy of 90.93%, followed by Support Vector Machines (SVMs) with 90.80%. On the other hand, among the four ML classifiers compared in [66], SVMs are slightly superior to ANNs with respective accuracy figures of 90.36% and 89.67%. In [66], besides the DSA Dataset, the ML algorithms are tested on four other publicly available datasets for which the accuracy varies between 42.35% and 91.03%. In [67], the 19 activities are divided into two broad categories as stationary and non-stationary, for which separate accuracy figures are provided, as well as the accuracy result of 90.93% over all activities for ANN. The works [68] and [69] use the result reported in [67] as reference for comparison with the performances of the position and orientation invariant algorithms developed in those studies. According to this table, using DL models

Table 2.9: Comparison of accuracy performance metrics of the proposed model and the related work for processing the UCI HAR Dataset.

Reference Study	Model	Accuracy (%)
Present Work	2D CNN-LSTM	95.66 $\pm$ 0.63
[11] Yen 2020	1D CNN	95.99
[12] Xia 2020	LSTM-2D CNN	95.78
[16] Tufek 2020	LSTM	93.70
[31] Mutegeki 2020	1D CNN-LSTM	92.13
[32] Deep 2019	1D CNN-LSTM	93.40

Table 2.10: Comparison of accuracy performance metrics of the proposed model and the related work for processing the DSA Dataset.

Reference Study	Model	Accuracy $\pm$ std (%)
Present Work	2D CNN-LSTM	92.95 $\pm$ 0.47
[66] Yurtman 2017	SVM	90.36 $\pm$ 2.28
	ANN	89.67 $\pm$ 3.63
[67] Yurtman 2018	ANN	90.93 $\pm$ 3.95 (all activities)
		87.45 $\pm$ 11.69 (stationary)
		91.86 $\pm$ 3.85 (non-stationary)
[68] Barshan 2020; [69] Yurtman 2021	ANN	90.93 $\pm$ 3.95 (result used from [67])

instead of ML classifiers can improve the classification accuracy by up to 3.28%. We also note that the standard deviation of the accuracy of the proposed model is considerably lower than those of the previous studies presented in this table.

Although there is not much common basis to make a fair comparison between the studies reviewed above in two groups, in any case, we have presented the results of recent studies together with our results in Tables 2.9 and 2.10. Once again, it should be noted that for the reasons explained above, this comparison is not completely like-for-like.

In this chapter, we have proposed a novel 2D CNN-LSTM architecture for the HAR problem. Additionally, we have implemented five commonly used DL models which are three single (1D CNN, 2D CNN, LSTM), two hybrid (standard 1D CNN-LSTM, and an alternative 1D CNN-LSTM) models. We have optimized the hyper-parameters of the six models and compared their activity recognition

performances on the UCI HAR and the DSA Datasets. The proposed architecture has achieved the highest average accuracies of 95.66% and 92.95% on the two datasets, which are, respectively, 2.45% and 3.18% above those of the 2D CNN model that ranks the second while keeping the model complexity comparable with other models.



Chapter 3

Adopting Extracted Features for Different Problem Domains via Transfer Learning

In the previous chapter, we have shown that the proposed 2D CNN-LSTM architecture enables extraction of characteristic features of daily and sports activities which results in accurate activity recognition performance. However, training such DL models has quite high computational requirements. Transfer learning (TL) proposes a solution to this problem. By transferring the sample features from the source domain to the target domain, we bypass the feature extraction part of model training. Therefore, computational requirements are decreased significantly. In this chapter, we show that the extracted features by the proposed model for the HAR problem can provide high quality representations for another human activity-related problem which is user identity recognition. In Section 3.1, we investigate feature transfer from the activity recognition domain to the user identity recognition domain. In Section 3.2, we investigate feature transfer in the reverse direction. As a result, we show that successful feature transfer between the activity recognition and the user identity recognition tasks is possible and such transfer of features has the benefit of reducing the training times drastically.

3.1 Transfer of Features from the Activity Domain to the User Identity Domain

3.1.1 Selecting a Useful Group of Activities for User Identity Recognition

Since individuals perform physical activities in their own characteristic style, wearable motion sensor signals can be used to recognize the user of the wearable device. However, users are not recognizable while performing all kinds of activities. Therefore, a group of activities that is useful for user identity recognition need to be determined. For this purpose, each activity signal in the given datasets is used separately to recognize user identities and the 2D CNN-LSTM architecture, proposed in Chapter 2, is employed on this task. By inspecting the results for the UCI HAR dataset given in Table 3.1, we observe that while the dynamic activities (walking, walking upstairs, and walking downstairs) allow user identity recognition accuracies above 93%, static activities (sitting, standing, and laying) cannot provide satisfactory user identity recognition accuracies. Therefore, only the dynamic activity signals will be used to determine user identities for this dataset. On the other hand, Table 3.2 for the DSA Dataset shows that all the activities in this dataset are useful for high accuracy user identity recognition. The reason for this difference between the two datasets is that while the UCI HAR Dataset has only one sensor unit carried by the users, the DSA Dataset has five different sensor units worn on different body parts which are the chest, both arms, and both legs. Moreover, the DSA Dataset has an additional sensor type which is the magnetometer. Therefore, multiple sensor units and the additional sensor type provide more comprehensive information which enables recognition of users as they perform static and dynamic activities. The activities of moving around in an elevator, rowing, jumping, and playing basketball enable lower accuracy compared to other activities. Therefore, these activities can be excluded for improved user identity recognition. However, we use all the activities in this dataset when we train our model for user identity recognition.

Table 3.1: User identity recognition accuracies for each type of activity in the UCI HAR Dataset.

Activity	Accuracy (%)
walking	98.10 $\pm$ 0.97
walking upstairs	96.51 $\pm$ 0.86
walking downstairs	93.41 $\pm$ 0.92
sitting	35.39 $\pm$ 3.51
standing	46.05 $\pm$ 5.95
laying	54.52 $\pm$ 3.59

Table 3.2: User identity recognition accuracies for each type of activity in the DSA Dataset.

Activity	Accuracy (%)
sitting	99.72 $\pm$ 0.55
standing	100.00 $\pm$ 0.00
lying on back side	100.00 $\pm$ 0.00
lying on right side	100.00 $\pm$ 0.00
ascending stairs	99.44 $\pm$ 0.52
descending stairs	99.86 $\pm$ 0.28
standing in an elevator still	99.17 $\pm$ 0.81
moving around in an elevator	89.58 $\pm$ 1.58
walking in a parking lot	99.44 $\pm$ 0.81
walking on a treadmill with a speed of 4 km/h (in flat position)	100.00 $\pm$ 0.00
walking on a treadmill with a speed of 4 km/h (in 15° inclined position)	100.00 $\pm$ 0.00
running on a treadmill with a speed of 8 km/h	100.00 $\pm$ 0.00
exercising on a stepper	99.58 $\pm$ 0.56
exercising on a cross trainer	100.00 $\pm$ 0.00
cycling on an exercise bike in horizontal position	99.72 $\pm$ 0.34
cycling on an exercise bike in vertical position	99.31 $\pm$ 0.62
rowing	84.72 $\pm$ 1.46
jumping	85.83 $\pm$ 2.18
playing basketball	86.25 $\pm$ 2.76

3.1.2 Selecting the Features to be Transferred

We determined to employ walking, walking upstairs, and walking downstairs activity signals for the UCI HAR Dataset and all the activity signals for the DSA Dataset for the user identity recognition. After determining which activity signals to employ for user identity recognition, it is time to determine which of the extracted features to transfer from the activity domain to the user identity domain. DL models extract features in a hierarchical way which means that the initial layers extract more primitive features and the last few layers extract more complicated and domain-specific features. Therefore, it is usually not feasible to use the features extracted by the last few layers. On the other hand, transferring features from the very first layers may require retraining a larger part of the DL model. In order to determine the optimal layer depth for feature usage, we have conducted experiments by using features of each layer combination. We present the results in Tables 3.3, 3.4, and 3.5.

Columns and rows of Table 3.3 correspond to the 2D CNN and LSTM branches of the proposed 2D CNN-LSTM model for the UCI HAR Dataset, respectively. The lower right corner of the table corresponds to the initial layers of the model and more primitive features while the upper left corner corresponds to the last few layers and more advanced features. For the proposed 2D CNN-LSTM model illustrated in Fig. 3.1, the best accuracy values are obtained by using the features extracted after the dropout 1 and dropout 6 layers. Dropout 1 is the second initial layer of the CNN branch and dropout 6 is the second to last layer of the LSTM branch. This means that the features extracted by the 2D CNN branch of the network from the spectrogram images are getting domain specific after the second layer and one 2D CNN layer is sufficient to produce useful features for different human activity related problem domains. On the other hand, the LSTM branch extracts better and more generalizable features as the network gets deeper. In this case, three LSTM layers have the optimal network depth for time series raw sensor data.

Fig. 3.2 illustrates the feature transfer for the DSA Dataset and Table 3.4

shows the layers of the proposed 2D CNN-LSTM network after the concatenation layer for the DSA Dataset. Concatenated features cannot provide high accuracy levels when they are transferred to the user identity recognition domain. This shows that the extracted features are getting too domain specific after the concatenation layers. Rows and columns of Table 3.5 correspond to the 2D CNN and LSTM branch layers of the model for the DSA Dataset. As in Table 3.3, the lower right corner of the table corresponds to the initial layers of the model and more primitive features and the upper left corner corresponds to the last few layers and more advanced features. Results show that the initial layers of the CNN branch extract more useful features for TL while the LSTM branch extracts more useful features from the last few layers. This is similar as in the case of the proposed model for the UCI HAR Dataset. Figs. 3.1 and 3.2 summarize the TL approach we have adopted from the activity domain to the user identity domain.

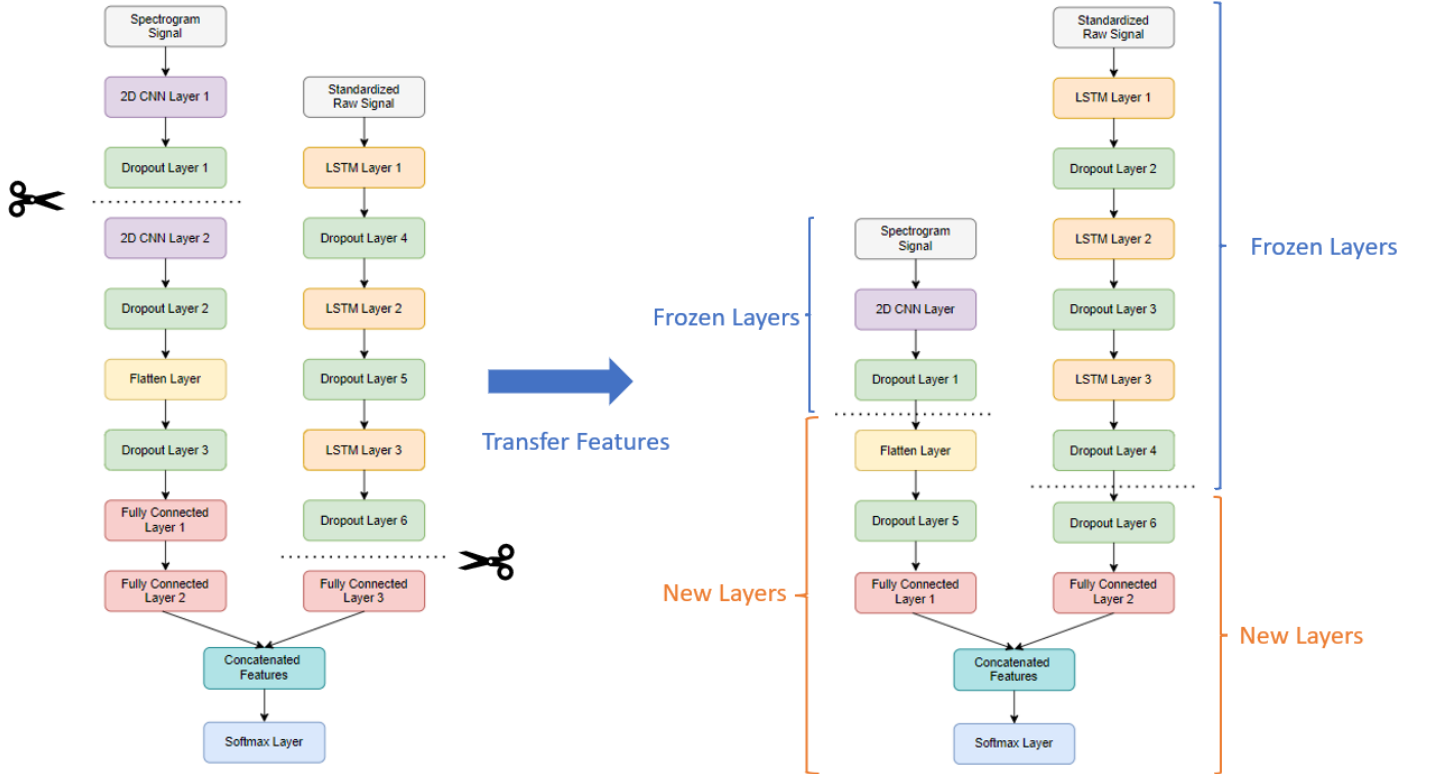


Figure 3.1: Transfer learning from the activity domain to the user identity domain for the UCI HAR Dataset.

We have used the layer weights of the models which already have been trained

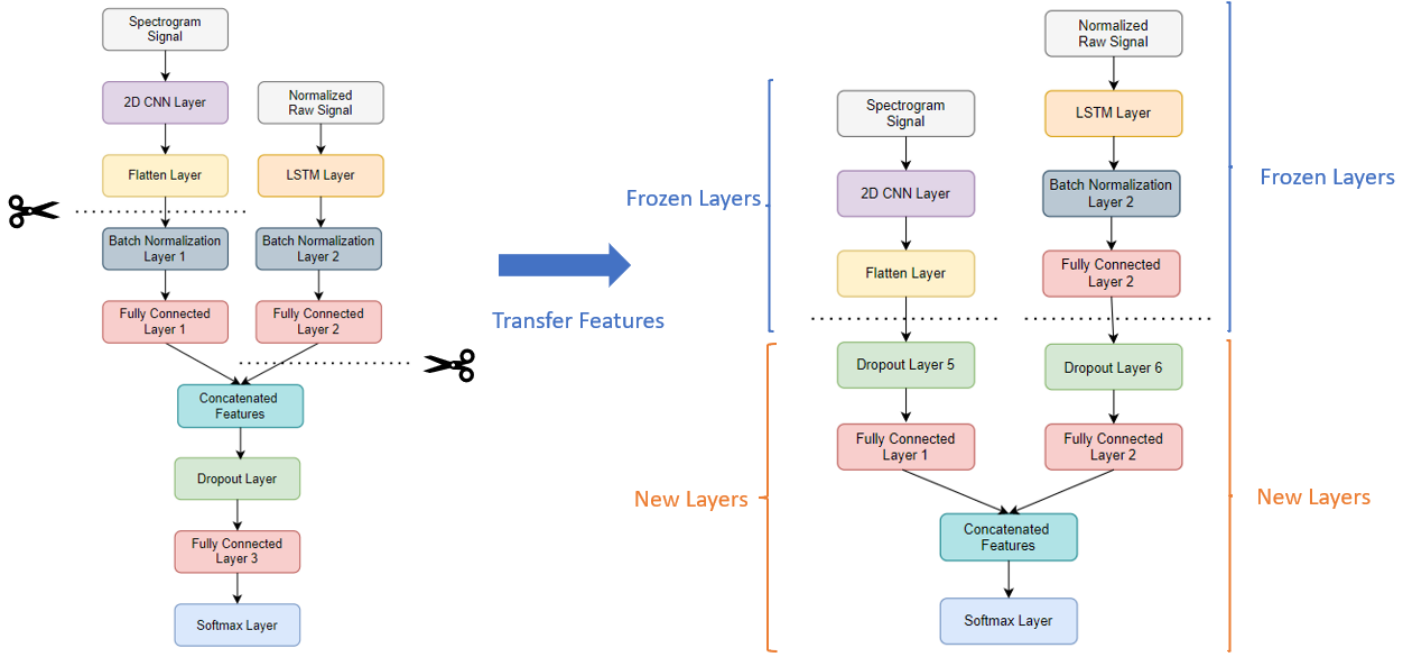


Figure 3.2: Transfer learning from the activity domain to the user identity domain for the DSA Dataset.

Table 3.3: User identity recognition accuracies by features extracted from each layer for the UCI HAR Dataset.

Branch Layers	dense 3	dropout 6	LSTM 3	dropout 5	LSTM 2	dropout 4	LSTM 1
dense 2	17.47±1.06	19.46±0.82	19.07±0.96	51.28±1.12	50.78±0.98	62.52±1.18	63.15±1.17
dense 1	26.33±1.24	25.64±0.91	27.29±1.02	52.44±0.71	52.51±0.66	64.95±1.13	63.11±1.16
dropout 3	88.62±0.62	89.71±0.98	89.24±0.36	89.53±0.91	88.43±1.29	91.23±0.11	91.31±0.87
flatten	88.82±0.92	89.00±1.09	89.54±0.59	89.49±0.66	89.13±1.22	91.11±0.71	90.56±0.39
dropout 2	88.18±0.69	89.14±0.50	89.17±1.59	89.53±0.84	89.37±0.98	90.84±1.74	89.89±1.36
2D CNN 2	87.93±0.86	89.49±1.09	89.42±0.68	89.46±0.73	88.84±0.61	90.19±1.07	89.02±1.03
dropout 1	92.58±0.54	93.31±0.82	93.14±1.14	90.30±1.19	90.20±0.74	91.09±0.39	91.34±1.02
2D CNN 1	93.07±0.34	92.31±0.79	93.11±0.94	90.40±1.12	89.97±0.67	90.49±0.42	91.06±0.51

Table 3.4: User identity recognition accuracies by features extracted from each layer for the DSA Dataset.

Main Layers	dense 3	dropout	concatenation
Accuracy (%)	61.55±1.52	59.14±1.55	59.80±0.65

Table 3.5: User identity recognition accuracies by features extracted from each layer for the DSA Dataset.

Branch Layers	dense 2	batch 2	LSTM
dense 1	57.62 $\pm$ 1.14	59.75 $\pm$ 0.87	56.71 $\pm$ 0.91
batch 1	95.27$\pm$0.26	94.83$\pm$0.19	95.15$\pm$0.56
flatten	95.91$\pm$0.33	95.75$\pm$0.48	95.85$\pm$0.68

on the HAR task as the initial layers and feature extractors of the models for the user identity recognition task. Weights of these layers are frozen and they are not varied during the training part. We have only trained the newly added layers and changed their weight values. Since we have used the previously trained layers as the feature extractors, we have used the newly added layers as classifiers. Therefore, we have chosen the newly added layers in the simplest possible form which are one dense layer for each branch and one softmax layer. We have also used dropout layers as regularizers. The only purpose of the additional flatten layer in Fig. 3.1 is to convert the tensor shape so that it would be compatible with the next layer.

3.1.3 Hyper-parameter Selection

Newly added layers have two hyper-parameters to be selected which are the number of dense layer neurons and the dropout probabilities. Since the number of hyper-parameters is small for the newly added part of the DL network in the user identity recognition domain, it is possible to select the hyper-parameter values by experimenting with different values one at a time. Tables 3.6 and 3.7 show the accuracy results for different number of neurons on the UCI HAR and the DSA Datasets, respectively. Dropout probabilities are fixed as 0.5 while conducting these experiments. Neuron numbers 128, 256, and 512 give quite close accuracy values for the UCI HAR Dataset. By considering the complexity issues and the small gap between the accuracy values, 128 is chosen as the optimal neuron number. Values equal to or above 128 give the best accuracy results for the DSA Dataset and there is not much difference between the accuracy values for

different neuron numbers. Therefore, the smallest neuron number 128 is chosen to reduce complexity. Tables 3.8 and 3.9 show that suitable dropout probability values for the UCI HAR and the DSA Datasets are 0.5 and 0.7, respectively. While conducting these experiments, neuron numbers are 128 for both datasets.

Table 3.6: User identity recognition accuracies as the number of neurons changes for the UCI HAR Dataset.

Dense Neuron Number	Accuracy (%)
32	90.77 $\pm$ 0.84
64	93.67 $\pm$ 0.66
128	93.82 $\pm$ 0.70
256	93.81 $\pm$ 0.52
512	94.00 $\pm$ 0.59
1024	93.29 $\pm$ 0.85
2048	92.78 $\pm$ 0.71
4096	91.67 $\pm$ 1.14

Table 3.7: User identity recognition accuracies as the number of neurons changes for the DSA Dataset.

Dense Neuron Number	Accuracy (%)
8	86.03 $\pm$ 0.87
16	93.38 $\pm$ 0.37
32	95.22 $\pm$ 0.34
64	95.85 $\pm$ 0.25
128	96.40 $\pm$ 0.10
256	96.38 $\pm$ 0.11
512	96.70 $\pm$ 0.36
1024	96.55 $\pm$ 0.28
2048	96.40 $\pm$ 0.75
4096	96.72 $\pm$ 0.35
8192	96.52 $\pm$ 0.35

Table 3.8: User identity recognition accuracies as the dropout probability changes for the UCI HAR Dataset.

Dropout Probability	0.1	0.3	0.5	0.7	0.9
Accuracy (%)	93.55 $\pm$ 0.41	93.57 $\pm$ 0.46	93.71 $\pm$ 0.84	93.25 $\pm$ 1.05	91.04 $\pm$ 0.68

Table 3.9: User identity recognition accuracies as the dropout probability changes for the DSA Dataset.

Dropout Probability	0.1	0.3	0.5	0.7	0.9
Accuracy (%)	95.91 $\pm$ 0.43	96.05 $\pm$ 0.49	96.08 $\pm$ 0.35	96.26 $\pm$ 0.34	94.23 $\pm$ 0.39

3.1.4 Comparison of the Full and Transfer Models

We have determined the useful features to be transferred to the user identity domain and selected the hyper-parameter values for the newly added part of the DNN model. Hereafter, the performance of the model by using the transferred features can be evaluated. Tables 3.10 and 3.11 show the comparison between the full network and the transfer network for the UCI HAR and the DSA Datasets, respectively. There is quite a small performance gap between the full and transfer networks. While the accuracy is smaller than that of the full model for the transfer network on the UCI HAR Dataset, it is greater for the DSA Dataset. There are two possible reasons. Firstly, five different sensor units were used while acquiring the DSA Dataset. Therefore, knowledge from the different body parts are transferred to the user identity recognition domain which enables better recognition for user identities. Secondly, there is an additional sensor type in the DSA Dataset which is the magnetometer. Again, this sensor type transfers additional information from the activity domain to the user identity domain which results in better feature transfer performance compared to the UCI HAR Dataset. In addition to the high user identity recognition accuracies, transfer models reduce the training times considerably. For the UCI HAR Dataset, the training time for the transfer network is approximately 17 times less than the full network while for the DSA Dataset it is nine times less.

Table 3.10: Full versus transferred network from the activity domain to the user identity domain for the UCI HAR Dataset.

	Full	Transfer
Accuracy (%)	96.83 $\pm$ 0.31	94.38 $\pm$ 0.09
Training Time	547.35 (s)	32.16
Parameter Number	1,367,966	1,277,726

Table 3.11: Full versus transferred network from the activity domain to the user identity domain for the DSA Dataset.

	Full	Transfer
Accuracy (%)	94.59 $\pm$ 0.16	96.15 $\pm$ 0.62
Training Time	496.43 (s)	52.91
Parameter Number	1,533,764	950,408

The main reason for the reduced training time is obviously eliminating some layers of the network. While their parameter numbers are not so high, their impact on the training time is significant. In order to provide a more quantitative result, we approximately measured the time spent for training each layer of the given networks. For this purpose, each layer is excluded one at a time and the training time of the rest of the network is measured. The difference between the measured value and the training time of the full network is considered as the time spent on that layer during training. While this method does not provide the exact time spent for a particular layer, it can still give an approximate measure. Tables 3.12 and 3.13 show that the bulk of the training time is spent on the LSTM layers. The first reason is that LSTM cells require many more matrix operations in order to compute the cell outputs. Secondly, LSTM cells operate recurrently which multiplies the amount of computation time as the number of time steps in the input signals. Therefore, we can conclude that the time saving by the transfer networks is mainly achieved by eliminating the training phase of the LSTM layers.

Table 3.12: Training times of each layer in processing the UCI HAR Dataset.

Layers	Training Time (s)
2D CNN	25.60
dense 1	9.73
LSTM 1	197.98
LSTM 2	195.96
LSTM 3	198.10
dense 2	3.36

Table 3.13: Training times of each layer in processing the DSA Dataset.

Layers	Training Time (s)
LSTM	436.72
dense 2	10.47
dense 3	34.87
2D CNN	65.46
dense 1	42.29

Figs. 3.3 and 3.5 display the accuracy versus epoch number plots during the training process of the full and the transfer networks for the two datasets. Both figures illustrate that the transfer models converge to maximum accuracy values in earlier epochs compared to the full models. This is because there is a smaller number of layers and parameters to be trained in the transfer networks. Therefore, Adam optimizer can reach to the optimal loss point after a smaller number of epochs. Figs. 3.4 and 3.6 show the accuracy versus time plots for the two datasets. As the most time-consuming layers of the full models are eliminated, the training time spent on each epoch is reduced considerably. Therefore, these figures illustrate that the transfer models reach to the best accuracy level much faster than the full networks.

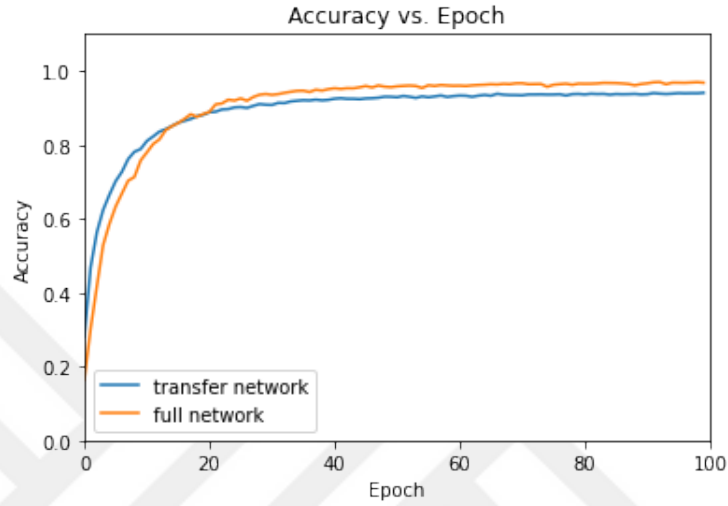


Figure 3.3: User identity domain convergence with respect to epoch numbers for the UCI HAR Dataset.

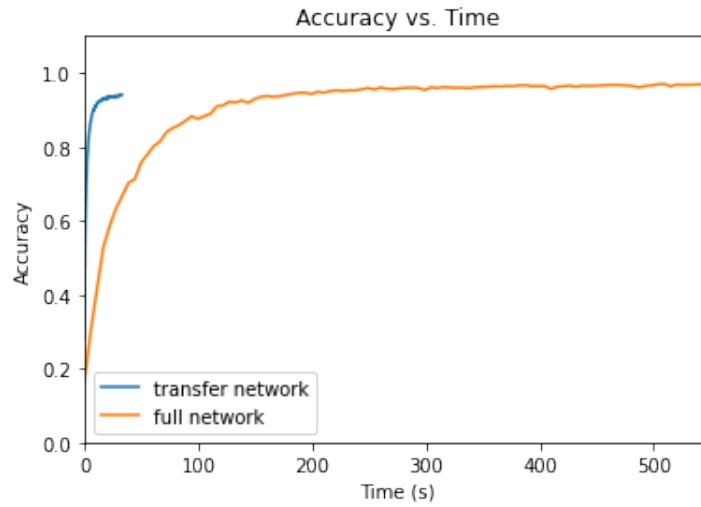


Figure 3.4: User identity domain convergence with respect to time for the UCI HAR Dataset.

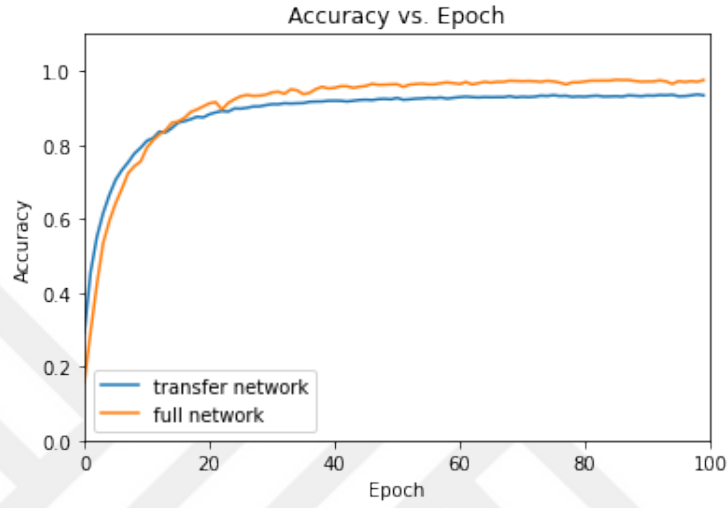


Figure 3.5: User identity domain convergence with respect to epoch numbers for the DSA Dataset.

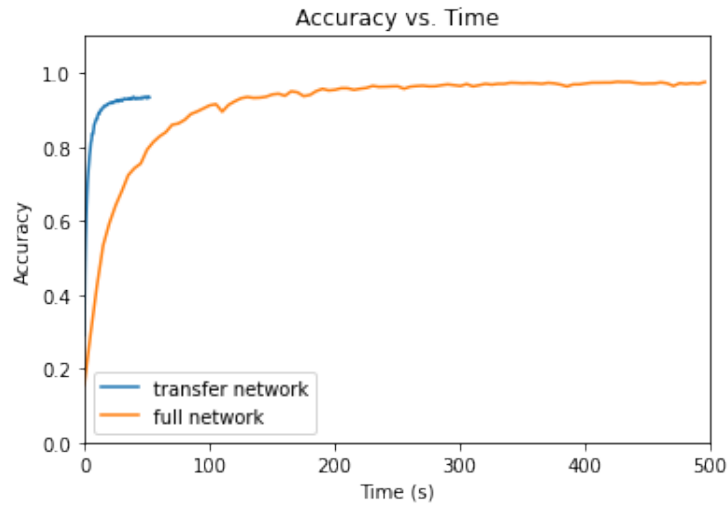


Figure 3.6: User identity domain convergence with respect to time for the DSA Dataset.

3.1.5 Binary Classifier for Useful Activity Selection

We have achieved recognition of device users accurately with full networks and transfer networks. Transfer networks have enabled several times smaller training times. However, we have assumed that the signals are acquired while the users perform certain types of activities which allow accurate user identity recognition. In real life, users do not always perform specified activities. Therefore, real-life application of user identity recognition algorithms should be able to classify the sensor signals as useful or not useful for user identity recognition. For the UCI HAR Dataset, the dynamic activities of walking, walking upstairs, and walking downstairs are labeled as useful and other activities are labeled as not useful for user identity recognition. For the DSA Dataset moving around in an elevator, rowing, jumping, and playing basketball are labeled as not useful and remaining activities are labeled as useful. Then, we train a binary classifier model with these data labels. If the sensor signal is classified as useful for user identity recognition, then the proposed identity recognition module is activated. This flow is illustrated in Fig. 3.7.

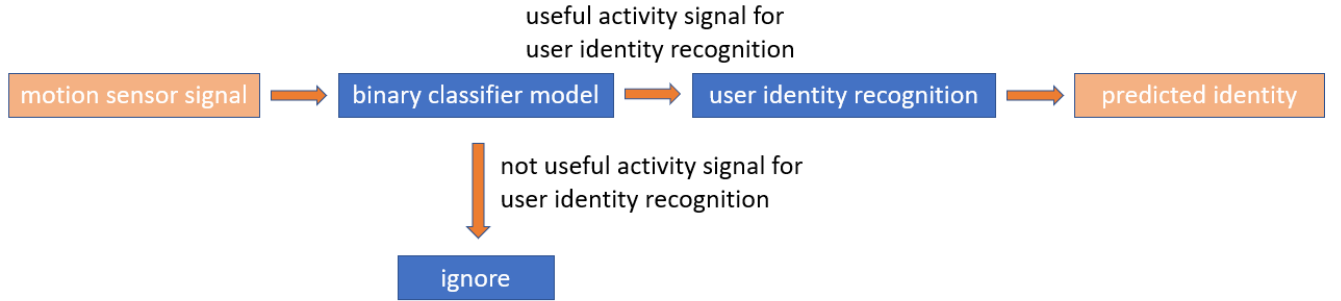


Figure 3.7: User identity recognition scheme with wearable motion sensor data.

The proposed binary classifier uses the transferred features which have been extracted by the frozen part of the models in Figs. 3.1 and 3.2. Newly added layers of the binary classifier is a simple model which has one dense layer for the features coming from the 2D CNN branch and another dense layer for the features coming from the LSTM branch. Concatenated features are classified by a softmax layer. Each dense layer has eight neurons and the models have 79,410 and 59,306 parameters, respectively, for the UCI HAR and the DSA Datasets.

The network architecture is exactly the same for both datasets. Binary classifiers achieve 100.00% and 98.92% accuracies for the UCI HAR and the DSA Datasets, respectively, as displayed in Tables 3.14 and 3.15.

Table 3.14: Binary classifier results for the UCI HAR Dataset.

Accuracy (%)	100.00 $\pm$ 0.00
Training Time (s)	12
Parameter Number	79,410

Table 3.15: Binary classifier results for the DSA Dataset.

Accuracy (%)	98.92 $\pm$ 0.15
Training Time (s)	10.30
Parameter Number	59,306

3.1.6 Comparison with Recent Studies

There is a limited number of studies for the user identity recognition problem by using wearable motion sensor data. Sakorn et al. [72] uses the UCI HAR Dataset which allow us to make a comparison between the results of that study and ours. The most successful model in their study is the cascaded 2D CNN-LSTM architecture. This model achieves 92.43% accuracy by using dynamic activities. Our proposed model surpasses this score by 96.83% and 94.38% accuracies using the full and transfer networks, respectively. Table 3.16 also shows the performance of the user identity recognition models by using signals from single activities. The proposed model in our study achieves much greater performance for the dynamic activities of walking, walking upstairs, and walking downstairs when both their signals are used separately and collectively. These are the activities that enable user identity recognition at a reasonable level. Signals of the remaining activities are not as useful for user identity recognition at a reasonable level and those signals will be just ignored as depicted in Fig. 3.7. Therefore, user identity recognition performance with those signals is insignificant.

Table 3.16: Results in the literature for the UCI HAR Dataset.

Activities	Sakorn et al. [72]	Proposed Full Model	Proposed Transfer Model
walking	94.57	98.84 $\pm$ 0.32	98.49 $\pm$ 0.50
walking upstairs	92.98	99.05 $\pm$ 0.22	95.52 $\pm$ 0.42
walking downstairs	89.78	95.59 $\pm$ 0.82	91.28 $\pm$ 0.63
sitting	55.74	48.28 $\pm$ 2.53	30.71 $\pm$ 1.74
standing	62.18	51.26 $\pm$ 3.32	41.68 $\pm$ 1.30
laying	70.44	73.84 $\pm$ 3.26	63.94 $\pm$ 1.86
walking related activities together	92.43	96.83 $\pm$ 0.31	94.38 $\pm$ 0.09

3.2 Transfer Features from the User Identity Domain to the Activity Domain

In the previous section, we have shown that the features obtained for the activity recognition domain are applicable in the user identity recognition domain. Therefore, user identity recognition models can directly use activity recognition models as feature extractors in the user identity recognition domain without any further training. In this subsection, we investigate to see if the features extracted by the user identity recognition models are applicable in the activity recognition domain.

3.2.1 Selecting the Features to be Transferred

As the first step, the 2D CNN-LSTM models illustrated on the left sides of Figs. 3.8 and 3.9 are trained for the user identity recognition problem. For both directions of TL, these models are used as the base models to determine which features to transfer. Table 3.17 shows the accuracy results for the features extracted from each layer for the UCI HAR Dataset. As in Table 3.3, rows and columns show the CNN and LSTM branches, respectively, and layers get deeper towards the top left corner of the table. Most accurate activity recognition is achieved when the features extracted after dropout 1 and dropout 4 layers are used which are the second-most shallow layers. This is different from the case

for feature transfer in the opposite direction. More complicated features which have been extracted by the deeper layers of the LSTM branch were more useful to transfer from the activity recognition domain to the user identity recognition domain. This is an indicator that the user identity recognition features are more domain specific than the activity recognition features. Tables 3.18 and 3.19 display the accuracy results in the activity domain by transferring the features from the user identity domain for the DSA Dataset. Most accurate results are obtained by using the features extracted by the LSTM and flatten layers which are the most shallow ones. These results also verify that the features extracted in the user identity domain are more domain specific while the features extracted in the activity domain are more generalizable. Figs. 3.8 and 3.9 illustrate the layer transfer from the user identity domain to the activity domain. Weights of the transferred layers are frozen as in the user identity domain and the weights of the rest of the network are trained from scratch for the activity recognition problem.

Table 3.17: Activity recognition accuracies by features extracted from each layer for the UCI HAR Dataset.

Branch Layers	dense 3	dropout 6	LSTM 3	dropout 5	LSTM 2	dropout 4	LSTM 1
dense 2	55.28±2.18	56.99±1.66	56.47±0.84	77.73±0.90	77.10±1.07	84.19±0.53	83.76±0.40
dense 1	67.47±1.69	67.76±0.74	67.06±1.22	82.45±0.89	83.28±0.53	88.57±0.64	88.60±0.58
dropout 3	79.65±0.69	78.70±0.51	79.77±1.30	88.79±0.93	89.15±0.66	91.27±0.66	91.86±0.80
flatten	78.55±0.82	78.18±0.95	79.35±1.10	88.13±0.50	88.38±0.94	91.72±0.60	91.39±0.60
dropout 2	79.47±1.14	78.84±0.39	78.22±0.61	88.83±1.16	88.48±0.41	91.70±0.55	91.66±0.49
2D CNN 2	79.75±0.61	79.28±0.86	79.20±0.97	88.52±0.68	88.62±0.50	91.57±0.97	91.42±0.75
dropout 1	87.73±1.03	88.57±0.67	87.60±0.75	90.46±0.75	91.15±0.33	92.39±0.32	92.37±0.44
2D CNN 1	88.21±0.70	87.77±0.82	87.82±0.58	90.69±0.41	90.74±0.67	92.29±0.71	92.33±0.30

Table 3.18: Activity recognition accuracies by features extracted from each layer for the DSA Dataset.

Main Layers	dense 3	dropout	concatenation
Accuracy (%)	28.29±0.65	51.59±0.65	44.77±0.35

Table 3.19: Activity recognition accuracies by features extracted from each layer for the DSA Dataset.

Branch Layers	dense 2	batch 2	LSTM
dense 1	45.53 $\pm$ 1.37	47.44 $\pm$ 0.77	47.58 $\pm$ 0.40
batch 1	88.30$\pm$0.54	88.62$\pm$0.30	88.40$\pm$0.48
flatten	88.40$\pm$0.21	88.24$\pm$0.20	88.67$\pm$0.34

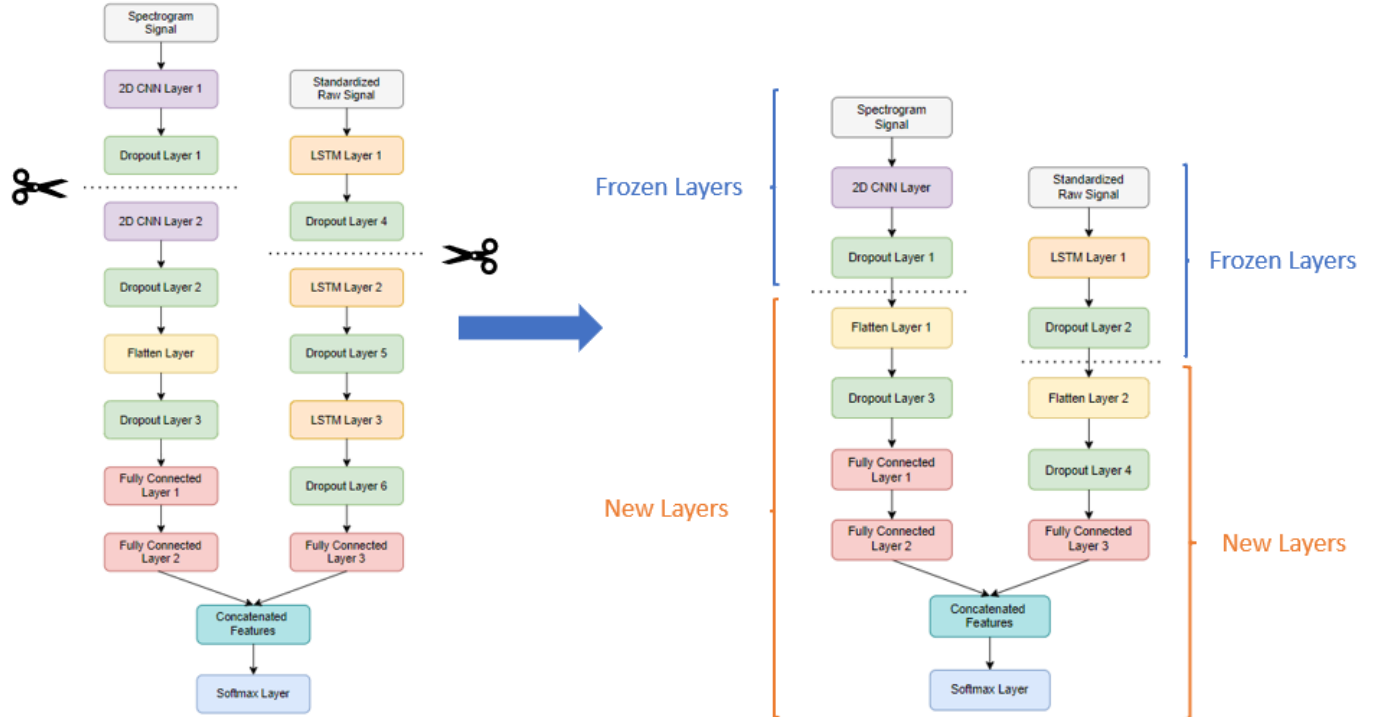


Figure 3.8: Transfer learning from the user identity domain to the activity domain for the UCI HAR Dataset.

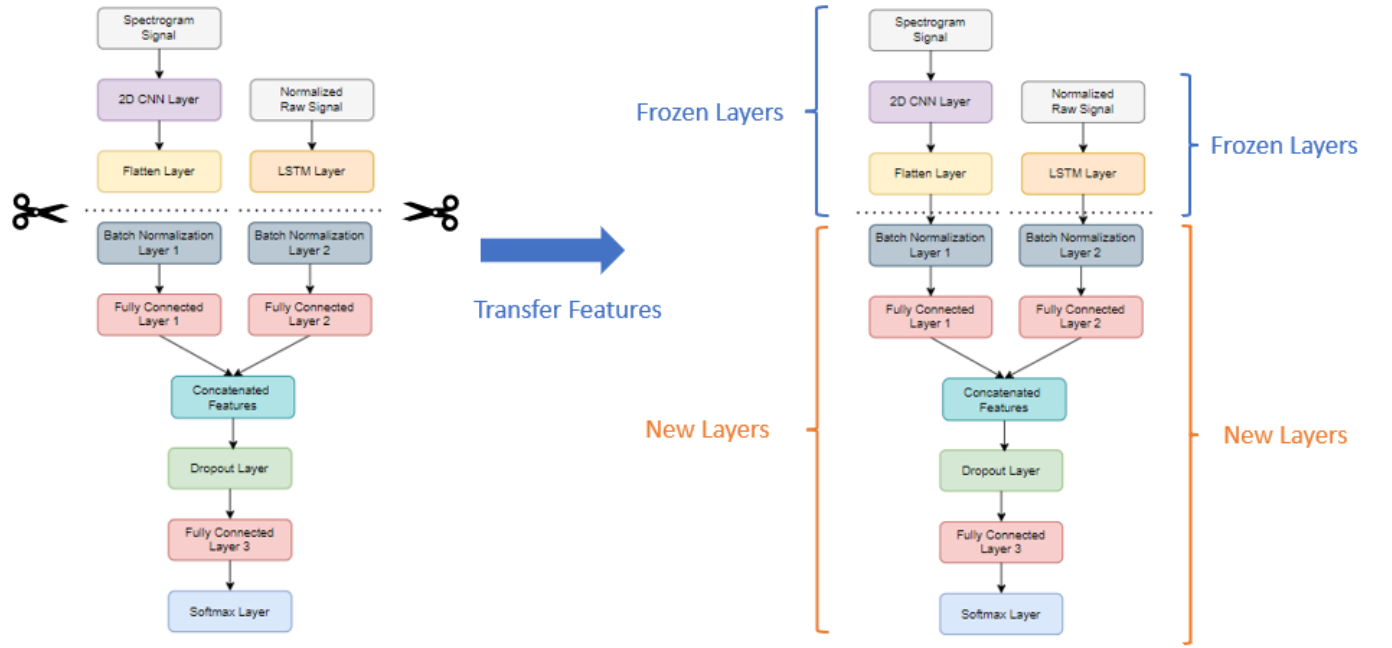


Figure 3.9: Transfer learning from the user identity domain to the activity domain for the DSA Dataset.

3.2.2 Hyper-parameter Selection

Hyper-parameters of the newly added layers have been selected by experimenting with several values one at a time. Tables 3.20 and 3.21 show the results for the number of neurons. While conducting these experiments dropout probability is set to 0.5 for both datasets. We have observed that the number of neurons does not have a high impact on the accuracy values. This is an indicator that the capacity of the models is large enough to learn data features and increasing the number of neurons does not provide much advantage for their learning capacities. Number of neurons is chosen as 64 and 128 for the UCI HAR and the DSA Datasets, respectively. Although the neuron number 128 does not provide the highest accuracy, it has been chosen since the gap between the highest accuracy is quite small and it provides much lower model complexity. Dropout probability has been selected as 0.5 for both datasets since this value provides the highest accuracy value as in Tables 3.22 and 3.23. During these experiments number of neurons is 64 and 128 for the UCI HAR and the DSA Datasets, respectively

Table 3.20: Activity recognition accuracy as the number of neurons is varied for the UCI HAR Dataset.

Neuron Numbers	16	32	64	128	256
Accuracy (%)	92.00 $\pm$ 0.33	92.37 $\pm$ 0.41	92.61 $\pm$ 0.35	92.55 $\pm$ 0.32	92.16 $\pm$ 0.21

Table 3.21: Activity recognition accuracy as number of neurons is varied for the DSA Dataset.

Neuron Numbers	32	64	128	256	512	1024
Accuracy (%)	82.10 $\pm$ 1.12	87.12 $\pm$ 0.33	87.75 $\pm$ 0.60	87.96 $\pm$ 0.13	87.48 $\pm$ 0.31	87.39 $\pm$ 0.25

Table 3.22: Activity recognition accuracy as dropout probability is varied for the UCI HAR Dataset.

Dropout Probability	0.1	0.3	0.5	0.7	0.9
Accuracy (%)	92.48 $\pm$ 0.21	92.45 $\pm$ 0.31	92.53 $\pm$ 0.45	92.48 $\pm$ 0.53	92.00 $\pm$ 0.35

Table 3.23: Activity recognition accuracies as dropout probability is varied for the DSA Dataset.

Dropout Probability	0.1	0.3	0.5	0.7	0.9
Accuracy (%)	87.54 $\pm$ 0.53	87.72 $\pm$ 0.57	88.00 $\pm$ 0.26	87.78 $\pm$ 0.53	86.66 $\pm$ 0.37

3.2.3 Comparison of the Full and Transfer Models

Tables 3.24 and 3.25 show the performance comparison between the full and the transfer networks. Accuracy values of the transfer networks are below the full networks by a small margin. On the other hand, training times of the transfer networks are approximately 15 times and nine times less than the full networks for the UCI HAR and the DSA Datasets, respectively.

Table 3.24: Full versus transfer network from the user identity domain to the activity domain for the UCI HAR Dataset.

	Full	Transfer
Accuracy (%)	95.66 $\pm$ 0.63	93.67 $\pm$ 0.35
Training Time (s)	1,236	78.56
Parameter Number	767,110	1,160,134

Table 3.25: Full versus transferred network from the user identity domain to the activity domain for the DSA Dataset.

	Full	Transfer
Accuracy (%)	92.95 $\pm$ 0.47	87.57 $\pm$ 0.50
Training Time (s)	118	13.52
Parameter Number	1,794,638	1,026,527

Figs. 3.10 and 3.12 display the accuracy versus epoch number plots during the training process of the full and transfer networks for the two datasets. Transfer networks enable higher accuracies in the initial epochs and faster convergence to maximum accuracy values. This is achieved by eliminating the training process of the initial layers of the networks. Figs. 3.11 and 3.13 show the accuracy versus time plots for the same process for the two datasets. The difference between the full and transfer networks is more obvious in these plots. Transfer networks reduce the training times considerably by eliminating the initial layers of the networks which are responsible for most of the computational cost. By considering the major reduction in the training time and the acceptable gap between the per-

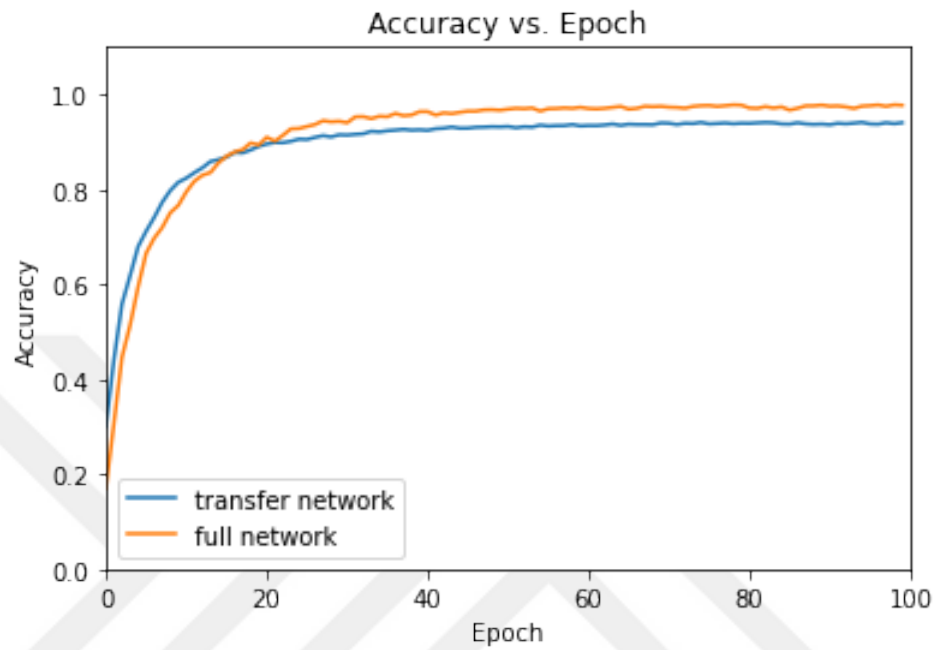


Figure 3.10: Activity domain convergence with respect to epoch numbers in processing the UCI HAR Dataset.

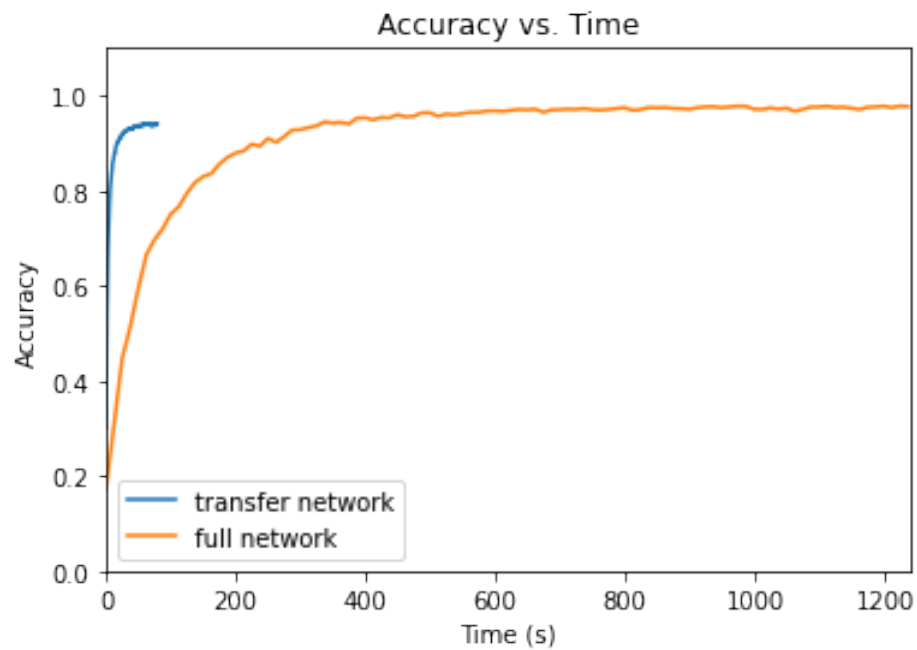


Figure 3.11: Activity domain convergence with respect to time in processing the UCI HAR Dataset.

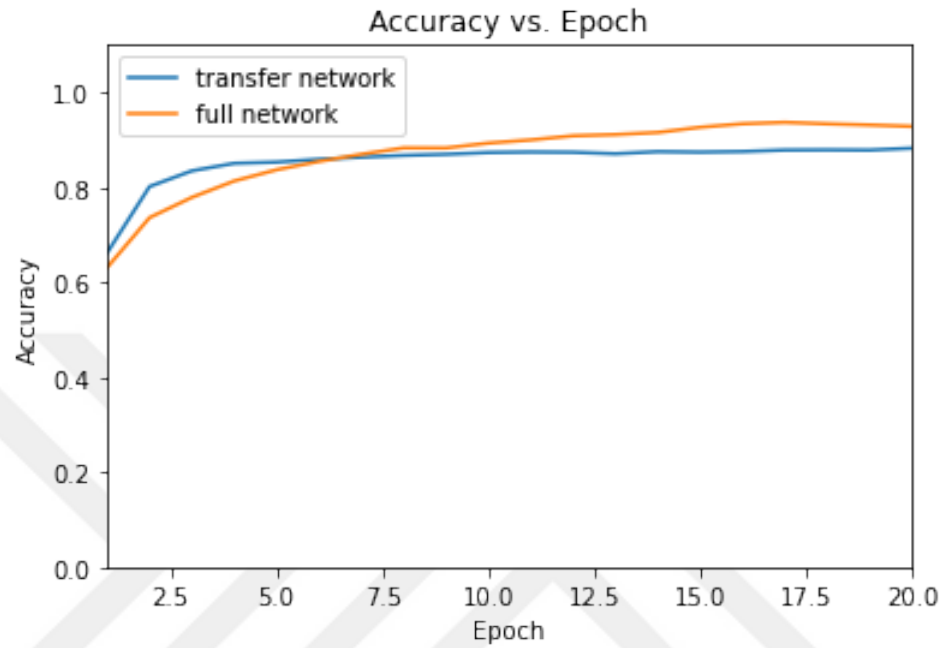


Figure 3.12: Activity domain convergence with respect to epoch numbers in processing the DSA Dataset.

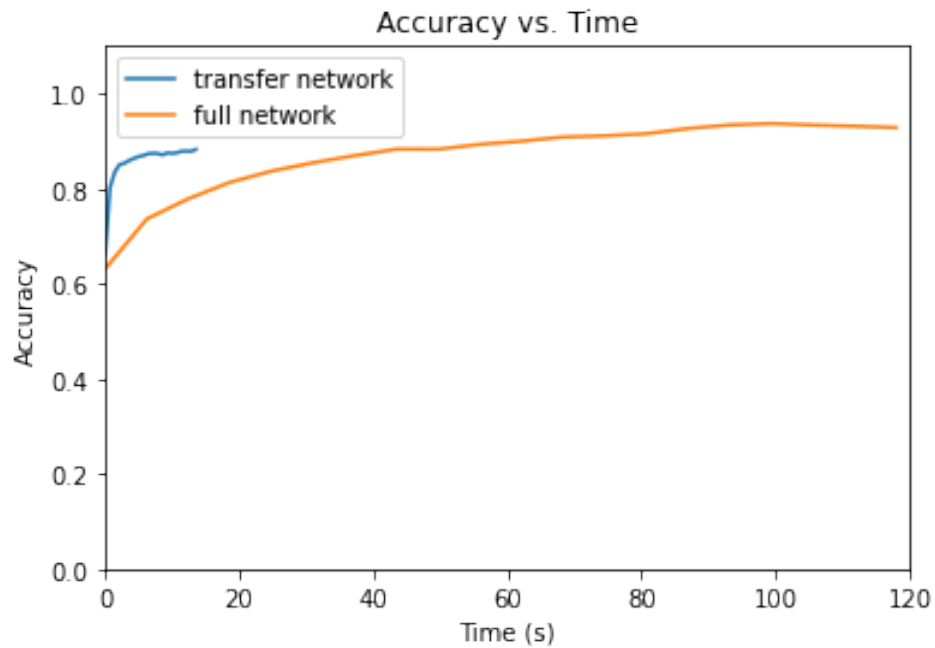


Figure 3.13: Activity domain convergence with respect to time in processing the DSA Dataset.

formances between the full and transfer networks, it can be concluded that the extracted features in the user identity domain can be used in the activity recognition domain with acceptable performance when the computational resources are limited. Additionally, by considering that the performance gaps between the full and transfer networks are smaller after the feature transfer from the activity recognition domain to the user identity domain, it can be concluded that the feature transfer in this direction is more effective and the activity recognition domain enables more generalizable feature extraction than the user identity domain.

In this chapter, we have demonstrated that successful feature transfer between the activity recognition and the user identity recognition tasks is feasible in both directions and has the benefit of reducing the training times of the DL models drastically. Also, we have implemented an end-to-end system to recognize user identities by adding a binary classifier module before the user identity recognition module to detect the useful activity signals for user identity recognition.

Chapter 4

Reducing the Model Sizes of the Proposed Models for Memory Limited Devices

The methods proposed in Chapters 2 and 3 provide benefits while training a DL model for the HAR problem. The training part of a DL model is usually done in an offline setup which has large memory size and high computational power. However, trained DL models are usually deployed on embedded devices which typically have limited memory space. Therefore, it is important to keep the size of the DL models minimal. We apply the proposed DL architecture on two different tasks which are human activity recognition and user identity recognition. Additionally, we employ a binary classifier as the initial block of the user identity recognition task to classify the input signal as useful or not for user identity recognition. We train these models on the two datasets (UCI HAR and DSA Datasets). Therefore, we use six models (3 tasks x 2 datasets) in total as illustrated in Figs. 4.1, 4.2, and 4.3 for the above mentioned tasks and datasets. In this chapter, we show how to decrease the sizes of these six DL models by 17 to 116 times.

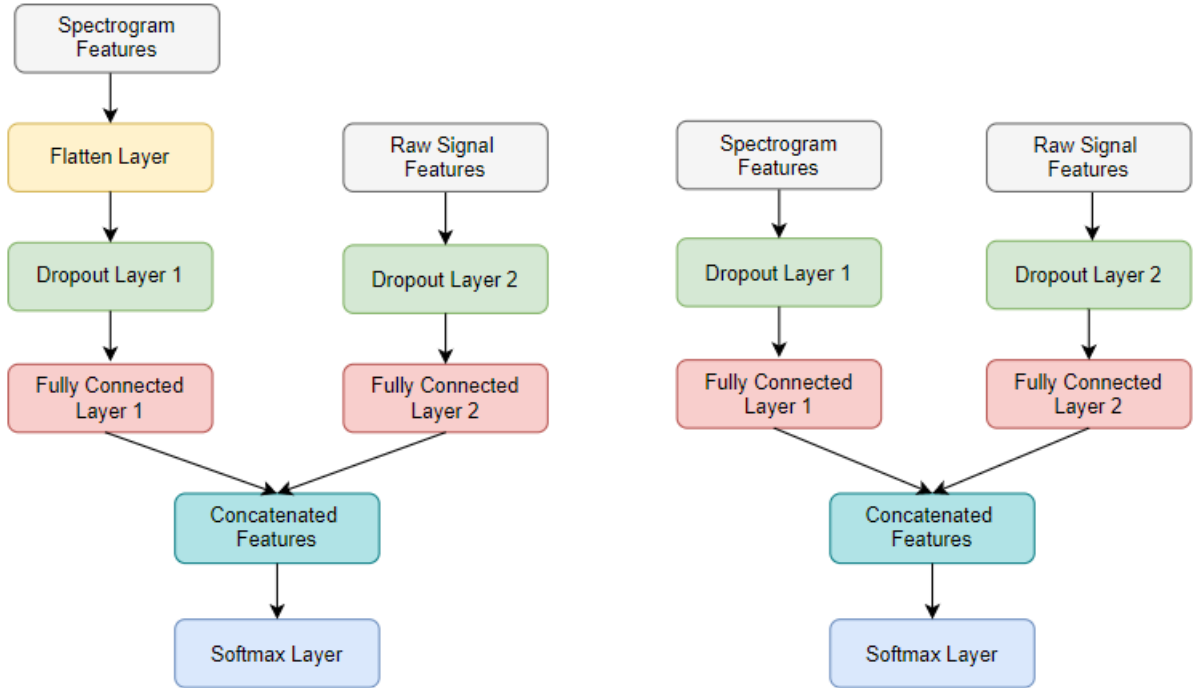


Figure 4.1: Transfer models for identity recognition on UCI HAR (left) and DSA (right) Datasets.

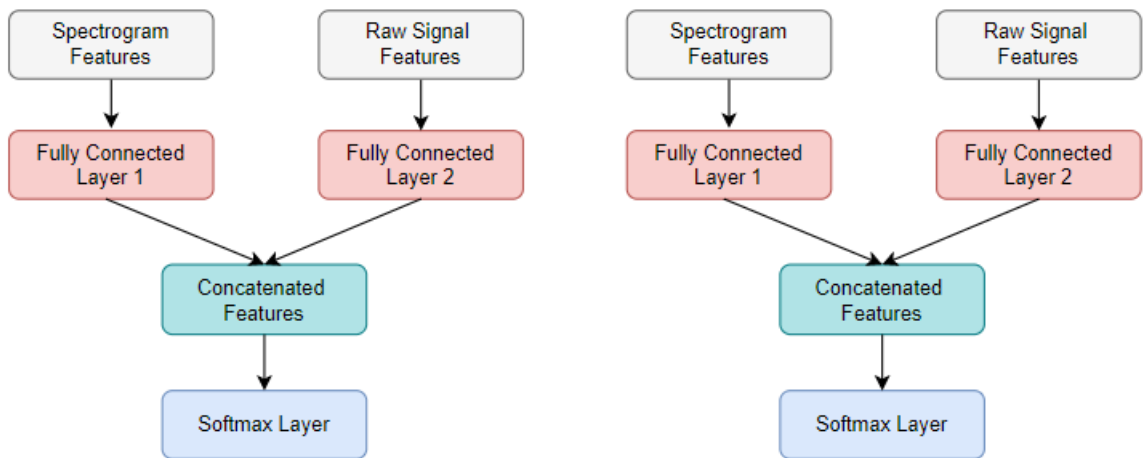


Figure 4.2: Binary models for useful activity detection on UCI HAR (left) and DSA (right) Datasets.

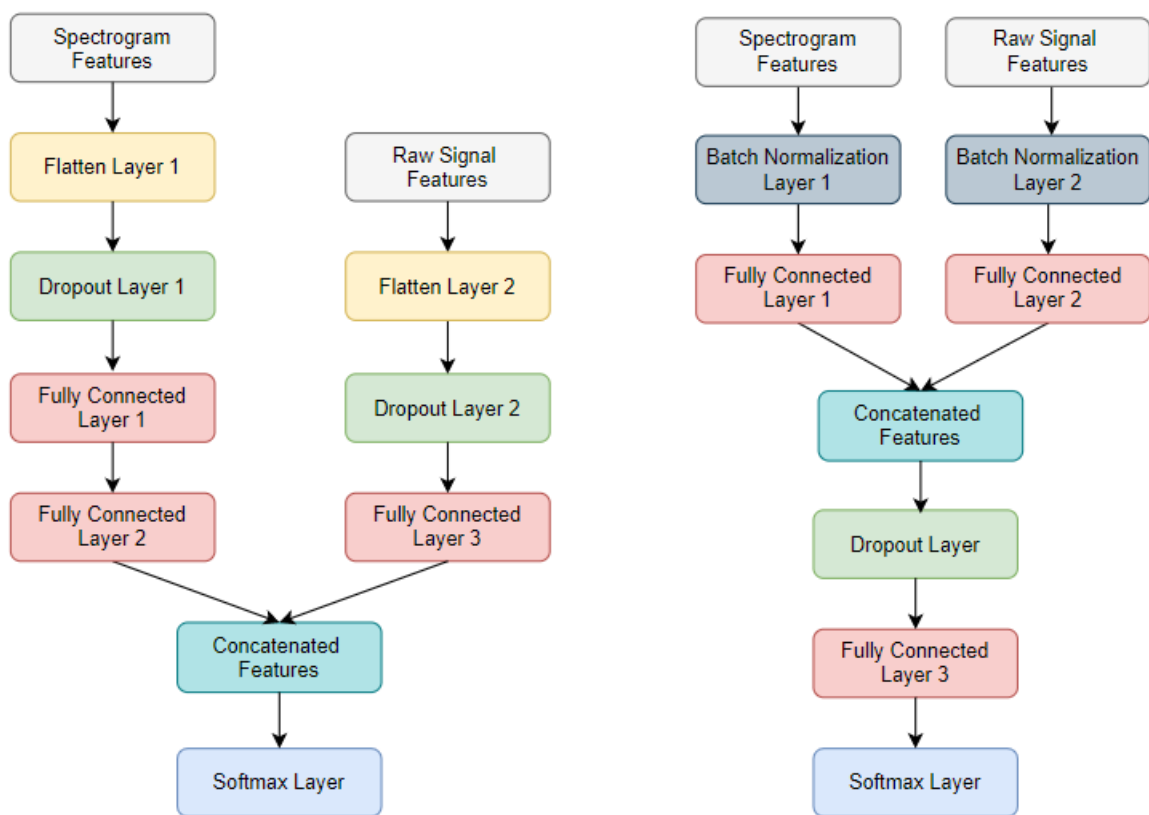


Figure 4.3: Transfer models for activity recognition on UCI HAR (left) and DSA (right) Datasets.

4.1 Investigating the Factors Increasing the Size of the Proposed Models

Initially, we need to investigate the factors that increase the size of the proposed models. Since a DL model is simply a set of parameters, the size of a DL model is proportional to the number of model parameters. Table 4.1 shows the number of parameters of each layer of the proposed full model for the user identity recognition task on the UCI HAR Dataset. We can observe that the vast majority of the parameters belong to the fully connected layer 1. The remaining five models also have the majority of the parameters on their last fully connected layer on the 2D CNN branch. Although this layer has only 128 neurons, the huge size of the input vector which is 9,856 makes the parameter number of this layer considerably large. Input sizes of the previous layers in the 2D CNN branch are also quite large including the initial input which has dimensions of (11, 14, 9). However, the types of the previous layers require a small number of parameters independent from the input size, while the fully connected layer has parameters directly proportional to the input size. Therefore, fully connected layer has a large number of parameters while the previous layers in the 2D CNN branch do not.

The input of the LSTM branch has the dimensions of (128, 9) which is close to the input dimensions of the 2D CNN branch. However, the number of LSTM parameters is not dependent on the number of time steps which corresponds to the first element of the dimensionality. Therefore, LSTM layers have small number of parameters although the input dimensions are large. After the LSTM 3 layer, input dimensions are only 64. Hence, the parameter numbers of the layers after the LSTM 3 layer are quite small. We observe that the input sizes of LSTM and 2D CNN branches are similarly large. However, the types of the initial layers of both branches allow having small parameter numbers. Since the LSTM 3 layer has an output of a small size, the layers in the LSTM branch after the LSTM 3 layer have small input sizes. On the other hand, the 2D CNN branch does not have such a layer to decrease the input sizes. Therefore, all the layers in the 2D

Table 4.1: Number of parameters of the proposed full model layers for the user identity recognition task in processing the UCI HAR Dataset.

Layer (Type)	Output Shape	Parameter Number	Connected to
input layer (spectrogram signal)	(none, 11, 14, 9)	0	none
2D CNN layer	(none, 11, 14, 64)	5,248	input layer (spectrogram signal)
dropout layer 1	(none, 11, 14, 64)	0	2D CNN layer
flatten layer	(none, 9856)	0	dropout layer 1
dropout layer 5	(none, 9856)	0	flatten layer
fully connected layer 1	(none, 128)	1,261,696	dropout layer 5
input layer (raw signal)	(none, 128, 9)	0	none
LSTM layer 1	(none, 128, 64)	18,944	input layer (raw signal)
dropout layer 2	(none, 128, 64)	0	LSTM layer 1
LSTM layer 2	(none, 128, 64)	33,024	dropout layer 2
dropout layer 3	(none, 128, 64)	0	LSTM layer 2
LSTM layer 3	(none, 64)	33,024	dropout layer 3
dropout layer 4	(none, 64)	0	LSTM layer 3
dropout layer 6	(none, 64)	0	dropout layer 4
fully connected layer 2	(none, 128)	8,320	dropout layer 6
concatenate layer	(none, 256)	0	fully connected layer 1 and 2
softmax layer	(None, 30)	7,710	concatenate layer

Table 4.2: Number of parameters of the proposed transfer model layers for the user identity recognition task in processing the UCI HAR Dataset.

Layer (Type)	Output Shape	Parameter Number	Connected to
input layer (spectrogram signal)	(none, 11, 14, 64)	0	none
flatten layer	(none, 9856)	0	input layer (spectrogram signal)
dropout layer 5	(none, 9856)	0	flatten layer
fully connected layer 1	(none, 128)	1,261,696	dropout layer 5
input layer (raw signal)	(none, 64)	0	none
dropout layer 6	(none, 64)	0	input layer (raw signal)
fully connected layer 2	(none, 128)	8,320	dropout layer 6
concatenate layer	(none, 256)	0	fully connected layer 1 and 2
softmax layer	(none, 30)	7,710	concatenate layer

CNN branch have large input sizes. Types of certain layers allow keeping the number of parameters small even when the input size is large. However, the parameter number of the fully connected layer 1 is directly proportional to the input size. Therefore, large input size causes the fully connected layer 1 to have a large number of parameters. As we display in Table 4.2, although TL removes the initial layers of the model, the total number of parameters still remains large. This is because the fully connected layer 1 contains the vast majority of the parameters and it is not removed by TL since it is near the end of the network. As a result, the large input size of the fully connected layer 1 is the main reason for the large parameter number and the 2D CNN branch needs some additional layers before this layer which we can use to reduce the input size.

4.2 Methods to Reduce the Size of the Proposed Model

In this chapter, we investigate two different methods to reduce the size of the six proposed models. These are the pooling layers and weight quantization methods which we apply to the models after the training phase. Therefore, in this chapter, we do not conduct any training for the DL models but only apply the above mentioned methods on already trained models. Transfer learning which we have already applied in Chapter 3 has also some benefits for reducing the model size besides the major impact on reducing the training times. Therefore, we apply the above mentioned methods to the models after TL for the user identity recognition task. For the activity recognition task, we apply the same two methods without TL to prevent a noticeable decrease in activity recognition accuracy.

We explain the max pooling operation visually in Fig. 4.4. We use non-overlapping $n \times m$ sliding windows for this operation. We select the maximum value from the region of the feature map covered by the window and we dismiss the other values. This operation provides a down-sampled representation of the signal and decreases the size of the input. Max pooling layers are commonly

used after the 2D CNN layers to reduce the number of parameters. In Table 4.3, we observe that the 2×2 max pooling layer decreases the input size from (11, 14, 64) to (5, 7, 64). The first two values correspond to the height and the width of the input both of which are almost halved. On the other hand, the last value which corresponds to the channel size does not change after the max pooling layer. This is because traditional max pooling layers are used for feature-wise pooling. However, the channel size is also quite large and we can reduce it. For this purpose, we can use convolution with 1×1 filters. This operation is a linear weighting of the channel values and provides a projection on a space with lower dimensions as we can see in Fig. 4.5. It does not change the width and the height dimensions. In Table 4.3, we can see that the CNN layer with filter size 1×1 reduces the input size from (5, 7, 64) to (5, 7, 32). As a result of the dimensionality reduction by max pooling layer and channel-wise pooling, we reduce the input size of the fully connected layer 1 from 1,261,696 to 143,488. We reduce the total number of parameters of the model from 1,277,726 to 161,598. We achieve similar reduction in the number of parameters for all six models and summarize these in Tables 4.4–4.9. Tables 4.8 and 4.9 show that for activity recognition tasks, the amount of compression is still useful but it is smaller compared to the other models. This is because this operation reduces the number of parameters on the 2D CNN branch and the ratio of the number of the parameters on the 2D CNN branch for the activity recognition models is not as high as in the other models. Another reason is that we keep the size of the filter of max pooling layer smaller compared to the filters of the other models to prevent a noticeable decrease in the activity recognition accuracy.

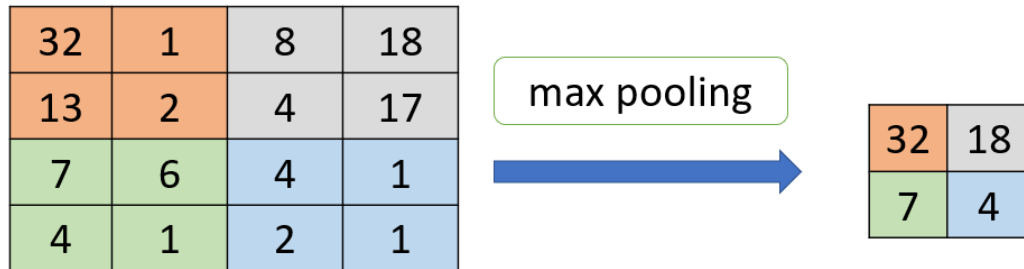


Figure 4.4: Max pooling operation.

Table 4.3: Number of parameters of the proposed transfer model with pooling layers for the user identity recognition task in processing the UCI HAR Dataset.

Layer (Type)	Output Shape	Parameter Number	Connected to
input layer (spectrogram signal)	(none, 11, 14, 64)	0	none
2D max pooling layer	(none, 5, 7, 64)	0	input layer (spectrogram signal)
1D CNN layer	(none, 5, 7, 32)	2,080	2D max pooling layer
flatten layer	(none, 1120)	0	1D CNN layer
dropout layer 5	(none, 1120)	0	flatten layer
fully connected layer 1	(none, 128)	143,488	dropout layer 5
input layer (raw signal)	(none, 64)	0	none
dropout layer 6	(none, 64)	0	input layer (raw signal)
fully connected layer 2	(none, 128)	8,320	dropout layer 6
concatenate layer	(none, 256)	0	fully connected layer 1 and 2
softmax layer	(none, 30)	7,710	concatenate layer

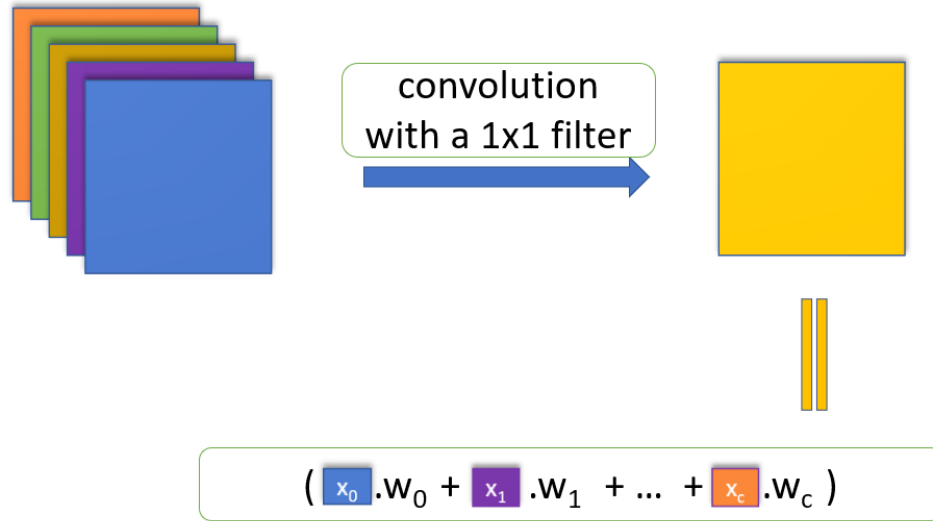


Figure 4.5: Convolution operation with a filter size of 1×1 .

The weight quantization method which we apply on the proposed models besides the max pooling layers is visually explained in Figs. 4.6 and 4.7. We store the model weights as 32-bit floating point numbers as default in computer memory in TensorFlow. However, such high precision may not always be necessary for the DL models and the same or similar performance can be sustained by representing the model weights with lower number of bits. TensorFlow supports this operation by TFLiteConverter package. Fig. 4.7 illustrates the implementation of this operation. The minimum and the maximum values of the model weights correspond to the 0 and 255 values in eight-bit encoding. The interval in between the minimum and the maximum values of 32-bit original numbers are divided into 256 equal subintervals. Then, an original 32-bit number is represented by the corresponding eight-bit number which is the closest interval value for the original 32-bit number. Hence, model weights are stored in this format and memory space requirement of the model is reduced by several times. While using the stored model during the inference phase, eight-bit encoded numbers are converted to 32-bit floating numbers. As illustrated in Fig. 4.7, there is some precision loss during this process. However, it does not affect the model performance noticeably as we will show in the next section.

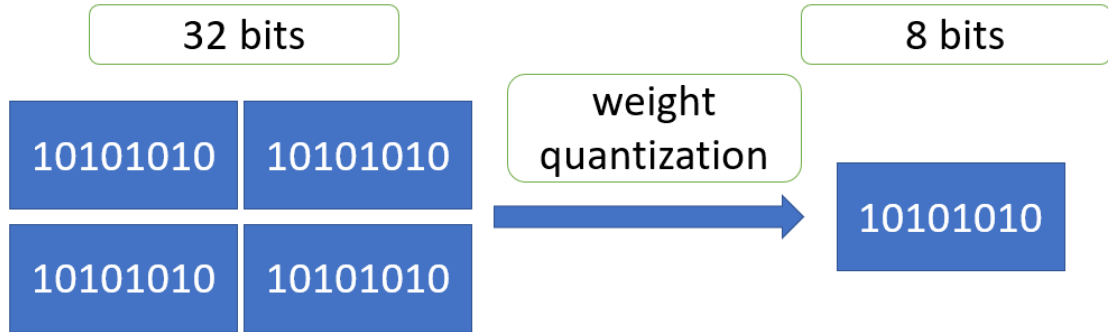


Figure 4.6: Quantization operation.

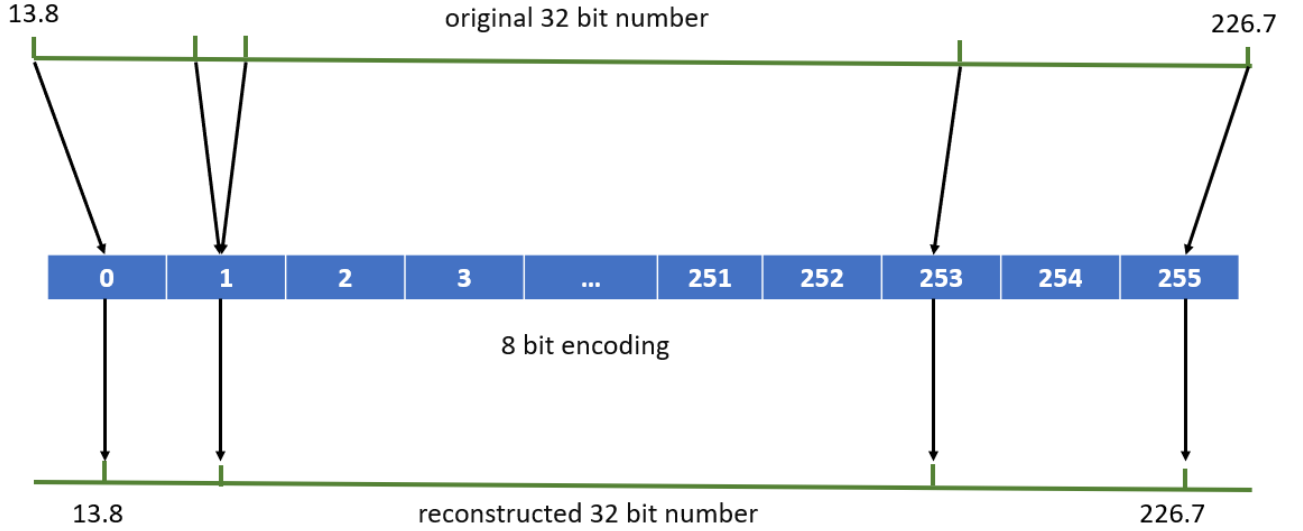


Figure 4.7: Eight-bit quantization.

4.3 Experimental Results

In the previous sections, we have introduced pooling layers and eight-bit weight quantization methods to be applied on the six proposed models in the previous chapters. These models are the ones which we have trained for the activity recognition, user identity recognition, and binary classification tasks for the UCI HAR and the DSA Datasets. We have also used TL methods for the training phase of those models. In this subsection, we investigate the effects of above mentioned methods on the model accuracy, model parameter number, and the model size.

In Chapter 3, we have shown that TL has significant benefits for reducing the training times of the DL models without a noticeable degradation in the classification accuracies for the user identity recognition and binary classification tasks. Tables 4.4 and 4.5 show that it has also benefits for reducing the sizes of the DL models deployed in the user identity recognition task. This is because TL eliminates the initial layers of the models and their parameters. For the binary classification models, TL does not provide a noticeable reduction in the number

of parameters. This is because binary classification is a simple task and even the full model has a small number of parameters. For the activity recognition task, TL degrades the classification accuracy by 1.99–5.38%. This is because the user identity recognition domain features are less generalizable compared to the activity recognition domain features. Therefore, we have not applied the TL method on the activity recognition models to keep the model accuracy high on the final model which would be used on user device.

Table 4.4: User identity recognition accuracies, number of parameters, and the model size for the UCI HAR Dataset.

Model	Accuracy (%)	Parameter Number	Model Size
fully trained	96.95±0.66	1,367,966	15.7 MB
transfer	94.38±0.09	1,277,726	14.7 MB
transfer + pooling	94.65±0.48	161,598	1.9 MB
transfer + pooling + quantization	94.58±0.55	161,598	159 KB

Table 4.5: User identity recognition accuracies, number of parameters, and the model size for the DSA Dataset.

Model	Accuracy (%)	Parameter Number	Model Size
fully trained	94.55±0.20	1,533,764	17.6 MB
transfer	96.94±0.16	950,408	10.9 MB
transfer + pooling	97.18±0.18	157,064	1.8 MB
transfer + pooling + quantization	97.04±0.11	157,064	151 KB

Pooling layers have provided significant reduction in the number of parameters for all six models without causing a noticeable accuracy degradation. Moreover, it improves the accuracy of some of the models by bringing the regularization effect. It reduces the size of the user identity recognition models by 7.7 times and six times and the size of the binary classifier model by 8.33 times and 9.49 times for the UCI HAR and the DSA Datasets, respectively. Pooling layers also provide significant reduction in the model sizes for the activity recognition models. However, the amount of reduction is smaller compared to the other models. The main reason is that the pooling layer provides reduction in the number of parameters on the 2D CNN branch of the proposed models and the ratio of the number of parameters in the 2D CNN branch to the LSTM branch is smaller in

the activity recognition models. Additionally, these models require smaller filter size to prevent noticeable degradation in the model accuracy. Therefore, the amount of reduction in the activity recognition model size is not as large as for the other models but it is still significant.

Table 4.6: Binary classifier accuracies, number of parameters, and the model size for the UCI HAR Dataset.

Model	Accuracy (%)	Parameter Number	Model Size
fully trained	99.91 $\pm$ 0.01	10,178	161 KB
transfer	99.97 $\pm$ 0.00	39,706	500 KB
transfer + pooling	99.97 $\pm$ 0.05	1,102	60 KB
transfer + pooling + quantization	99.99 $\pm$ 0.01	1,102	6 KB

Table 4.7: Binary classifier accuracies, number of parameters, and the model size for the DSA Dataset.

Model	Accuracy (%)	Parameter Number	Model Size
fully trained	98.76 $\pm$ 0.87	200,978	2.3 MB
transfer	99.14 $\pm$ 0.10	59,306	731 KB
transfer + pooling	99.24 $\pm$ 0.09	2,522	77 KB
transfer + pooling + quantization	99.21 $\pm$ 0.09	2,522	6 KB

In order to decrease the model sizes further, we apply the weight quantization method on the six proposed models. Model weights which are represented in 32-bit floating point values are converted to be represented in eight-bit integer values as illustrated in Fig. 4.7. This conversion decreases the model size by four times since the model weights are represented by four times smaller number of bits. Moreover, TensorFlow Lite library which we use for weight quantization operation saves the models in the protocol buffer format which is an efficient way of storing and transporting data. This file format enables to reduce the model size even further. As a result, Tables 4.4–4.9 show that the weight quantization method by using TensorFlow library and the protocol buffer format enables reduction in the model sizes by 10.6 to 12.8 times. Therefore, this method is the most effective one since it decreases the model sizes of all six proposed models by more than 10 times without noticeable degradation in the accuracy of any of the models.

Table 4.8: Activity recognition accuracies, number of parameters, and the model size for the UCI HAR Dataset.

Model	Accuracy (%)	Parameter Number	Model Size
fully trained	95.66 $\pm$ 0.63	767,110	8.9 MB
pooling	95.58 $\pm$ 0.83	296,102	3.5 MB
pooling + quantization	95.51 $\pm$ 0.80	296,102	331 KB

Table 4.9: Activity recognition accuracies, number of parameters, and the model size for the DSA Dataset.

Model	Accuracy (%)	Parameter Number	Model Size
fully trained	92.95 $\pm$ 0.47	1,794,638	20.5 MB
pooling	92.67 $\pm$ 0.43	1,261,098	14.5 MB
pooling + quantization	92.57 $\pm$ 0.37	1,261,098	1.2 MB

In this chapter, we have investigated the factors increasing the model sizes of the six proposed models. We have implemented pooling layers and eight-bit weight quantization methods to reduce the memory requirements of the proposed models. Implemented methods have reduced the sizes of the six proposed models by 17 to 116 times without any noticeable accuracy degradation which would enable deploying these models on memory limited wearable devices.

Chapter 5

Summary, Conclusions, and Future Work

In this thesis, we have investigated three significant issues related to the wearable motion sensor based HAR problem. These are accurate classification of the motion sensor signals, feature transfer between two different HAR-related tasks, and enabling the proposed models on memory limited devices. In the first chapter, we have presented a comprehensive literature review on the above mentioned issues.

In the second chapter, we have proposed a novel 2D CNN-LSTM architecture to enhance the classification accuracy of the activity signals. Proposed model comprises LSTM and 2D CNN branches that run in parallel. We have normalized/standardized the input signals and created spectrogram images for the LSTM and 2D CNN branches, respectively. Proposed architecture has enabled the extraction of a broader range of representative features which have resulted in higher activity recognition accuracy. We have compared the performance of the proposed model with five existing DL models which are the three single (1D CNN, 2D CNN, and LSTM) and two hybrid (standard 1D CNN-LSTM and the alternative 1D CNN-LSTM) models. We implemented all six models to process two publicly available datasets (UCI HAR and DSA Datasets). The proposed model proved to be superior in terms of the accuracy, f_1 score, recall, and precision

performance metrics, while offering an acceptable level of complexity compared to the other models.

In the third chapter, we have considered the user identity recognition task and implemented transfer learning between the activity recognition and the user identity recognition domains in both directions. We have shown that successful feature transfer between the two domains is possible and useful. The main benefit of feature transferring is the dramatic reduction in the training times of the models. Feature transfer from the activity domain to the user identity domain shortens the training process by a factor of 17 and nine for the UCI HAR and the DSA Datasets, respectively, without a noticeable accuracy degradation. On the other hand, feature transfer from the user identity domain to the activity domain shortens this process by 15.6 times and 8.7 times for the UCI HAR and the DSA Datasets, respectively, with quite a small accuracy degradation compared to the full models. We concluded that the activity recognition features are more generalizable compared to the user identity recognition features. Therefore, feature transfer from the activity domain to the user identity domain enables user identity recognition with almost the same accuracy as the fully trained model.

In the fourth chapter, we have studied how to enable the proposed six models which are the activity recognition, user identity recognition, and binary classification models for the UCI HAR and the DSA Datasets, on memory limited devices. Initially, we inspected the main factors increasing the model size. As a solution, we applied pooling layers and eight-bit weight quantization methods on the pretrained models. This way, we achieved 17 to 116 times reduction in the model sizes without a significant degradation in the classification accuracies.

We have proposed a novel 2D CNN-LSTM hybrid architecture in this thesis for wearable motion sensor based human activity and user identity recognition tasks. The proposed architecture has enabled enhanced classification accuracy for both tasks. Moreover, we have shown that feature transfer between these two tasks is beneficial and enables dramatic reduction in the training times. Finally, pooling layers and eight-bit weight quantization methods have enabled deploying the proposed models on memory limited devices.

As future work, there are several directions to extend the work that is presented in this thesis. To begin with, extracted features by our proposed architecture can be transferred accross different domains such as cross-body part, cross-sensor, cross-user, cross-dataset, and cross-activity. Therefore, it can be investigated if the extracted features have sufficient representation capacity for different problem domains. Furthermore, different domain adaptation methods such as GANs and autoencoders with appropriate objective functions can be employed to adapt the extracted features for different domains. Another potential research direction can be quantization of the model parameters with less than eight bits. Although our preliminary experiments resulted in significant accuracy degradation with two-bit quantization method, different solutions can be adopted to maintain model accuracy with less than eight-bit representations. Therefore, computational requirements of the models can be reduced further. Besides computational resources, wearable devices typically have also limited battery capacities. Hence, deployed models are required to be energy efficient. As another future research direction, energy efficiency of the proposed models can be investigated by measuring the energy consumption on embedded devices and novel methods can be proposed to deploy models with lower energy requirements.

Bibliography

- [1] N. Niknejad, W. B. Ismail, A. Mardani, H. Liao, and I. Ghani, “A comprehensive overview of smart wearables: The state of the art literature, recent advances, and future challenges,” *Engineering Applications of Artificial Intelligence*, vol.90, article no.103529, April 2020, <https://doi.org/10.1016/j.engappai.2020.103529>.
- [2] E. Ramanujam, T. Perumal, and S. Padmavathi, “Human activity recognition with smartphone and wearable sensors using deep learning techniques: A review,” *IEEE Sensors Journal*, vol.21, no.12, pp.13029–13040, June 2021, <https://doi.org/10.1109/jsen.2021.3069927>.
- [3] C. Dhiman and D. K. Vishwakarma, “A review of state-of-the-art techniques for abnormal human activity recognition,” *Engineering Applications of Artificial Intelligence*, vol. 77, pp.21–45, January 2019, <https://doi.org/10.1016/j.engappai.2018.08.014>.
- [4] M. Gil-Martín, R. San-Segundo, F. Fernández-Martínez, and J. Ferreiros-López, “Improving physical activity recognition using a new deep learning architecture and post-processing techniques,” *Engineering Applications of Artificial Intelligence*, vol.92, article no.103679, June 2020, <https://doi.org/10.1016/j.engappai.2020.103679>.
- [5] S. Zhang, Y. Li, S. Zhang, F. Shahabi, S. Xia, Y. Deng, and N. Alshurafa, “Deep learning in human activity recognition with wearable sensors: A review on advances,” *Sensors (MDPI)*, vol.22, no.4, article no.1476, February 2022, <https://doi.org/10.3390/s22041476>.

- [6] E. Haktanır and C. Kahraman, “A novel picture fuzzy CRITIC & REGIME methodology: Wearable health technology application,” *Engineering Applications of Artificial Intelligence*, vol.113, article no.104942, August 2022, <https://doi.org/10.1016/j.engappai.2022.104942>.
- [7] E. Lattanzi and V. Freschi, “Evaluation of human standing balance using wearable inertial sensors: A machine learning approach,” *Engineering Applications of Artificial Intelligence*, vol.94, article no.103812, September 2020, <https://doi.org/10.1016/j.engappai.2020.103812>.
- [8] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, “Deep learning for sensor-based activity recognition: A survey,” *Pattern Recognition Letters*, vol.119, pp.3–11, March 2019, <https://doi.org/10.1016/j.patrec.2018.02.010>.
- [9] K. Chen, D. Zhang, L. Yao, B. Guo, Z. Yu, and Y. Liu, “Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities,” *ACM Computing Surveys*, vol.54, no.4, article no.77, May 2022, <https://doi.org/10.1145/3447744>.
- [10] S. Mekruksavanich and A. Jitpattanakul, “LSTM networks using smartphone data for sensor-based human activity recognition in smart homes,” *Sensors (MDPI)*, vol.21, no.5, article no.1636, February 2021, <https://doi.org/10.3390/s21051636>.
- [11] C.-T. Yen, J.-X. Liao, and Y.-K. Huang, “Human daily activity recognition performed using wearable inertial sensors combined with deep learning algorithms,” *IEEE Access*, vol.8, pp.174105–174114, October 2020, <https://doi.org/10.1109/access.2020.3025938>.
- [12] K. Xia, J. Huang, and H. Wang, “LSTM-CNN architecture for human activity recognition,” *IEEE Access*, vol.8, pp.56855–56866, March 2020, <https://doi.org/10.1109/access.2020.2982225>.
- [13] O. Barut, L. Zhou, and Y. Luo, “Multitask LSTM model for human activity recognition and intensity estimation using wearable sensor data,” *IEEE Internet of Things Journal*, vol.7, no.9, pp.8760–8768, September 2020, <https://doi.org/10.1109/jiot.2020.2996578>.

- [14] S. Chung, J. Lim, K. J. Noh, G. Kim, and H. Jeong, "Sensor data acquisition and multimodal sensor fusion for human activity recognition using deep learning," *Sensors* (MDPI), vol.19, no.7, article no.1716, April 2019, <https://doi.org/10.3390/s19071716>.
- [15] T. Lv, X. Wang, L. Jin, Y. Xiao, and M. Song, "Margin-based deep learning networks for human activity recognition," *Sensors* (MDPI), vol.20, no.7, article no.1871, March 2020, <https://doi.org/10.3390/s20071871>.
- [16] N. Tufek, M. Yalcin, M. Altintas, F. Kalaoglu, Y. Li, and S. K. Bahadir, "Human action recognition using deep learning methods on limited sensory data," *IEEE Sensors Journal*, vol.20, no.6, pp.3101–3112, March 2020, <https://doi.org/10.1109/jsen.2019.2956901>.
- [17] R. Zhu, Z. Xiao, Y. Li, M. Yang, Y. Tan, L. Zhou, S. Lin, and H. Wen, "Efficient human activity recognition solving the confusing activities via deep ensemble learning," *IEEE Access*, vol.7, pp.75490–75499, June 2019, <https://doi.org/10.1109/access.2019.2922104>.
- [18] J. Sena, J. Barreto, C. Caetano, G. Cramer, and W. R. Schwartz, "Human activity recognition based on smartphone and wearable sensors using multiscale DCNN ensemble," *Neurocomputing*, vol.444, pp.226–243, July 2021, <https://doi.org/10.1016/j.neucom.2020.04.151>.
- [19] Z. Qin, Y. Zhang, S. Meng, Z. Qin, and K.-K. R. Choo, "Imaging and fusing time series for wearable sensor-based human activity recognition," *Information Fusion*, vol.53, pp.80–87, January 2020, <https://doi.org/10.1016/j.inffus.2019.06.014>.
- [20] J. Zhu, H. Chen, and W. Ye, "A hybrid CNN-LSTM network for the classification of human activities based on micro-Doppler radar," *IEEE Access*, vol.8, pp.24713–24720, February 2020, <https://doi.org/10.1109/access.2020.2971064>.
- [21] J. Park, R. J. Javier, T. Moon, and Y. Kim, "Micro-Doppler based classification of human aquatic activities via transfer learning of convolutional neural networks," *Sensors* (MDPI), vol.16, no.12, article no.1990, November 2016, <https://doi.org/10.3390/s16121990>.

- [22] Y. Kim and B. Toomajian, “Hand gesture recognition using micro-Doppler signatures with convolutional neural network,” *IEEE Access*, vol.4, pp.7125–7130, October 2016, <https://doi.org/10.1109/access.2016.2617282>.
- [23] Y. Kim and T. Moon, “Human detection and activity classification based on micro-Doppler signatures using deep convolutional neural networks,” *IEEE Geoscience and Remote Sensing Letters*, vol.13, no.1, pp.8–12, January 2016, <https://doi.org/10.1109/lgrs.2015.2491329>.
- [24] D. Ravi, C. Wong, B. Lo, and G.-Z. Yang, “A deep learning approach to on-node sensor data analytics for mobile or wearable devices,” *IEEE Journal of Biomedical and Health Informatics*, vol.21, no.1, pp.56–64, January 2017, <https://doi.org/10.1109/jbhi.2016.2633287>.
- [25] S. Yao, S. Hu, Y. Zhao, A. Zhang, and T. Abdelzaher, “DeepSense: A unified deep learning framework for time-series mobile sensing data processing,” *Proc. 26th Int. Conf. World Wide Web (WWW)*, 3–7 April 2017, Perth, Australia, pp.351–360, <https://doi.org/10.1145/3038912.3052577>.
- [26] R. Pravallika, D. Sreenivasarao, and S. K. Saheb, “Deep learning for human activity recognition using on-node sensors,” *International Journal of Recent Technology and Engineering*, vol.8, issue: 5, pp.607–614, January 2020, <https://doi.org/10.35940/ijrte.E5654.018520>.
- [27] I. A. Lawal and S. Bano, “Deep human activity recognition with localisation of wearable sensors,” *IEEE Access*, vol.8, pp.155060–155070, August 2020, <https://doi.org/10.1109/access.2020.3017681>.
- [28] C. Ito, X. Cao, M. Shuzo, and E. Maeda, “Application of CNN for human activity recognition with FFT spectrogram of acceleration and gyro sensors,” *Proc. ACM Int. Joint Conference and Int. Symposium on Pervasive and Ubiquitous Computing and Wearable Computers (UBICOMP)*, 8–12 October 2018, Singapore, Singapore, pp.1503–1510, <https://doi.org/10.1145/3267305.3267517>.

- [29] Y. Li, S. Zhang, B. Zhu, and W. Wang, “Accurate human activity recognition with multi-task learning,” *CCF Transactions on Pervasive Computing and Interaction*, vol.2, pp.288–298, December 2020, <https://doi.org/10.1007/s42486-020-00042-2>.
- [30] L. B. Pardo, D. B. Perez, and C. O. Uruñuela, “Detection of tennis activities with wearable sensors,” *Sensors* (MDPI), vol.19, no.22, article no.5004, November 2019, <https://doi.org/10.3390/s19225004>.
- [31] R. Mutegeki and D. S. Han, “A CNN-LSTM approach to human activity recognition,” *Proc. 2nd IEEE Int. Conf. Artificial Intelligence in Information and Communication (ICAIIIC)*, 19–21 February 2020, Fukuoka, Japan, pp.362–366, <https://doi.org/10.1109/icaaic48513.2020.9065078>.
- [32] S. Deep and X. Zheng, “Hybrid model featuring CNN and LSTM architecture for human activity recognition on smartphone sensor data,” *Proc. 20th Int. Conf. Parallel and Distributed Computing, Applications and Technologies (PDCAT)*, 5–7 December 2019, Gold Coast, QLD, Australia, pp.262–267, <https://doi.org/10.1109/pdcat46702.2019.00055>.
- [33] F. J. Ordóñez and D. Roggen, “Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition,” *Sensors* (MDPI), vol.16, no.1, article no.115, January 2016, <https://doi.org/10.3390/s16010115>.
- [34] S. Mekruksavanich and A. Jitpattanakul, “Smartwatch-based human activity recognition using hybrid LSTM network,” *Proc. IEEE Sensors Conference*, 25–28 October 2020, Rotterdam, Netherlands, <https://doi.org/10.1109/sensors47125.2020.9278630>.
- [35] H. Wang, J. Zhao, J. Li, L. Tian, P. Tu, T. Cao, Y. An, K. Wang, and S. Li, “Wearable sensor-based human activity recognition using hybrid deep learning techniques,” *Security and Communication Networks*, vol.2020, article ID: 2132138, July 2020, <https://doi.org/10.1155/2020/2132138>.
- [36] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” arXiv:1409.1259v2, 2014.

- [37] R. A. Hamad, L. Yang, W. L. Woo, and B. Wei, “Joint learning of temporal models to handle imbalanced data for human activity recognition,” *Applied Sciences*, vol.10, no.15, article no.5293, July 2020, <https://doi.org/10.3390/app10155293>.
- [38] T. Huynh-The, C.-H. Hua, N. A. Tu, and D.-S. Kim, “Physical activity recognition with statistical-deep fusion model using multiple sensory data for smart health,” *IEEE Internet of Things Journal*, vol.8, no.3, pp.1533–1543, February 2021, <https://doi.org/10.1109/jiot.2020.3013272>.
- [39] L. Peng, L. Chen, Z. Ye, and Y. Zhang, “AROMA: A deep multi-task learning based simple and complex human activity recognition method using wearable sensors,” *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol.2, no.2, article no.74, June 2018, <https://doi.org/10.1145/3214277>.
- [40] D. Mukherjee, R. Mondal, P. K. Singh, R. Sarkar, and D. Bhattacharjee, “EnsemConvNet: A deep learning approach for human activity recognition using smartphone sensors for healthcare applications,” *Multimedia Tools and Applications*, vol.79, no.41, pp.31663–31690, August 2020, <https://doi.org/10.1007/s11042-020-09537-7>.
- [41] N. Hernandez, J. Lundström, J. Favela, I. McChesney, and B. Arnrich, “Literature review on transfer learning for human activity recognition using mobile and wearable devices with environmental technology,” *SN Computer Science*, vol.1, no.2, pp.1–16, February 2020, <https://doi.org/10.1007/s42979-020-0070-4>.
- [42] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, “A survey on deep transfer learning,” *Proc. Int. Conf. on Artificial Neural Networks (ICANN)*, Rhodes, Greece, 4–7 October 2018, pp.270–279, https://doi.org/10.1007/978-3-030-01424-7_27.
- [43] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, “A comprehensive survey on transfer learning,” *Proc. IEEE*, vol.109, no.1, pp.43–76, June 2020, <https://doi.org/10.1109/JPROC.2020.3004555>.

- [44] X. Du, K. Farrahi, and M. Niranjan, “Transfer learning across human activities using a cascade neural network architecture,” *Proc. 23rd Int. Symposium on Wearable Computers (ISWC)*, London, United Kingdom, 9–13 September 2019, <https://doi.org/10.1145/3341163.3347730>.
- [45] J. Wang, Y. Chen, L. Huz, X. Peng, and P. S. Yu, “Stratified transfer learning for cross-domain activity recognition,” *IEEE Int. Conf. on Pervasive Computing and Communications (PerCom)*, Athens, Greece, 19–23 March 2018, pp.1–10, <https://doi.org/10.1109/PERCOM.2018.8444572>.
- [46] A. R. Sanabria, F. Zambonelli, S. Dobson, and J. Ye, “ContrasGAN: Unsupervised domain adaptation in human activity recognition via adversarial and contrastive learning,” *Pervasive and Mobile Computing*, vol.78, article no.101477, November 2021, <https://doi.org/10.1016/j.pmcj.2021.101477>.
- [47] A. Akbari and R. Jafari, “Transferring activity recognition models for new wearable sensors with deep generative domain adaptation,” *Proc. 18th Int. Conf. Information Processing in Sensor Networks (IPSN)*, Montreal, Quebec, Canada, 16–18 April 2019, <https://doi.org/10.1145/3302506.3310391>.
- [48] E. Soleimani and E. Nazerfard, “Cross-subject transfer learning in human activity recognition systems using generative adversarial networks,” *Neurocomputing*, vol.426, pp.26–34, February 2021, <https://doi.org/10.1016/j.neucom.2020.10.056>.
- [49] D. H. Hu, V. W. Zheng, and Q. Yang, “Cross-domain activity recognition via transfer learning,” *Pervasive and Mobile Computing*, vol.7, no.3, pp.344–358, June 2011, <https://doi.org/10.1016/j.pmcj.2010.11.005>.
- [50] M. A. A. H. Khan and N. Roy, “TransAct: Transfer learning enabled activity recognition,” *Proc. IEEE Int. Conf. on Pervasive Computing and Communications Workshops (PerCom)*, Kona, HI, U.S.A., 13–17 March 2017, pp.545–550, <https://doi.org/10.1109/PERCOMW.2017.7917621>.
- [51] Y. Liu, M. Peng, G. Shou, Y. Chen, and S. Chen, “Toward edge intelligence: Multiaccess edge computing for 5G and internet of things,”

- IEEE Internet of Things Journal*, vol.7, no.8, pp.6722–6747, August 2020, <https://doi.org/10.1109/JIOT.2020.3004500>.
- [52] F. Wang, M. Zhang, X. Wang, X. Ma, and J. Liu, “Deep learning for edge computing applications: A state-of-the-art survey,” *IEEE Access*, vol.8, pp.58322–58336, April 2020, <https://doi.org/10.1109/ACCESS.2020.2982411>.
- [53] Y. Tang, Q. Teng, L. Zhang, F. Min, and J. He, “Layer-wise training convolutional neural networks with smaller filters for human activity recognition using wearable sensors,” *IEEE Sensors Council*, vol.21, no.1, pp.581–592, January 2021, <https://doi.org/10.1109/JSEN.2020.3015521>.
- [54] P. Agarwal and M. Alam, “A lightweight deep learning model for human activity recognition on edge devices,” *Proc. Int. Conf. on Computational Intelligence and Data Science (ICCIDS)*, Gurugram, India, 6–7 September 2019, <https://doi.org/10.1016/j.procs.2020.03.289>.
- [55] W. Huang, L. Zhang, W. Gao, F. Min, and J. He, “Shallow convolutional neural networks for human activity recognition using wearable sensors,” *IEEE Transactions on Instrumentation and Measurement*, vol.70, pp.1–11, June 2021, <https://doi.org/10.1109/TIM.2021.3091990>.
- [56] M. Gholamrezaii and S. AlModarresi, “A time-efficient convolutional neural network model in human activity recognition,” *Multimedia Tools and Applications*, vol.80, no.13, pp.19361–19376, February 2021, <https://doi.org/10.1007/s11042-020-10435-1>.
- [57] D. Ravi, C. Wong, B. Lo, and G. Yang, “Deep learning for human activity recognition: A resource efficient implementation on low-power devices,” *Proc. IEEE 13th Int. Conf. on Wearable and Implantable Body Sensor Networks (BSN)*, San Francisco, California, U.S.A., 14–17 June 2016, pp.71–76, <https://doi.org/10.1109/BSN.2016.7516235>.
- [58] X. Li, Y. Wang, B. Zhang, and J. Ma, “PSDRNN: An efficient and effective HAR scheme based on feature extraction and deep learning,” *IEEE Transactions on Industrial Informatics*, vol.16, no.10, pp.6703–6713, October 2020, <https://doi.org/10.1109/TII.2020.2968920>.

- [59] M. Edel and E. Köppe, “Binarized-BLSTM-RNN based human activity recognition,” *Proc. Int. Conf. on Indoor Positioning and Indoor Navigation (IPIN)*, Alcalá de Henares, Madrid, Spain, 4–7 October 2016, pp.1–7, <https://doi.org/10.1109/IPIN.2016.7743581>.
- [60] Z. Yang, O. I. Raymond, C. Zhang, Y. Wan, and J. Long, “DFTer-Net: Towards 2-bit dynamic fusion networks for accurate human activity recognition,” *IEEE Access*, vol.6, pp.56750–56764, October 2018, <https://doi.org/10.1109/ACCESS.2018.2873315>.
- [61] J. L. Reyes-Ortiz, D. Anguita, L. Oneto, and X. Parra, “Smartphone-based recognition of human activities and postural transitions data set,” *UCI Machine Learning Repository*, School Inf. Comput. Sci., Univ. California at Irvine, Irvine, CA, U.S.A., 2015, available online: <http://archive.ics.uci.edu/ml/datasets/Smartphone-Based+Recognition+of+Human+Activities+and+Postural+Transitions>
- [62] D. Anguita, A. Ghio, L. Oneto, X. Parra Perez, and J. L. Reyes-Ortiz, “A public domain dataset for human activity recognition using smartphones,” *Proc. 21st European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, Bruges, Belgium, 24–26 April 2013, pp.437–442.
- [63] J. L. Reyes-Ortiz, L. Oneto, A. Samà, X. Parra, and D. Anguita, “Transition-aware human activity recognition using smartphones,” *Neurocomputing*, vol.171, pp.754–767, August 2015.
- [64] K. Altun and B. Barshan, “Daily and sports activities dataset,” *IEEE Data Port*, February 2019, <https://doi.org/10.21227/at1v-6f84>.
- [65] K. Altun and B. Barshan, “Daily and sports activities dataset,” *UCI Machine Learning Repository*, School Inf. Comput. Sci., Univ. California at Irvine, Irvine, CA, U.S.A., 2013, available online: <http://archive.ics.uci.edu/ml/datasets/Daily+and+Sports+Activities>.

- [66] A. Yurtman and B. Barshan, “Activity recognition invariant to sensor orientation with wearable motion sensors,” *Sensors* (MDPI), vol.17, no.8, article no.1838, August 2017, <https://doi.org/10.3390/s17081838>.
- [67] A. Yurtman, B. Barshan, and B. Fidan, “Activity recognition invariant to wearable sensor unit orientation using differential rotational transformations represented by quaternions,” *Sensors* (MDPI), vol.18, no.8, article no.2725, August 2018, (Special Issue on Data Analytics and Applications of Wearable Sensors in Healthcare), <https://doi.org/10.3390/s18082725>.
- [68] B. Barshan and A. Yurtman, “Classifying daily and sports activities invariantly to the positioning of wearable motion sensor units,” *IEEE Internet of Things Journal*, vol.7, no.6, pp.4801–4815, June 2020, <https://doi.org/10.1109/jiot.2020.2969840>.
- [69] A. Yurtman, B. Barshan, and S. Redif, “Position invariance for wearables: Interchangeability and single-unit usage via machine learning,” *IEEE Internet of Things Journal*, vol.8, no.10, pp.8328–8342, May 2021, <https://doi.org/jiot.2020.3044754>.
- [70] A. V. Oppenheim, R. W. Schaffer, and J. R. Buck, *Discrete-Time Signal Processing*, Prentice Hall: Upper Saddle River, NJ, U.S.A., 2nd ed., 1999 (see Ch.10, pp.714–722 on the time-dependent Fourier transform).
- [71] J. Snoek, H. Larochelle, and R. P. Adams, “Practical Bayesian optimization of machine learning algorithms,” in: *Advances in Neural Information Processing Systems: Proc. 25th Int. Conf. Neural Information Processing Syst. (NIPS)*, Lake Tahoe, NV, U.S.A., 3–6 December 2012, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), vol.25, Curran Associates Inc.: Red Hook, NY, U.S.A., 2012.
- [72] S. Mekruksavanich and A. Jitpattanakul, “Biometric user identification based on human activity recognition using wearable sensors: An experiment using deep learning models,” *Electronics*, vol.10, no.3, article no.308, January 2021, <https://doi.org/10.3390/electronics10030308>.

Appendix A

Hyper-parameters

Table A.I: Optimized hyper-parameters of the 1D CNN in processing the UCI HAR Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer 1 filter number	255	[4, 256]
CNN layer 2 filter number	251	[4, 256]
CNN layer 3 filter number	62	[4, 256]
CNN layer 4 filter number	155	[4, 256]
CNN layer 1 filter width	4	[3, 7]
CNN layer 2 filter width	7	[3, 7]
CNN layer 3 filter width	6	[3, 7]
CNN layer 4 filter width	5	[3, 7]
fully connected layer neuron number	242	[8, 512]
dropout layer 1 probability	0.33500743	[0.1, 0.9]
dropout layer 2 probability	0.21856031	[0.1, 0.9]
dropout layer 3 probability	0.12632442	[0.1, 0.9]
dropout layer 4 probability	0.70007760	[0.1, 0.9]
learning rate	$4.40541658 \times 10^{-5}$	$[10^{-6}, 0.1]$

Table A.II: Optimized hyper-parameters of the 2D CNN in processing the UCI HAR Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer 1 filter number	7	[4, 256]
CNN layer 2 filter number	23	[4, 256]
CNN layer 1 filter width	4	[3, 7]
CNN layer 2 filter width	7	[3, 7]
fully connected layer 1 neuron number	72	[8, 512]
fully connected layer 2 neuron number	493	[8, 512]
dropout layer 1 probability	0.11793672	[0.1, 0.9]
dropout layer 2 probability	0.14386716	[0.1, 0.9]
dropout layer 3 probability	0.79227985	[0.1, 0.9]
learning rate	0.00354435	$[10^{-6}, 0.1]$

Table A.III: Optimized hyper-parameters of the LSTM in processing the UCI HAR Dataset.

Hyper-Parameter	Optimum Value	Search Interval
LSTM layer 1 neuron number	474	[8, 512]
LSTM layer 2 neuron number	362	[8, 512]
LSTM layer 3 neuron number	156	[8, 512]
fully connected layer neuron number	275	[8, 512]
dropout layer 1 probability	0.13905174	[0.1, 0.9]
dropout layer 2 probability	0.28673553	[0.1, 0.9]
dropout layer 3 probability	0.51215028	[0.1, 0.9]
learning rate	0.00011890	$[10^{-6}, 0.1]$

Table A.IV: Optimized hyper-parameters of the standard 1D CNN-LSTM in processing the UCI HAR Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer 1 filter number	180	[4, 256]
CNN layer 2 filter number	25	[4, 256]
CNN layer 1 filter width	6	[3, 7]
CNN layer 2 filter width	7	[3, 7]
LSTM layer neuron number	506	[8, 512]
fully connected layer neuron number	309	[8, 512]
dropout layer 1 probability	0.25003751	[0.1, 0.9]
dropout layer 2 probability	0.40697846	[0.1, 0.9]
dropout layer 3 probability	0.48942447	[0.1, 0.9]
dropout layer 4 probability	0.44661285	[0.1, 0.9]
learning rate	0.00015457	$[10^{-6}, 0.1]$

Table A.V: Optimized hyper-parameters of the alternative 1D CNN-LSTM in processing the UCI HAR Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer 1 filter number	154	[4, 256]
CNN layer 2 filter number	161	[4, 256]
CNN layer 3 filter number	96	[4, 256]
CNN layer 4 filter number	92	[4, 256]
CNN layer 1 filter width	7	[3, 7]
CNN layer 2 filter width	6	[3, 7]
CNN layer 3 filter width	4	[3, 7]
CNN layer 4 filter width	6	[3, 7]
LSTM layer 1 neuron number	63	[8, 512]
LSTM layer 2 neuron number	291	[8, 512]
LSTM layer 3 neuron number	101	[8, 512]
fully connected layer 1 neuron number	493	[8, 512]
fully connected layer 2 neuron number	239	[8, 512]
dropout layer 1 probability	0.23502093	[0.1, 0.9]
dropout layer 2 probability	0.25219255	[0.1, 0.9]
dropout layer 3 probability	0.13254391	[0.1, 0.9]
dropout layer 4 probability	0.18250190	[0.1, 0.9]
dropout layer 5 probability	0.35352521	[0.1, 0.9]
dropout layer 6 probability	0.17141114	[0.1, 0.9]
dropout layer 7 probability	0.42956279	[0.1, 0.9]
learning rate	0.00033614	$[10^{-6}, 0.1]$

Table A.VI: Optimized hyper-parameters of the 1D CNN in processing the DSA Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer filter number	83	[4, 256]
CNN layer filter width	5	[3, 7]
fully connected layer 1 neuron number	498	[8, 512]
fully connected layer 2 neuron number	341	[8, 512]
dropout layer probability	0.86982345	[0.1, 0.9]
learning rate	0.00172552	$[10^{-6}, 0.1]$

Table A.VII: Optimized hyper-parameters of the 2D CNN in processing the DSA Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer filter number	27	[4, 256]
CNN layer filter width	3	[3, 7]
fully connected layer 1 neuron number	66	[8, 512]
fully connected layer 2 neuron number	141	[8, 512]
dropout layer probability	0.26342746	[0.1, 0.9]
learning rate	0.00026890	$[10^{-6}, 0.1]$

Table A.VIII: Optimized hyper-parameters of the LSTM in processing the DSA Dataset.

Hyper-Parameter	Optimum Value	Search Interval
LSTM layer neuron number	157	[8, 512]
fully connected layer 1 neuron number	122	[8, 512]
fully connected layer 2 neuron number	391	[8, 512]
dropout layer probability	0.17112627	[0.1, 0.9]
learning rate	0.00306971	$[10^{-6}, 0.1]$

Table A.IX: Optimized hyper-parameters of the standard 1D CNN-LSTM in processing the DSA Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer filter number	66	[4, 256]
CNN layer filter width	5	[3, 7]
LSTM layer neuron number	104	[8, 512]
fully connected layer 1 neuron number	68	[8, 512]
fully connected layer 2 neuron number	21	[8, 512]
dropout layer probability	0.10679134	[0.1, 0.9]
learning rate	0.00212826	$[10^{-6}, 0.1]$

Table A.X: Optimized hyper-parameters of the alternative 1D CNN-LSTM in processing the DSA Dataset.

Hyper-Parameter	Optimum Value	Search Interval
CNN layer filter number	76	[4, 256]
CNN layer filter width	7	[3, 7]
LSTM layer neuron number	58	[8, 512]
fully connected layer 1 neuron number	127	[8, 512]
fully connected layer 2 neuron number	110	[8, 512]
fully connected layer 3 neuron number	392	[8, 512]
dropout layer probability	0.55504952	[0.1, 0.9]
learning rate	0.00074461	$[10^{-6}, 0.1]$